Workshop on Virtual Reality Interaction and Physical Simulation VRIPHYS (2011)
J. Bender, K. Erleben, and E. Galin (Editors)

Topological Operations for Geomorphological Evolution

R. Bezin[1], B. Crespin[1], X.Skapin[2], O. Terraz[1], P. Meseure[2]

[1]XLIM - UMR CNRS 6172, Université de Limoges, France
[2]XLIM - UMR CNRS 6172, Université de Poitiers, France

Abstract

Geomorphological processes sculpt the shape of our everyday landscapes and must therefore be simulated to generate plausible digital landscapes. In particular, topological changes must be taken into account during the formation of complex geometries such as natural arches, bridges or tunnels. We present a novel approach to simulate the geomorphological evolution of a 3D terrain represented as a set of volumes stored in a topological model, and describe a set of atomic operations to handle topological events in a robust way. These operations form the basis to successfully implement more complex evolution scenarios in a modelling software based on generalized maps, which could be used to reduce the storage needed by other methods relying on voxel grids or layered data structures.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Line and curve generation I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Physically based modeling

1. Introduction

In this paper we address the problem of handling topological events during the geomorphological evolution of a 3D terrain. The simulation of geomorphological processes finds its main application in the semi-automatic authoring of visually realistic landforms, where a digital artist is able to simulate a specific phenomenon on a 3D terrain and control the result dynamically. Such phenomena include erosion processes (due to rain, wind, river flows, etc.) which contribute to shape the geometry of our everyday landscapes and add realism to digital environments found in computer-generated films, persistent cyberworlds and other types of video games.

Geomorphological evolution can generate very complex and breathtaking geometries in nature, for example concave shapes such as caverns, hoodoos and overhangs. Furthermore, even the topology of a terrain can be modified over the years with the emergence of natural arches, bridges, tunnels. In digital applications these topological events must be detected and handled correctly in order to maintain a coherent geometry during the simulation, as illustrated in Fig. 1.

Some geomorphological processes also depend on the

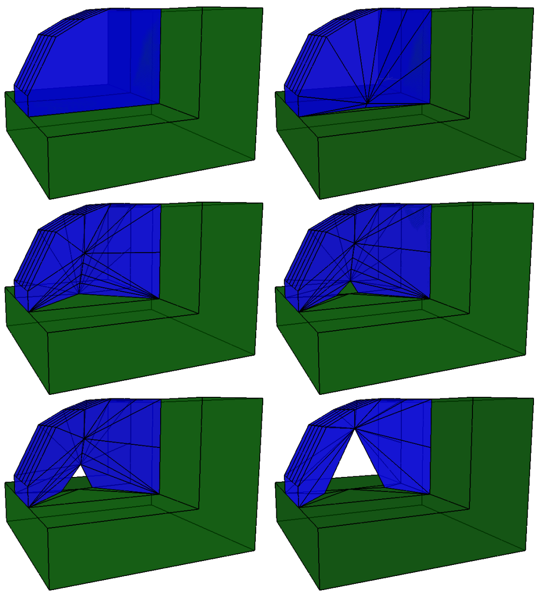
Figure 1: *Formation of an arch by progressively eroding the base of the blue volume.*

DOI: 10.2312/PE/vriphys/vriphys11/139-148

type of various materials which constitute the soil. For example splash erosion caused by raindrops will have a different impact on soil depending on whether it is made of sand, clay or granite material: this is called *differential* erosion. Again this can generate complex geometries if soft rock layers with high erosion rates are completely washed away, exposing more resistant igneous rocks beneath. Thus, for computer simulation purposes one must be able to instantiate different geological layers into the soil and handle differential erosion scenarios.

Existing works have a major limitation due to the significant memory consumption induced by the representation of 3D terrains as voxel grids or layered data structures. Although different types of material and complex geometries can be taken into account, single material volumes must necessarily be discretized, with the notable exception of [TJ10] which relies on unorganized points but requires expensive triangulations on the fly.

Relying on previous works restricted to two dimensions [LSM08], we propose to use models such as *generalized maps* instead of discretization-based approaches in order to detect and handle topological events. In this paper we focus on topological operations that maintain the coherence of the subsoil structure at each simulation step of a dynamic geomorphological process. Our goal in this paper is not to generate visually realistic scenes but rather to show a proof of concept on how complex scenarios involving topology changes can be easily reduced to a set of successive atomic topological operations. We also show how these operations can be implemented in a topological modelling software.

This remainder of the paper is organized as follows. The next section discusses related works and introduces topological models. Section 3 describes the atomic operations applied to the topological model of a 3D terrain, based on topological events. Implementations of complex geomorphological processes in a modelling software are presented in Sect. 5, and Sect. 6 gives our conclusions and ideas for future work.

2. Related works

2.1. 3D Terrain representation for geomorphological applications

Existing works related to our paper are mostly found in the field of physically-based terrain modeling. Weathering, hydraulic erosion and other natural phenomena can be simulated to generate visually realistic terrains for various applications, by iteratively modifying an initial terrain representation. Terrain representation is therefore one of the most critical parts, but other aspects must also be taken into account in order to produce plausible results, such as using efficient stream flow models in the case of hydraulic erosion simulation.

Geological representations of a terrain usually include all different elements that constitute the soil as shown of Fig. 2a. These elements are defined by different attributes such as 3D geometry, type of rock, durability, porosity, permeability, water or other fluids saturation, etc. They have an influence on geomorphological processes at different spatial and temporal scales, from tectonics events that shape continents to biological processes such as animal burrows. In the case of hydraulic or aeolian processes, *differential* erosion can be observed on soils composed of rocks with nonuniform durability.

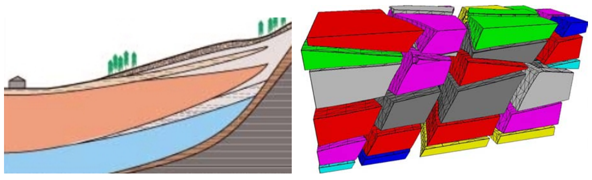
(a) (b)

Figure 2: *(a) Standard 2D geological representation of a real terrain. (b) Exploded view of the 3D model of a synthetic terrain.*

Most existing works in computer graphics focus on 3D terrain modelling applied to erosion simulation, where 3D geometry and erosion rate are the most important attributes. Terrain representation can be limited only to the surface, using heightfields [Nag98, ZSTR07, KBKO09, BPC*10]. This straightforward approach exhibits serious limitations: erosion is usually limited to vertical deformations, since lateral erosion may create self intersections or other topological inconsistencies. As a result, even if visually impressing results are obtained in [KBKO09] using a SPH fluid simulation, concavities or holes cannot be handled which forbids representing arches or caverns for examples.

Since the last decade several authors investigated voxel-based representations. In this case each voxel stores information about the type and amount of material it contains, which makes it possible to model real soils including different types of rocks and erosion rates as in [DEJ*99, OF99]. Explicit connections between rock volumes are represented in [IFMC03]; these cohesive connections can break under stress forces to model fractures or faults at different scales. Solid bodies are represented by level sets in [WCMT07] to simulate different phenomena such as sand erosion or acid corrosion, which may change the topology and create fractures. Since voxels can also contain air (or simply be marked as empty), the 3D voxel grid can model any complex geometry including concavities and holes, which allows to carve caves into the rock [BTHB05] or produce hoodoo shapes [BFO*07, MMJM09]. This ability to simulate real phenomena is demonstrated in [CSC*10] by comparing real experiments and an erosion simulation on a 3D terrain with an SPH model.

The major drawback of voxel-based representations is their excessive memory and computational cost, even if only active voxels need to be stored [BFO*07]. *Layered data structures* can be employed to overcome this limitation by combining heightmaps and voxels: for a given point on the map, instead of a set of voxels one stores a set of layers defined by their type and height. Consecutive voxels of the same type will thus be considered as a single layer, however care must be taken to adapt existing erosion/deposition algorithms as in [BF01, vBBK08]. This approach can also be implemented on GPU [NWD05, MDH07, KBKO09]. Layered materials that include air, water, rocks, sand and bedrock are used in [PGMG09] to create very complex geometries such as arches, rock and sand piles. Unfortunately this approach still needs a significant amount of memory, and needs a polygonization step to generate triangles used for rendering.

As shown in Fig. 2b, an alternative approach would be to directly represent the subsoil by a B-Rep representation as in [BSP*04]. Boolean operations on a volumetric model are implemented in [ZLL07] to simulate excavation, drilling, etc. However boolean operations are not well-suited to erosion simulations that usually imply very small successive displacements. A volumetric representation is computed in [TJ10] from a set of unorganized point clouds representing the terrain surface by a dynamic Delaunay Deformable Mesh [PB07]. This approach is able to produce arches and other complex 3D terrain geometries by using implicitly-defined rock durability, while reducing memory requirements since a point-based representation scales with the surface area rather than the volume of subsoil elements. Nevertheless it requires computing a Delaunay triangulation at each simulation step in order to maintain a correct labeling of volumes and visualize the surface, and representing different types of rock is also not straightforward with this approach.

After reviewing these works we conclude that even if volumetric models overcome memory limitations of voxels or layered data structures representations, investigation is still needed to simulate geomorphological processes that involve subvolumes of different material types, and handle interactions between these subvolumes. Subvolumes are obtained by subdividing “real” geological layers, and contain a single type of material. This macro-level representation will be used during the geomorphological simulation to prevent interpenetrations between subvolumes and detect *topological* changes, as illustrated in Fig. 1.

2.2. Topological models

As mentioned above, we need a topological model able to split volumes and by extension represent the dynamic evolution of a 3D terrain, as well as detect topological inconsistencies.

We choose to use a model based on *generalized maps* [Lie91, Lie94], an extension of combinatorial maps [Edm60] that can represent the topological structure of a subdivided volume as the one shown in Fig. 2b. This topological model is based on a compact and uniform representation which simplifies definitions and evolution algorithms in any dimension. It also makes it easier to implement mesh modifications such as triangulation, insertion of vertices, edges, faces and volumes, as well as neighborhood search in constant time. A similar choice is made in [LSM08] to simulate the evolution of a 2D terrain, which limits the number of possible cases. Generalized maps were successfully used for other purposes in the field of physically- or biologically-inspired simulations such as collision detection [JCD10] or simulated growth [TGM*08].

2.3. Our Contributions

The main contributions of our work are:

- **Atomic evolution operations** derived from a set of simple topological operations such as edge contraction, vertex insertion (section 3).
- **A generic approach to simulate the evolution of a 3D subdivided volume.** Our work relies on data structures and algorithms as generic as possible. We show how to apply animation and deformation processes to a 3D terrain, and how these operations can be implemented within the generalized maps formalism (section 4).
- **Complex evolution scenarios** are implemented using only atomic operations to create natural arches, tunnels or other concave geometries arising from geomorphological processes (section 4).

The explicit B-rep definition of the surface embedded in the topological model is another advantage over the methods listed in the previous section.

3. Topological operations

The 3D structured terrain is subdivided into several different volumes as shown in Fig. 2, and its boundary representation is a collection of faces.

Our geomorphological simulation can be divided in three main steps:

- At each time step, geomorphological simulation impacts terrain boundaries (*e.g.* animated water particles hitting the ground).
- Vertex displacements are computed depending on the direction and magnitude of impacts and the properties of the impacted volume (material, erosion rate, etc.). We assume that those displacements are linear only since computing non-linear trajectories would be too time-consuming to achieve interactive rates.
- Vertex displacements may lead to topological inconsistencies that we must detect and settle before the next time step by modifying the topological model.

Note that our work is a proof of concept only and does not rely on any precise description of geomorphological phenomena. We assume that the parameters of those phenomena are associated with the geometrical model to control the geometrical evolution. But this association falls outside the scope of this paper: we only focus on the topological evolution resulting from geometrical changes. Therefore, even if a specific geomorphological simulation drives the topology evolution through geometry, the atomic topological updates needed to ensure the model consistency could be reused for any other kind of geomorphological simulation. In the remainder of this section we present the operations to process topological inconsistencies. After describing the general context of our approach regarding geomorphological evolution, we describe the four atomic operations that will be used in section 4 to implement more complex evolution scenarios.

3.1. General context and incoherence detection

We assume that our structural model is able to represent both geometrical and topological information, and more specifically neighboring relationships between vertices, edges, faces and volumes: edges using a vertex, faces bordering an edge, volumes adjacent to a volume. We consider only planar faces to simplify collision detection; non-planar faces are triangulated in a preprocessing step.

Vertex displacements are stored as straight line segments P_tP_{t+1} in an event priority queue, sorted according to the time of collision with an edge or a face bordering a volume. When the first item has been processed and removed from the queue, we must recalculate the time of collision for remaining items in case colliding edges or faces have been modified. If the vertex crosses an edge or face at collision point P_I, we also must re-inject displacement P_IP_{t+1} in the event queue to take possible collisions with other volumes into account. This process is repeated until the event queue is empty after each simulation step.

Finally, even if vertex displacement can follow any direction, we restrict ourselves to displacements that imply a loss of volume, as it is the most common case with erosion simulation. This *contextual constraint* means that sedimentation or other phenomena that would add some volume to the 3D terrain are forbidden, nevertheless such cases are symmetric to the erosion case.

A vertex P in the 3D terrain always belongs to several edges and faces, and to one or more volumes, found in constant time thanks to the topological model. During its displacement from location P_t to P_{t+1} it may collide with different elements. Since displacements can not lead to volume growth, P can either slide on a face (and possibly collide with a vertex or an edge belonging to the boundary of this face), or move into a volume (and possibly collide with a face belonging to this volume). In this latter case, either the displacement stops inside the volume, or P collides with the volume boundary; in order to limit the intersection case study, we assume that P always collides with a face by using a symbolic perturbation scheme to avoid collisions with other vertices or edges.

Thus, detecting inconsistencies rely on four potential cases, described in the following sections:

- P moves but belongs to more than one volume, which implies a preprocessing step (section 3.2)
- P slides along a face and and collides with a vertex or an edge (section 3.3)
- P collides with a face, which can either create a hole (section 3.4), or “pinch” a volume and split it in two parts (section 3.5)

In all other cases, the displacement of P does not involve any topological change and can be safely removed from the event queue.

3.2. Vertex belonging to several volumes

Due to the contextual constraint mentioned in the previous section, displacing a vertex belonging to several volumes may not lead to topological problems but is not allowed if it implies volume growth. Therefore we need a specific preprocessing step in order to differentiate the behaviour of the vertex for each volume it belongs to.

Consider for example the initial configuration shown in Fig. 3a, with a vertex belonging to two different volumes, each with its own properties such as erosion rate. Here we assume that the upper volume has greater resistance to erosion. Simply displacing the vertex violates our contextual constraint (Fig. 3b). To solve this problem we split the vertex in two, with locations P_t and P_{t+1}, and create new faces to connect them to their respective volume. Now each of these two vertices belongs to a single volume, and can be re-injected in the event queue to be displaced by the geomorphological simulation (Fig. 3c).

In this example we could also erode the upper volume if lower resistance to erosion was considered. If a vertex belongs to more than two volumes, we randomly choose two volumes and repeat the aforementioned operation until each split vertex belongs to a single volume.

3.3. Vertex sliding on a face

A vertex P with displacement P_tP_{t+1} slides on a face if and only if there exists one face F having P_t as a vertex and such that either P_{t+1} belongs to F, or P_tP_{t+1} collides with one of its edges, as shown in Fig. 4a. If P collides with an edge, we need a topological operation to insert a new vertex on this edge corresponding to the collision (Fig. 4). Then the remainder of the displacement is re-injected back in the event queue, as explained in section 3.1.

We also use the aforementioned perturbation scheme to

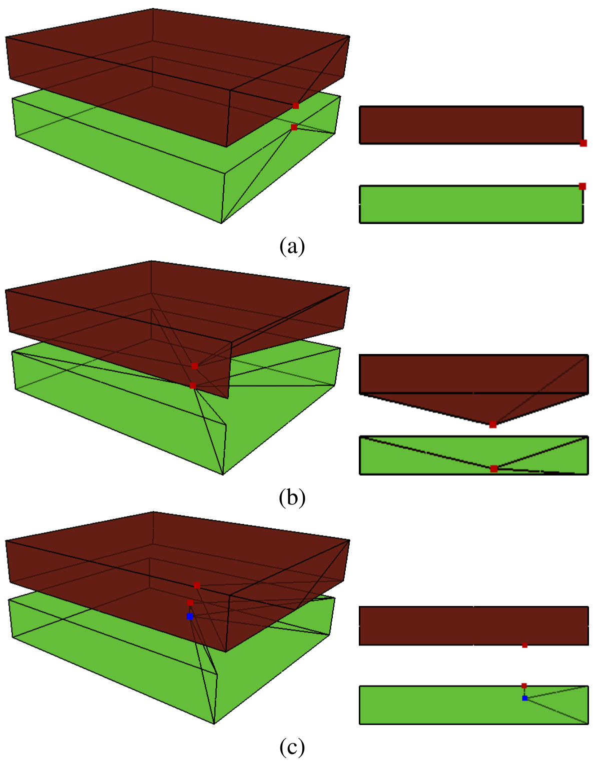

Figure 3: *(a) Exploded and sectional view of the initial configuration where two volumes share a common vertex marked in red. The upper volume has greater resistance to erosion. (b) Vertex displacement towards the center and slightly downwards leading to volume growth on the top. (c) Vertex split in two vertices (red and blue) allows to deform the lower volume only.*

avoid vertex-vertex collision, by slightly modifying the displacement and go back to the case depicted in Fig. 4.

3.4. Vertex-face collision

We consider potential collisions between the displacement of vertex P and all faces belonging to the same volume. Two cases can arise, depending if the collided face is connected by at least one edge to vertex P or not.

In the latter case, illustrated in Fig. 5, we must create a polygonal hole in the face; its vertices are the intersections of the edges adjacent to P with the collided face. After the hole is created, the extra volume added by displacing P is removed. Finally, in order to maintain coherent adjacency relationships in the model and avoid relying on other data structures such as inclusion trees, we connect an edge from the border of the face to the hole as shown in Fig. 6. Depending on the simulation, it is also possible that the face colliding with P belongs to another volume. The only difference is that we do not remove the extra volume added by displacing P (see Fig. 5d).

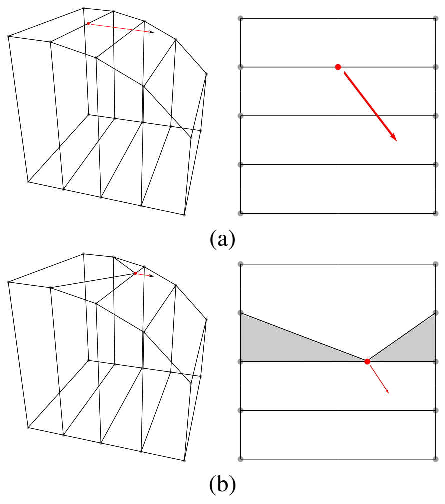

Figure 4: *(a) 3D and top view of a vertex sliding on a face and colliding with an edge. (b) A new vertex is created at intersection point to change the topology of the original faces before the simulation continues.*

The case of vertex P colliding with a face connected to P by at least one edge can be subdivided into 3 different cases:

1. if P is connected by one edge exactly, we simply contract this edge (Fig. 7a).
2. if P is connected by two edges as in Fig. 7b: we insert a new edge connecting P to the face, and process this case as case 3.
3. if P is connected by at least three edges, we contract all edges at once, except extremal ones (Fig. 7c).

This approach allows a generic process for any number of connecting edges, and helps maintain topological coherence as shown on the right side of Fig. 7.

Finally, any newly created vertex is re-injected back in the event queue and possibly continues its displacement.

3.5. Volume Pinching

When a volume is modified due to a vertex-face collision described in the previous section, it can be geometrically "pinched" as shown in Fig. 8b. But we also need to apply topological operations in order to split all pinched faces incident to vertex P. This process is very similar to the one described for sliding vertices (see section 3.3). To detect such cases we iterate over the neighbouring edges of P: if more than one cycle is found that goes back to P, then P is a "pinching point".

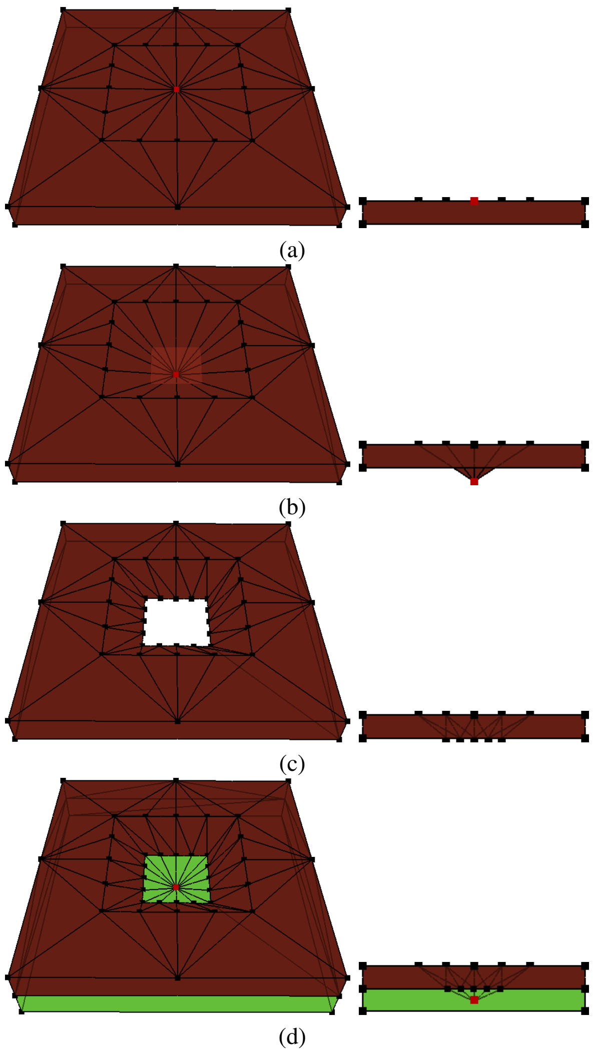

Figure 5: *3D and sectional view of a vertex "drilling" into a volume (a) Initial configuration where the vertex in red should be displaced vertically downwards. (b) Geometrical incoherence if the vertex crosses the lower face. (c) A hole is created and extra volume is removed. (d) If the face belongs to two volumes, a small dip is created inside the lower one.*

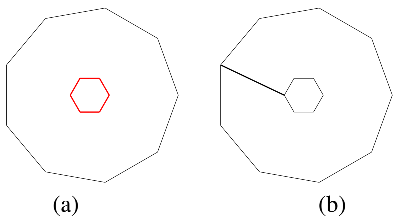

Figure 6: *(a) Polygonal face with a hole marked in red (b) Edge insertion between the hole and the border. Any edge that does not intersect existing edges can be chosen.*

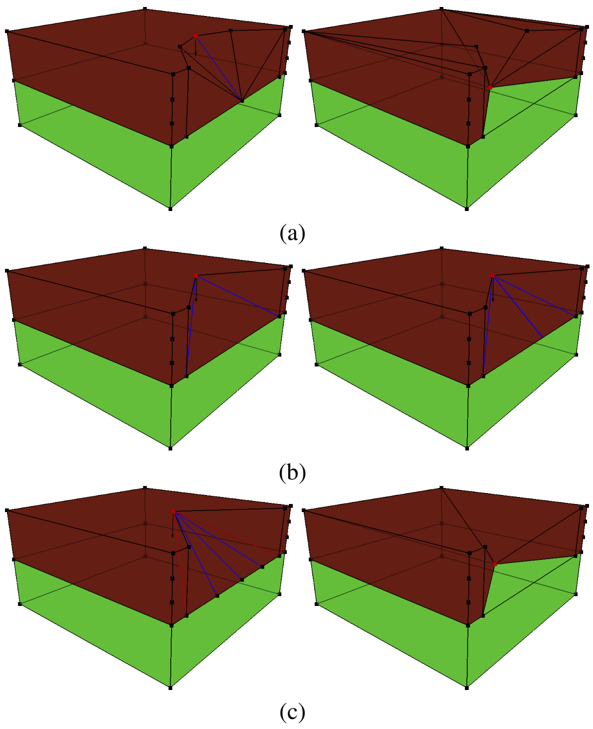

Figure 7: *The red vertex is displaced vertically downwards and collides with the face shared by the two volumes. (a) Vertex is connected to the colliding face by one edge marked in blue: this edge is simply contracted. (b) Connexion by two edges: a new edge is created. (c) Connexion by more than two edges: all edges are contracted except extremal edges marked in red.*

4. Implementation and results

4.1. Generalized maps and Moka

We use the "generalized map" topological model [Lie94] to subdivide geometric objects in cells (vertices, edges, faces, etc.) linked together by adjacency / incidence relationships. A n-dimensional generalized map (or n-G-map) is a set of abstract elements called darts, defined in an homogeneous way for any dimension n, associated with involutions respectively denoted as $\alpha_0, \alpha_1, \ldots, \alpha_n$. For convenience, we use 3-G-maps, since we want to represent vertices to volumes only. Any cell (vertex, edge, face, volume) is made of a collection of darts named *orbit* linked by some involutions α_i; as an example, the *vertex orbit* associated with a dart d is the set of all darts linked by any combination of $(\alpha_1, \alpha_2, \alpha_3)$ involutions, starting from d. Note also that adjacent i-dimensional cells (with $0 \leq i \leq n$) are connected by α_i involutions.

As an example we show how to implement the 2D face splitting operation described in section 3.3. Fig. 9 shows α_1 relationships in red and α_2 in green, vertices marked by gray dots and darts by dark segments. For the initial configuration

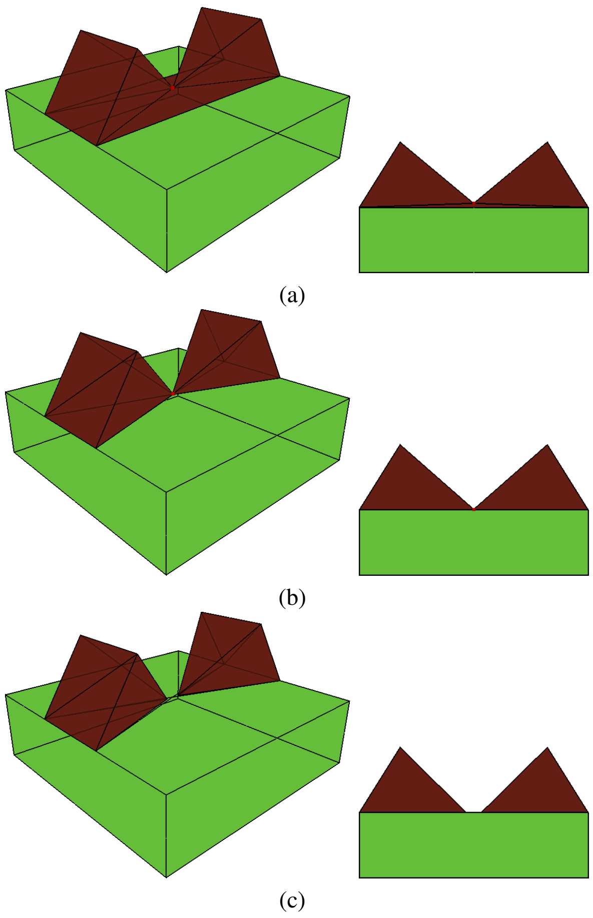

Figure 8: *3D and sectional view of pinching volume (a) Initial configuration where the red vertex is displaced vertically downwards. (b) Pinching detection. (c) Vertex is split to create two subvolumes. The video corresponding to this figure can be seen at:* `http://vimeo.com/user7861676`

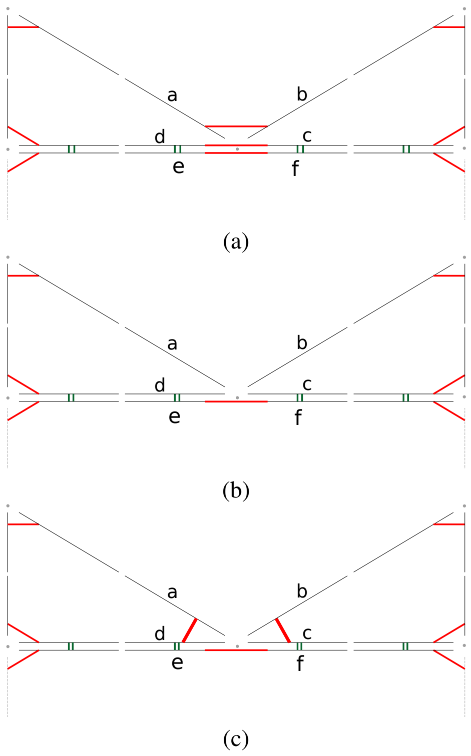

Figure 9: *Generalized map implementation of a face splitting operation.*

(Fig. 9a) we can see that $\alpha_1(a) = b$, $\alpha_1(b) = a$, $\alpha_1(c) = d$ and $\alpha1(d) = c$, meaning that we have two distinct vertex orbits (or vertices) that we want to merge. Edges corresponding to darts c and d are connected to the lower face using α_2. To split the upper face, we must first unlink α_1 relationships (Fig. 9b), then link again to obtain: $\alpha_1(a) = d$, $\alpha_1(b) = c$, $\alpha_1(c) = b$ and $\alpha_1(d) = a$. Now all darts belong to the same vertex orbit (Fig. 9c), which is verified for example as, starting from dart a, b can be reached following the path (α_1, α_2, α_1, α_2 and α_1). This operation as well as all other atomic operations described in the previous section are implemented in MOKA, a 3D geometrical modeler based on a generalized map kernel [VD], which contains different operations such as basic object creation, cell insertion, removal and contraction, triangulation. Cell insertion and triangulation operators were used for example to process non-planar faces (see section 3.1) and holes (section 3.4). Our operations are implemented as a separate library that interacts both with the kernel and the modeler libraries. Fig. 10a shows a snapshot of the application.

Storing 3-G-maps does not involve excessive memory usage, as each dart requires only 20 bytes. We must also store subvolumes labels and geometric locations of vertices, but for example the memory requirement for the arch model shown on the bottom right of Fig. 1 is less than 16 Kb.

4.2. Complex evolution scenarios

The user can interact with our application to generate an initial synthetic terrain and apply complex scenarios. These scenarios are implemented by a combination of atomic operations and attached to user-specified vertices. In the first scenario, a terrain defined as a stack of different materials is iteratively eroded by one vertex, as illustrated in Fig. 10. This example shows a combination of vertex-face collisions to drill into successive layers, and vertices sliding along faces to progressively enlarge the hole. The video corresponding to this figure can be seen at: `http://vimeo.com/user7861676`. In this simulation, each time the vertex displacement reaches an arbitrarily defined threshold between two frames, a new material appears as a complete sur-

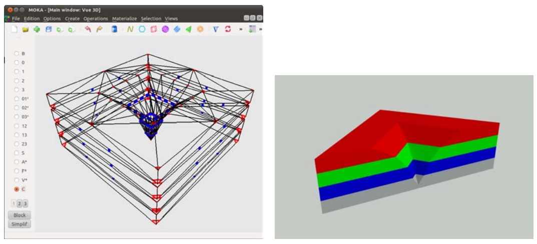
Figure 10: *A vertex drilling through several layers (a) Modelling with Moka. (b) Cross-sectional view.*

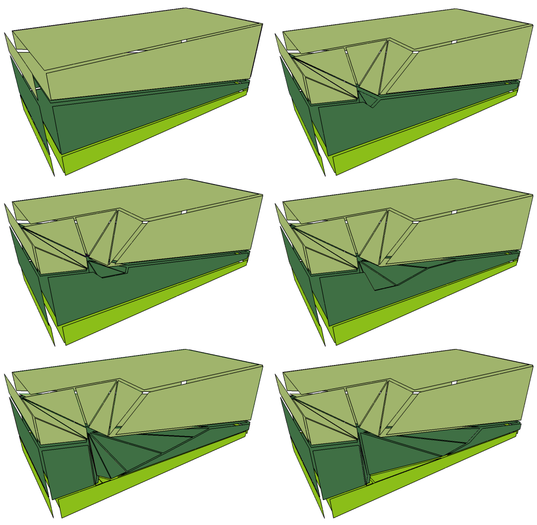
Figure 11: *Cross-sectional view of a cavern eroded inside a soft material volume. The top volume is eroded vertically until the drilling reaches the internal volume, then this volume is laterally excavated. This digging extends into the middle volume and also affects the bottom volume.*

face in one step. We have arbitrarily heightened this effect to make the visibility of topological changes easier. In a real simulation, vertex displacements would be driven by the parameters of a geological phenomenon and the smoothness of new material appearance would depend of these parameters.

The second example relies on a two volumes, one of them being progressively eroded at the base to form an arch (see Fig. 1). A vertex belonging to both volumes is first split at the base of the arch, and slides along the base until it reaches the other side. The blue volume is then pinched, and finally the arch is enlarged by successive vertex-face collisions.

The last example shows how differential erosion can be achieved with our approach, as illustrated in Fig. 11 with a cross-sectional view. The soft material layer in the middle has higher erosion rate than neighbouring volumes, which progressively creates a cavern.

These examples show the ability of our approach to maintain consistent topological relationships during a complex evolution scenario, avoiding unexpected self-intersections, holes, and so on. It relies on atomic operations executed in constant time, and does not need expensive tetrahedralizations as in [TJ10]. Even if our current models are both geometrically and topologically simple, our method scales with the boundary area of the 3D terrain rather than the volume as in [TJ10] since we need detailed representations only in areas where deformations can occur. Most parts of the 3D terrain can be represented roughly and stored in a compact way by our generalized map model, which is the main advantage over voxel grids or layered data structures.

It is also possible to refine geometry at boundaries if more precision is needed by the geomorphological simulation, but this refinement step should rely on topology-preserving smoothing and fairing filters. Geometric refinement can also be used to render visually realistic images. For example the arch obtained at the end of the simulation depicted on Fig. 1 was geometrically refined and rendered with Blender to produce photo-realistic images. The result, shown on Fig. 13, is close to real arches observed in nature (see Fig. 12). The refined model with approximately 55K faces and a fluid simulation was rendered in about 4 minutes on an Intel Core 2 Duo T9550 processor to produce a 1920x1080 picture.

5. Conclusion and future work

We have shown that the representation of a structured 3D terrain with a topological model can be used to simulate geomorphological evolution processes by defining a reduced set of atomic operations. These robust operations maintain consistent topological relationships thanks to their implementation within the generalized maps formalism, and can be used to simulate complex evolution scenarios. Our approach executes in constant time and is able to handle various topological features commonly found in nature, such as arches or caves, without the need for storing complex geometries.

We expect that a particle-based fluid simulation could be easily added to our modelling software, and used to implement more realistic erosion scenarios. Additional work is needed to define atomic operations corresponding to sedimentation in order to displace volumes around the 3D terrain, and other geomorphological processes could also be investigated (plate tectonics, glaciations, etc.).

Finally, we could also consider non-rectilinear vertex displacements. For example when a vertex slides on a face, this face could itself be deformed (due to tectonic activity), which leads to a non-linear vertex displacement. Since defining such displacements analytically would imply higher computation costs, we could investigate how geomorphology characterizes this type of trajectories and determine ad-hoc resolution systems.

References

[BF01] Beneš B., Forsbach R.: Layered data representation for visual simulation of terrain erosion. In *Spring conference on Computer Graphics* (2001). 3

[BFO*07] Beardall M., Farley M., Ouderkirk D., Reimschussel C., Smith J., Jones M., Egbert P.: Goblins by spheroidal weathering. In *Eurographics Workshop on Natural Phenomena* (2007), Ebert D. S., Mérillou S., (Eds.), Eurographics Association. 2, 3

[BPC*10] Bezin R., Peyrat A., Crespin B., Terraz O., Skapin X., Meseure P.: Interactive hydraulic erosion using cuda. In *Proceedings of the 2010 international conference on Computer vision and graphics: Part I* (Berlin, Heidelberg, 2010), ICCVG'10, Springer-Verlag, pp. 225–232. 2

[BSP*04] Brandel S., Schneider S., Perrin M., Guiard N., Rainaud J.-F., Lienhardt P., Bertrand Y.: Automatic building of structured geological models. In *Proceedings of the ninth ACM symposium on Solid modeling and applications* (Aire-la-Ville, Switzerland, Switzerland, 2004), SM '04, Eurographics Association, pp. 59–69. 3

[BTHB05] Beneš B., Těšínský V., Hornyš J., Bhatia S. K.: Hydraulic erosion. *Computer Animation and Virtual Worlds* (2005). 2

[CSC*10] Chen Z., Stuetzle C., Cutler B., Gross J., Franklin W. R., Zimmie T.: Quantitative analysis of simulated erosion for different soils. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems* (New York, NY, USA, 2010), GIS '10, ACM, pp. 482–485. 2

[DEJ*99] Dorsey J., Edelman A., Jensen H. W., Legakis J., Pedersen H. K.: Modeling and rendering of weathered stone. In *SIGGRAPH* (1999), pp. 225–234. 2

[Edm60] Edmonds J.: A combinatorial representation for polyhedral surfaces. *Notices Amer. Math. Soc. 7* (1960). 3

[IFMC03] Ito T., Fujimoto T., Muraoka K., Chiba N.: Modeling rocky scenery taking into account joints. In *Computer Graphics International* (2003), pp. 244–247. 2

[JCD10] Jund T., Cazier D., Dufourd J.-F.: Edge collision detection in complex deformable environments. In *VRIPHYS* (2010), pp. 69–78. 3

[KBKO09] Kristof P., Beneš B., Krivanek J., Ondrej S.: Hydraulic erosion using smoothed particle hydrodynamics. In *Eurographics* (2009), pp. 219–228. 2, 3

[Lie91] Lienhardt P.: Topological models for boundary representation: a comparison with n-dimensional generalized maps. *Computer-Aided Design 23*, 1 (1991), 59 – 82. 3

[Lie94] Lienhardt P.: N-dimensional generalized combinatorial maps and cellular quasi-manifolds. *Int. J. Comput. Geometry Appl. 4*, 3 (1994), 275–324. 3, 6

[LSM08] Léon P.-F., Skapin X., Meseure P.: A topology-based animation model for the description of 2d models with a dynamic structure. In *VRIPHYS* (2008), pp. 67–76. 2, 3

[MDH07] Mei X., Decaudin P., Hu B.-G.: Fast hydraulic erosion simulation and visualization on gpu. In *Pacific Graphics* (2007), pp. 47—-56. 3

[MMJM09] Michael D. J., McKay F., Joseph B., Matthew B.: Directable weathering of concave rock using curvature estimation. *IEEE Transactions on Visualization and Computer Graphics* (2009), 81–94. 2

[Nag98] Nagashima K.: Computer generation of eroded valley and mountain terrains. *The Visual Computer 13* (1998), 456–464. 10.1007/s003710050117. 2

Figure 12: *A natural arch by the sea in Ibiza, and the Delicate Arch located in Arches National Park, Utah.*

[NWD05] Neidhold B., Wacker M., Deussen O.: Interactive physically based fluid and erosion simulation. In *Eurographics Workshop on Natural Phenomena* (2005). 3

[OF99] Ozawa N., Fujishiro I.: A morphological approach to volume synthesis of weathered stone. In *Volume Graphics* (1999), pp. 367–378. 2

[PB07] Pons J.-P., Boissonnat J.-D.: Delaunay Deformable Models: Topology-Adaptive Meshes Based on the Restricted Delaunay Triangulation. In *Proceedings of the 20th IEEE Conference on Computer Vision and Pattern Recognition* (Minneapolis, France, 2007), IEEE, (Ed.), p. 200. 3

[PGMG09] Peytavie A., Galin E., Merillou S., Grosjean J.: Arches: a Framework for Modeling Complex Terrains. *Computer Graphics Forum (Proceedings of Eurographics) 28*, 2 (2009), 457–467. 3

[TGM*08] Terraz O., Guimberteau G., Merillou S., Plemenos D., Ghazanfarpour D.: 3gmap l-systems: an application to the modelling of wood. *The Visual Computer 25*, 2 (2008), 165–180. 3

[TJ10] Tychonievich L. A., Jones M. D.: Delaunay deformable mesh for the weathering and erosion of 3d terrain. *Vis. Comput. 26* (December 2010), 1485–1495. 2, 3, 8

[vBBK08] Šťava O., Beneš B., Brisbin M., Křivánek J.: Interactive terrain modeling using hydraulic erosion. In *Symposium on Computer Animation* (2008), pp. 201–210. 3

[VD] Vidil F., Damiand G.: Moka - a topology-based 3d geometric modeler (http://moka-modeller.sourceforge.net/). 7

[WCMT07] Wojtan C., Carlson M., Mucha P. J., Turk G.: Animating corrosion and erosion. In *NPH* (2007), Ebert D. S., Mérillou S., (Eds.), Eurographics Association, pp. 15–22. 2

[ZLL07] Zhong D., Li M., Liu J.: 3d integrated modeling approach to geo-engineering objects of hydraulic and hydroelectric projects. *Science in China Series E: Technological Sciences 50* (2007), 329–342. 10.1007/s11431-007-0042-0. 3

[ZSTR07] Zhou H., Sun J., Turk G., Rehg J. M.: Terrain synthesis from digital elevation models. *IEEE Transactions on Visualization and Computer Graphics 13* (July 2007), 834–848. 2

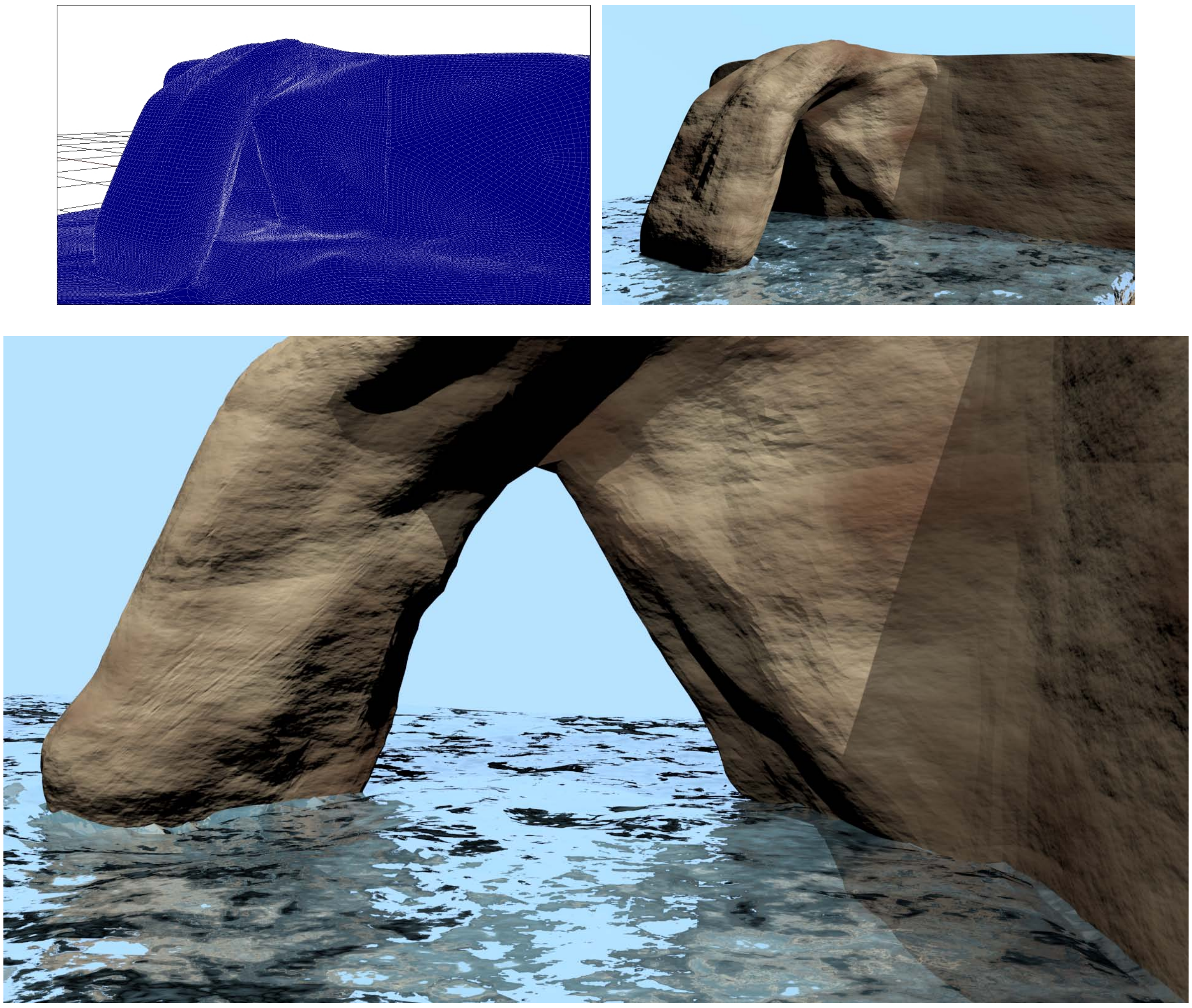

Figure 13: *A visually realistic image of the arch obtained with the simulation shown on Fig. 1. After a geometric refinement step combined with Perlin noise (top left), the scene was rendered with Blender using rock and ocean textures.*