UNIVERSITÉ DE LIMOGES
ÉCOLE DOCTORALE « Sciences et Ingénierie pour l'Information, Mathématiques »
FACULTÉ DES SCIENCES ET TECHNIQUES
Laboratoire XLIM(DMI-SIC)

Thèse N°

THÈSE
pour obtenir le grade de
DOCTEUR DE L'UNIVERSITÉ DE LIMOGES
Discipline / Spécialité : Informatique et Applications
présentée et soutenue par
Richard Bézin
le Jeudi 24 octobre 2013

Simulation d'évolution topologique : cas de l'interaction fluide / solide

Thèse dirigée par Philippe Meseure, Benoît Crespin,
Co-encadrée par Xavier Skapin, Olivier Terraz

JURY

Rapporteurs :	
M. Yannick REMION	Professeur à l'Université de Reims
M. David CAZIER	Maître de conférences HDR à l'Université de Strasbourg
Examinateurs :	
M. Pascal LIENHARDT	Professeur à l'Université de Poitiers
M. Philippe MESEURE	Professeur à l'Université de Poitiers
M. Guillaume DAMIAND	Chargé de recherche CNRS au LIRIS à Lyon
M. Benoît CRESPIN	Maître de conférences à l'Université de Limoges

Une personne qui n'a jamais commis d'erreur n'a jamais tenté d'innover.
Albert Einstein

Résumé

En synthèse d'image, le réalisme d'une scène dépend grandement du décor dans lequel sont intégrés les objets à représenter. Dans le cas d'un paysage naturel, celui-ci résulte d'une succession d'événements géomorphologiques qu'il est nécessaire de reproduire afin d'obtenir un résultat réaliste visuellement et cohérent géologiquement.

Nous nous sommes intéressés à l'évolution d'un terrain au cours du temps. Pour cela, nous devons tenir compte tant de son relief que des différentes couches géologiques qui le composent. En outre, nous devons déterminer quelles modifications opérer afin de simuler son évolution.

Dans un premier temps, nous nous intéressons à l'interaction entre un fluide et un maillage afin de simuler les phénomènes de sédimentation et d'érosion hydraulique d'un terrain par une rivière. Le modèle d'interaction proposé prend en compte les caractéristiques géologiques du terrain. Il permet d'éroder, de transporter et de déposer des sédiments en fonction de la vélocité du fluide, du type et de la taille des sédiments.

Nous présentons dans un second temps une nouvelle approche pour simuler les évolutions géomorphologiques d'un terrain en 3D conçu comme un ensemble de volumes définis dans un modèle topologique. Nous décrivons des opérations topologiques permettant de gérer les événements topologiques de manière robuste et cohérente. Ces opérations sont combinées pour produire des scénarios d'évolutions de terrain.

Mots clefs : Informatique graphique, phénomènes géologiques, modèle d'érosion et de sédimentation, simulation de fluide, animation géométrique à base topologique.

Abstract

In computer graphics, the realism of scenes greatly depends on the scenery in which represented objects are integrated. In the case of natural landscapes, this results in a succession of geomorphological events that it is necessary to reproduce to obtain a visually realistic and geologically coherent result.

The main focus of this thesis is the evolution of a terrain across time. We have to take into account the relief as well as the different geological layers composing the terrain. Furthermore, we have to determine which modifications are necessary to simulate its evolution.

First, we are interested in the interaction between a fluid and a mesh to simulate the hydraulic erosion and sedimentation phenomena by a river on a terrain. The proposed interaction model takes into account the geological properties of the terrain. It allows to erode, transport and deposit sediments according to fluid velocity, type and size of sediments.

Our second approach is intended to simulate the geomorphological evolution of a 3D terrain represented by a set of volumes and defined in the formalism of a topological model. We describe topological operations to manage topological events robustly and coherently. These operations are combined to produce scenarios of terrain evolutions.

Key words : Computer graphics, geological phenomena, erosion and sedimentation model, fluid simulation, topology-based geometric animation

Table des matières

Table des figures

Introduction

Les images de synthèses sont devenues en quelques décennies indispensables dans de nombreux domaines tels que le cinéma, la publicité, les jeux vidéos mais également dans la visualisation scientifique. Les maquettes utilisées jusque dans les premières années de l'audiovisuel ont été remplacées par des environnements 3D qui doivent se rapprocher le plus possible de la réalité pour que le spectateur ou l'utilisateur ne fasse plus la différence avec des objets réels. De grands progrès ont été effectués les dix dernières années grâce à l'évolution des ordinateurs. En effet un meilleur réalisme nécessite souvent rapidité et puissance de calcul, mais également capacités de stockage accrues. Néanmoins, la perception du réalisme est subjective : une image qui nous semblait réaliste hier l'est moins aujourd'hui. La synthèse d'images est donc un domaine en perpétuelle évolution.

Parmi les éléments à représenter, le décor est un élément clé, puisqu'il offre le cadre dans lequel se positionne l'ensemble des objets à représenter. La construction d'un décor réaliste tel qu'un paysage en images de synthèse est complexe, et fait donc en général l'objet d'un découpage en trois parties : la génération du terrain à l'échelle macroscopique, la création d'éléments plus petits tels que des arbres ou des bâtiments (échelle intermédiaire) et l'agencement de ces éléments sur le terrain. Les formes du relief d'un terrain sont déterminantes pour le placement ces éléments intermédiaires, il est donc essentiel de pouvoir représenter fidèlement ces formes.

Il existe des modèles scientifiques qui expliquent la formation et l'évolution des objets considérés sur lesquels nous pouvons nous baser pour augmenter le réalisme des scènes. Un paysage est un système décrit et analysé par la géographie et la géologie. En relation avec ces deux domaines, la géomorphologie est la science qui étudie les formes du relief terrestre et a pour objectif d'expliquer leurs origines. Ces formes sont modelées par des facteurs naturels abiotiques (physique, chimique) et biotiques (biologiques), mais aussi par des facteurs anthropiques (actions de l'Homme) (voir Figure 1). On peut donc considérer un paysage comme le produit de l'action du milieu abiotique, du monde vivant (animaux et végétaux) et des sociétés humaines. Ces différents processus interviennent à des échelles spatiales et temporelles variées. L'influence de la structure du système sur le relief, comme par exemple la tectonique des plaques et le volcanisme, est étudiée par la géomorphologie structurale. La géomorphologie dynamique étudie les processus externes au système qui contribuent à la formation et l'évolution des reliefs. L'érosion, le transport, le dépôt de matière sont des exemples de processus externes qui influent sur les formes particulières d'un paysage. La géomorphologie dynamique met également en relation l'aspect des formes et le climat actuel ou passé de la région du terrain observé.

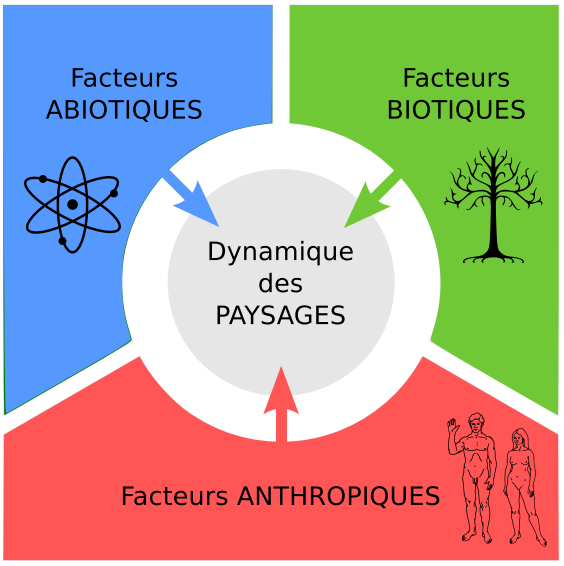


FIGURE 1 – Catégories de facteurs entrant en jeu dans la dynamique d'évolution des paysages. (schéma inspiré de http://osur.univ-rennes1.fr/)

L'évolution du terrain au cours du temps inclut la formation de modèles complexes tels que des grottes, des « cheminées de fée »[1], etc. Prenons l'exemple des falaises d'Étretat en Normandie (voir Figure 2). Nous pouvons observer plusieurs étapes de la création et de l'évolution d'une arche : les trois « portes » se succèdent sur le littoral à quelques centaines de mètres les unes des autres. La falaise d'Amont (Figure 2a) est une paroi fine qui a fini par être percée à sa base, ce qui a donné naissance à la plus petite des trois portes : la porte d'Amont. La Manneporte et la porte d'Aval sont plus larges et plus hautes. L'aiguille d'Aval accompagne l'arche, elle laisse penser à une évolution possible d'une arche ou d'une falaise qui s'effondre. Enfin, l'aiguille de Belval (sur la commune voisine d'Étretat) est isolée à une centaine de mètres de la falaise, ce qui montre que cette dernière a reculé sous l'effet de l'érosion. Ces formes particulières sont composées des mêmes matériaux et issues de la même falaise. Elles nous montrent qu'il est possible d'observer dans la nature des formes à différents stades de leur évolution. Il apparaît donc essentiel de pouvoir simuler leur évolution au cours du temps pour pouvoir représenter leur diversité.

Nous considérons dans la suite de ce document que le terme « terrain » désigne à la fois la surface visible d'un relief mais également sa structure et sa forme. Ainsi, ce terme englobera grottes, arches ou autres formes qui peuvent constituer son sous-sol. Puisque nous ne percevons que la partie visible du terrain, il semble logique de représenter uniquement celle-ci. Nous verrons dans l'état de l'art que des travaux ont opté pour cette solution en représentant seulement la surface du terrain. Ces méthodes sont peu coûteuses en mémoire mais ne sont pas suffisantes pour modéliser des détails fins du terrain ou bien des formes complexes, voire même simplement concaves. D'autres méthodes utilisent une approche

1. Une cheminée de fée est une sorte de grande colonne naturelle faite de roches friables surplombée par une roche plus résistante.

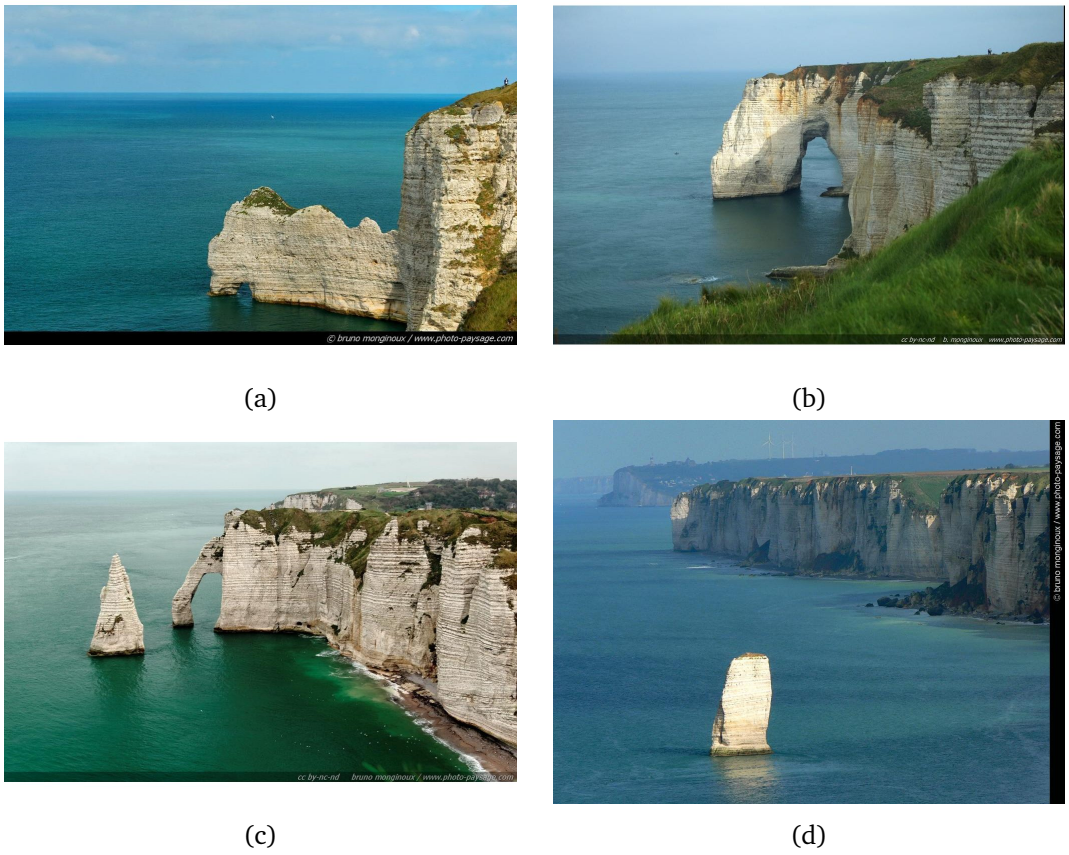


FIGURE 2 – Photographies des falaises d'Étretat. (a) la falaise d'Amont, (b) le Manneporte, (c) la porte d'Aval et (d) l'Aiguille de Belval. (Photo partagée par B. Monginoux - www.Photo-Paysage.com (Creative Commons BY-NC-ND))

physique pour simuler l'altération du terrain et ainsi générer les détails recherchés. Ces approches se basent par exemple sur l'écoulement d'un fluide sur le terrain pour déterminer les zones érodées et sédimentées. Mais elles ne simulent en général que les interactions chimiques entre le fluide et différents matériaux. En outre, les méthodes représentant uniquement la surface aussi bien que les méthodes représentant la structure et cherchant à simuler l'évolution du terrain se sont heurtées au même problème : comment conserver la cohérence géométrique, topologique et contextuelle au cours de la simulation ? Certaines méthodes utilisent un modèle topologique permettant de prendre en compte la structure en couches du terrain, mais elles ne permettent pas en général de simuler son évolution au cours du temps.

Les travaux présentés dans ce mémoire associent la simulation de fluide et la modélisation basée sur la topologie afin de simuler l'évolution d'un terrain au cours du temps. Nous proposons en premier lieu une méthode d'interaction en temps interactif entre un fluide et un terrain. Notre approche permet de simuler l'action d'un fluide entraînant l'érosion, le transport et le dépôt de matière de manière cohérente sur un terrain. Les interactions sont à la fois de type chimique et mécanique. De plus, le réglage des paramètres de la simulation est facilité car leurs unités sont exprimées dans le système international et ils

sont basés sur les observations géologiques. D'autre part, nous avons développé une approche basée sur un modèle d'évolution topologique 3D pour la géomorphologie. Nous avons utilisé un modèle topologique afin de modéliser la structure en couches du terrain et de garantir sa cohérence, puis un modèle d'animation à partir du modèle topologique pour représenter l'évolution du terrain au cours du temps. Pour conserver la cohérence géométrique, topologique et contextuelle du modèle, nous avons créé un ensemble d'opérations topologiques. Ces opérations peuvent être combinées entre elles ou avec d'autres opérations classiques de modélisation pour obtenir les formes complexes présentes dans un paysage.

Organisation du mémoire

Au cours du premier chapitre de ce mémoire, nous étudions les travaux en relation avec nos recherches. Nous consacrons une partie de ce chapitre à la génération de terrain. Nous discutons des différentes méthodes de représentation du terrain : fractales, voxels et représentation par les bords, puis nous les opposons aux approches inspirées de la physique qui permettent d'altérer un terrain. Une autre partie de ce chapitre est consacrée au modèle des cartes généralisées, il s'agit d'un pré-requis permettant une meilleure compréhension de nos travaux. Nous avons choisi d'utiliser ce modèle pour décrire notre terrain car il permet de représenter à la fois la géométrie des objets manipulés et les notions d'adjacence entre les différents éléments, ce qui facilite la création d'opérations complexes.

Le second chapitre est dédié à notre modèle interactif d'érosion et de sédimentation. Nous avons développé une approche mécanique basée sur un modèle de type « Discrete Element Method » (DEM) pour simuler le fluide. Le terrain est ici uniquement surfacique et représenté par un maillage triangulaire. Nous avons également développé une approche basée sur les modèles habituellement utilisés en géomorphologie, avec un modèle de fluide plus complexe appelé « Smoothed Particules Hydrodynamics » (SPH) permettant de discrétiser les équations de Navier-Stokes. La surface du terrain est également représentée par des particules, ce qui permet de conserver une structure unifiée entre le fluide et le terrain. Nous justifierons les choix effectués en discutant ces deux méthodes.

Dans le troisième chapitre, nous détaillons notre modèle d'évolution topologique 3D consacré à la géomorphologie. Nous représentons chaque couche du terrain par un volume qui est topologiquement lié aux couches adjacentes. Pour faire évoluer le terrain, nous déplaçons les sommets qui composent ces couches et détectons les collisions qui peuvent survenir. Nous détaillons les opérations permettant de traiter ces collisions. Nous décrivons ensuite l'ensemble des résultats que notre modèle permet d'obtenir : la formation d'arches, de grottes, et d'autres formes naturelles.

Enfin, nous concluons sur l'ensemble de nos travaux en analysant ses apports et ses limites ainsi que les perspectives possibles.

Chapitre 1

État de l’art

Sommaire

Chapitre 1

État de l'art

La création de paysage est un problème complexe qu'il est nécessaire de découper en plusieurs étapes. Reeves et Blau [RB85] proposent un processus de génération partant des plus gros éléments aux plus petits pour éviter les conflits à la création :

- Modélisation du terrain
- Création des éléments plus petits comme les arbres, les rochers, les bâtiments, l'herbe, *etc*.
- Positionnement de ces éléments sur le terrain.

Dans cet état de l'art, nous nous intéressons à la génération de terrain.
Plusieurs méthodes, dites procédurales, abordent cette problématique et permettent de construire un terrain à partir de sa surface ou bien de la composition des couches sous-jacentes au terrain. Ces approches se basent sur des formules mathématiques. Elles représentent le terrain de manière soit continue à diverses échelles, soit discrétisée. Nonobstant, le terrain ainsi généré est souvent figé dans le temps : ces méthodes ne prennent pas en compte les altérations du terrain au cours du temps. Ces premières approches sont présentées dans la section 1.1.
Il est également possible de simuler l'évolution d'un terrain au cours du temps à partir de phénomènes naturels comme l'érosion hydraulique. Ces méthodes, inspirées de la physique et recourant notamment à des simulations de fluide, sont décrites dans la section 1.2. Elles ne permettent généralement pas d'obtenir des détails topologiques complexes du terrain comme des arches, des tunnels, etc.
Enfin, d'autres travaux représentent la structure géologique à l'aide de volumes géométriques. Ces méthodes s'appuient sur un modèle topologique qui conserve la notion d'adjacence des couches, ce qui permet de modeler le terrain à l'aide d'opérations en prenant en compte les différentes propriétés des couches. Nous détaillons le modèle topologique utilisé et les méthodes qui en découlent dans la section 1.3. Cette dernière approche permet de conserver la continuité du terrain grâce à la topologie tout en permettant de le modifier en garantissant sa cohérence topologique.

1.1 Méthodes de construction de terrains

Les méthodes de constructions de terrain s'inspirent des observations faites dans la nature : la forme et l'aspect des objets, leur redondance dans un milieu, etc. Dans le domaine de la représentation de terrain, deux méthodes procédurales se distinguent : la première utilise des fractales ou des cartes de hauteur afin de générer les détails surfaciques du terrain, la seconde recourt à des structures de données (voxels, maillages) afin de représenter non seulement la surface du terrain, mais aussi sa structure en couches constituées de différents matériaux. Ces deux approches peuvent par ailleurs être utilisées conjointement pour recréer un terrain volumique avec des détails surfaciques.

1.1.1 Fractales et cartes de hauteur

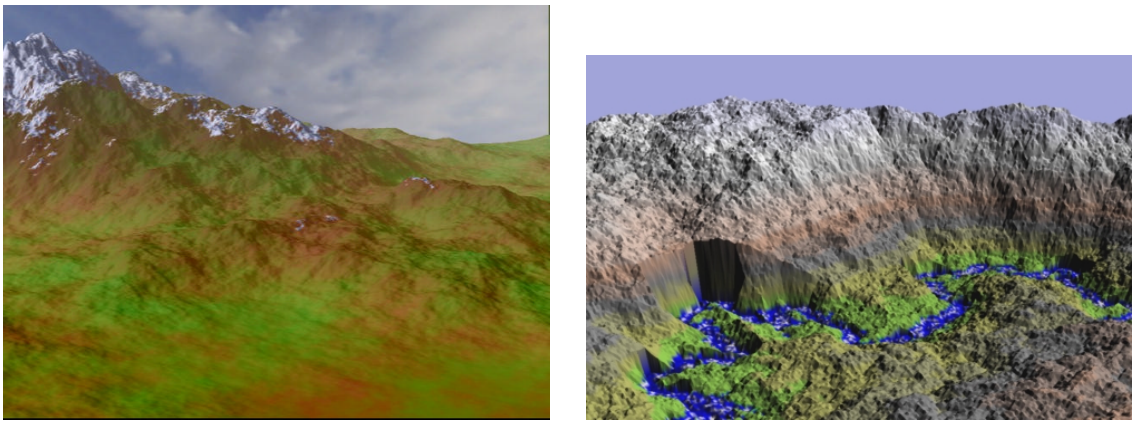
Les premières méthodes de génération de terrain représentent uniquement sa surface à partir de fonctions stochastiques, comme Fournier *et al.* [FFC82] qui génèrent des détails de terrain. [Man83] et [Lew87] utilisent des fractales pour générer automatiquement des terrains à grande échelle (Figure 1.1a). Une fractale est une courbe ou une surface de forme irrégulière créée à partir de règles déterministes ou stochastiques impliquant une homothétie interne. Mandelbrot présente ce nouvel ensemble en 1974, le nom est un néologisme venant du latin "fractus" qui signifie brisé, irrégulier. Mais ces méthodes ne permettent pas de représenter tous les constituants d'un paysage, les rivières par exemple.

Kelley *et al.* [KMN88] génèrent un terrain fractal autour de bassins versants prédéfinis. Belhadj *et al.* [Bel07] utilisent un modèle de fractales contraintes pour créer des détails locaux comme un cours d'eau. Génevaux *et al.* [GGG+12] utilisent un réseau hydrographique généré à l'aide de grammaires pour construire le relief du terrain en respectant les contraintes imposées par l'écoulement d'eau : l'eau suit la pente. Contrairement à ces méthodes qui nécessitent plusieurs passes, Prusinkiewicz *et al.* [PH93] créent directement les rivières pendant le processus de génération du terrain fractal (Figure 1.1b). Néanmoins, ces méthodes ne répondent pas à des critères de réalité physique, le terrain est obtenu après plusieurs itérations de la méthode qui permet d'augmenter le niveau de détails.

Musgrave *et al.* [MKM89] créent des lits de rivières en simulant l'érosion fluviale sur des montagnes fractales. Le modèle d'érosion utilisé est décrit dans la partie 1.2.2.

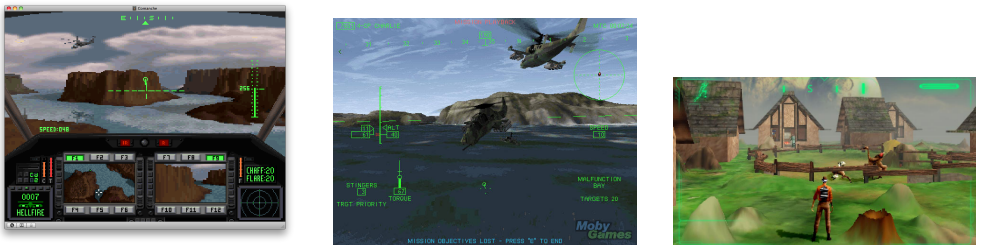
1.1.2 Méthodes à base de voxels et de strates

Les voxels sont exploités pour construire et visualiser un terrain dans les jeux vidéos dès les années 1990. *Comanche* de Novalogic en 1992 (Figure 1.2a) et plus tard *Comanche 3* en 1997 utilisent l'accélération 3D (Figure 1.2b) pour accélérer la visualisation à l'écran. *Outcast* de Appeal Software utilise également les voxels en 1999 pour représenter un terrain détaillé pour l'époque (Figure 1.2c). La résolution affichée du jeu est néanmoins limitée à 512x384.
Voxel est un terme provenant de la contraction de "volumetric picture element", il s'agit d'un échantillon de volume représentant une valeur dans une grille d'un espace en 3D. Un voxel basique n'a pas accès explicitement à ses coordonnées spatiales, celles-ci sont



(a) (b)

FIGURE 1.1 – (a) Terrain fractal obtenu par Lewis [Lew87], (b) Terrain fractal avec une rivière obtenu par Prusinkiewicz *et al.* [PH93].



(a) (b) (c)

FIGURE 1.2 – Différents jeux utilisant les voxels pour représenter le terrain : (a) Comanche en 1992, (b) Comanche 3 en 1997 et (c) Outcast en 1999.

déterminées selon sa position relative par rapport aux autres voxels dans la grille. Un voxel peut représenter diverses propriétés comme la couleur, l'opacité, etc. Dans le cas d'une représentation géologique du terrain, chaque voxel stocke la quantité et le type de matière.

Les méthodes de construction décrites ci-dessous permettent de construire des détails ou des modifications locales du terrain en se basant sur les voxels. Contrairement aux cartes de hauteur qui ne stockent que des données surfaciques, les voxels peuvent représenter des objets en 3D. Il est ainsi possible de représenter des concavités telles que des grottes, des arches ou des surplombs.

La représentation par voxel est coûteuse en mémoire. En effet, pour représenter un terrain de $1024 \times 1024 \times 1024$ éléments avec n octets de données pour chaque élément, la méthode basée sur les voxels nécessite $n\,1024^3$ octets, soit n Go. Or, pour représenter un terrain, la grille de voxels doit être suffisamment grande pour représenter à la fois l'aspect général du terrain (crêtes, vallées, etc) mais aussi les détails.
Pour remédier à ce problème, Benes *et al.* [BF01] introduisent une nouvelle structure s'inspirant des voxels : une structure en piles ou en couches. Ils utilisent une grille régulière 2D, chaque case de la grille représentant une pile de matériaux. Chaque élément de la pile définit l'épaisseur de la couche et le type de matériau (Figure 1.3). Cette méthode

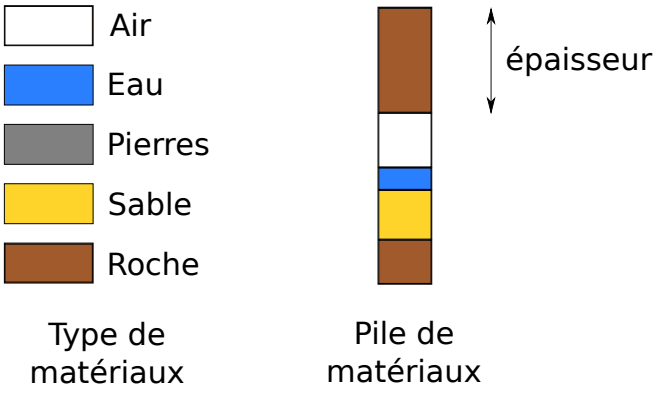


FIGURE 1.3 – Modèle en couches de Benes *et al.* ([BF01]).

est inspirée des carottages effectués par les géologues afin de connaître la nature du sol. Elle reprend les avantages des voxels puisqu'elle permet de représenter des couches de différentes matières, d'eau, d'air ou de gaz, ainsi que des concavités, tout en étant moins gourmande en mémoire. En effet, il suffit de $n\,1024 \times 1024$ octets, soit n Mo, avec la méthode par couches. De plus, la complexité de l'algorithme de parcours des voxels est en $O(n^3)$ alors que celle de la méthode des couches est en $O(k \cdot n^2)$ où k est le nombre de couches. La hauteur d'une pile est égale à la somme des hauteurs de toutes les couches qui la composent. Pour effectuer le rendu des cavernes et autres structures du sous-sol, Benes *et al.* utilisent la technique des *Marching Cubes* [LC87]. Bien que cette méthode permette de représenter à la fois un terrain et ses détails, la structure utilisée ne permet pas de le modifier de manière continue.

Peytavie *et al.* [PGMG09] ajoutent à cette structure de données discrètes une représentation implicite pour sculpter et générer la surface du terrain. La représentation implicite est une surface de convolution, permettant de passer à une surface continue des différentes couches de matières que l'on peut ainsi trianguler pour effectuer le rendu. En modifiant la surface, on modifie les couches de matériau. Lors de l'érosion, une partie de la roche est transformée en sable et en pierres. Il est parfois nécessaire de les redistribuer parmi les couches voisines moins élevées afin de stabiliser les couches. Une seule texture est créée pour la roche et les sédiments, les pierres étant traitées séparément. Le maillage est texturé en mélangeant différentes textures de matériaux en fonction du ratio des matériaux de surface du sol (Figure 1.4). Cependant, cette méthode ne permet pas de modéliser des rivières contenant des chutes d'eau par exemple.

Ito *et al.* [IFMC03] simulent les fissures de la roche sur un terrain macroscopique. Cette méthode utilise une grille de voxels et permet de créer des falaises fissurées (Figure 1.5b). Comme nous allons le voir, la scène est limitée par la taille et la résolution de la grille. Un voxel possède l'information d'adjacence des six voxels voisins, l'index du groupe de voxels joints, formant un bloc (Figure 1.5a). Une fissure est créée en supprimant des relations entre voxels. Un angle de friction est utilisé pour déterminer les glissements de blocs. L'algorithme d'érosion cherche les blocs de surface, un certain nombre sont choisis au hasard et pour chacun de ces blocs, les traitements suivants sont répétés jusqu'à ce que le nombre de blocs supprimés soit atteint : on estime la direction de chute ou de glissement

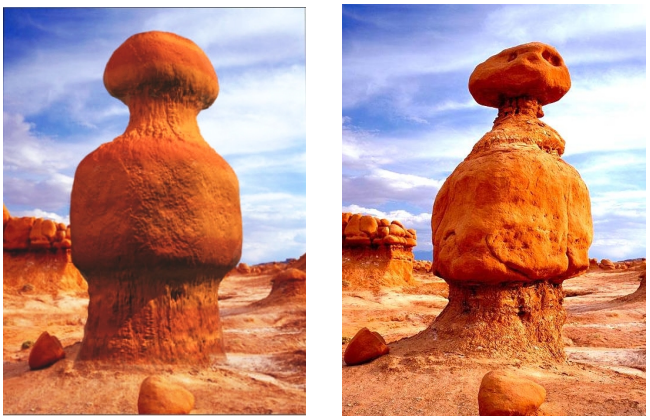
FIGURE 1.4 – Résultats obtenus par Peytavie *et al.* ([PGMG09]).

(a) (b)

FIGURE 1.5 – (a) blocs de voxels. (b) Exemple de résultats obtenus.

du bloc, puis on calcule les forces gravitationnelles externes dues à l'angle de friction sur le bloc pour déterminer s'il est toujours en équilibre. Si l'équilibre n'est pas maintenu, le bloc est déplacé. Les blocs qui sortent de la grille de voxels sont supprimés. Une fois le nombre de blocs supprimés atteint, un traitement pour faire tomber les blocs en suspension est effectué.

Dans [BFO+07], Beardall *et al.* conçoivent des cheminées de fées (goblins en anglais) à partir d'un cylindre, créé avec une image en niveau de gris, qu'ils discrétisent en voxels puis érodent à l'aide de volumes sphéroïdaux. Cette méthode nécessite l'intervention de l'utilisateur qui doit définir les zones à éroder, elle n'est donc pas adaptée à un modèle grande échelle tel qu'un terrain. En effet, l'utilisateur doit définir plusieurs niveaux d'érosion et de sédimentation pour pouvoir à la fois appliquer une modification globale du terrain, par exemple l'aplanissement des crêtes, et également les détails plus fins comme les corniches. Ce travail est long et relativement fastidieux. Afin de créer la forme particulière des goblins, les voxels du cylindre sont découpés en différentes couches qui possèdent chacune leur propre résistance et volume d'érosion. L'érosion se fait en prenant en compte la résistance à l'érosion du voxel et en calculant le ratio air / matière de chaque voxel contenu dans la forme sphéroïdale : les voxels des coins sont érodés plus rapidement que les voxels des faces. Quand la quantité de matière du voxel atteint zéro, il est supprimé. Beardall *et al.* [BFO+07] utilisent la méthode des Marching Cubes pour extraire la surface. La figure 1.6 montre un goblin synthétisé placé sur une photographie de la Goblin Valley



(a) (b)

FIGURE 1.6 – (a) Un goblin synthétisé placé sur une photographie de la Goblin Valley State Park en Utah. (b) La photographie originale (copyright Leping Zha, www.lepingzha.com)

State Park en Utah (Figure 1.6a) et la photographie originale (Figure 1.6b).

Dorsey *et al.* [DEJ+99] utilisent les voxels pour simuler l'érosion ou la dissolution chimique de la pierre. À partir d'un maillage polyédrique et d'une carte de dépôts et d'écoulement d'eau, le maillage est discrétisé en voxels. Chaque voxel possède une information d'impureté, d'humidité et de densité : un voxel complètement à l'intérieur du maillage possède une densité de 1, un voxel complètement à l'extérieur une densité de 0. À l'aide de cette densité, Dorsey *et al.* génèrent une pierre non altérée. Ensuite, pour recréer la composition initiale de la pierre, ils appliquent différentes textures procédurales comme un bruit de Perlin pour du marbre [Per85]. L'érosion se fait en plusieurs phases : une première, humide, où l'eau dissout les minéraux, et une seconde, sèche, où l'eau s'évapore et dépose les minéraux. Plus un voxel contient de l'eau longtemps, plus sa densité diminue. Une fois la densité à 0, la pierre n'est plus présente dans ce voxel. Dorsey *et al.* utilisent un rendu sous-surfacique prenant en compte les matériaux qui composent la pierre (Figure 1.7). Pour que les modifications apportées au maillage de base soient cohérentes, la grille de voxel doit être suffisamment fine : cela implique un coût important en mémoire.

1.1.3 Méthodes à base de représentation par les bords

Via la représentation par les bords, il est possible de modéliser la surface et les couches sous-jacentes du terrain à l'aide d'un seul modèle. Les données relatives aux roches qui composent le terrain sont soit liées aux sommets, soit aux faces, soit aux volumes.

Un maillage triangulaire est utilisé dans [KBKv09] afin de modéliser la surface du terrain, les sommets sont déplacés seulement verticalement, selon que le terrain est érodé ou sédimenté. Par construction du maillage, les triangles ont tous la même aire projetée, le changement de volume est toujours le même tant que la somme des déplacements des sommets du triangle est la même. Dans le cas de la sédimentation, le sommet le plus bas du triangle dans le sens vertical est privilégié, alors qu'il s'agit du sommet le plus haut dans le cas de l'érosion afin d'aplanir le terrain. Une simulation de fluide est utilisée pour

(a) (b)

FIGURE 1.7 – Statue d'Artémis (a) non altérée et (b) érodée.

déterminer les zones sédimentées ou érodées. Nous détaillerons cette partie dans la section 1.2.

Les n-G-cartes, dont nous parlerons plus en détail dans la partie 1.3, permettent de modéliser des objets de dimension n. Dans [PTM+11], un modèle de 2-G-cartes est utilisé pour modéliser la surface du terrain et générer des empreintes sur ce modèle (Figure 1.8a). Les sommets portent l'information des matériaux situés en dessous du terrain à l'aide d'une pile de couches similaire à celle utilisée dans [BF01] (Figure 1.9). Mais contrairement à ces derniers, le modèle proposé n'a pas de résolution fixe, ce qui permet d'augmenter localement la définition du maillage uniquement aux endroits où cela est utile. À l'aide d'un chemin proposé par l'utilisateur, Peyrat *et al.* [PTM+11] génèrent une trajectoire en prenant en compte la vitesse de déplacement et les emplacements des empreintes. En utilisant un motif sous forme de textures, le maillage est raffiné aux emplacements des empreintes puis extrudé pour créer un enfoncement dans le sol en prenant en compte les couches de matériaux sous-jacents. La méthode décrite permet de transférer de la matière sur la trajectoire comme si elle était accrochée à un pied ou une roue (Figure 1.8b).

La modélisation surfacique ne permet de gérer que l'érosion verticale. En effet, seules les couches sous-jacentes sont répertoriées, sans information sur les couches adjacentes dans les autres directions. De plus, un déplacement latéral des sommets peut entraîner des incohérences géométriques du maillage. Avec ces modèles, il est donc impossible de créer des cavernes, des surplombs ou des arches.

Dans [BSP+04], Brandel *et al.* se basent sur les données relatives aux événements géologiques pour recréer la structure du terrain. La géométrie générée est une représentation figée du terrain. À partir des failles et des horizons (surfaces séparant deux couches de matériaux différents), ils déterminent la chronologie des événements et reconstruisent des partitions de l'espace en utilisant un modèle de 3-G-Cartes et des opérations booléennes afin de découper les volumes représentant les couches du terrain (Figure 1.10).

(a) (b)

FIGURE 1.8 – Résultats obtenus par la méthode de Peyrat *et al.* ([PTM+11]). (a) Une empreinte de pas dans du sable. (b) Empreinte d'un véhicule traversant une route et laissant une trace.

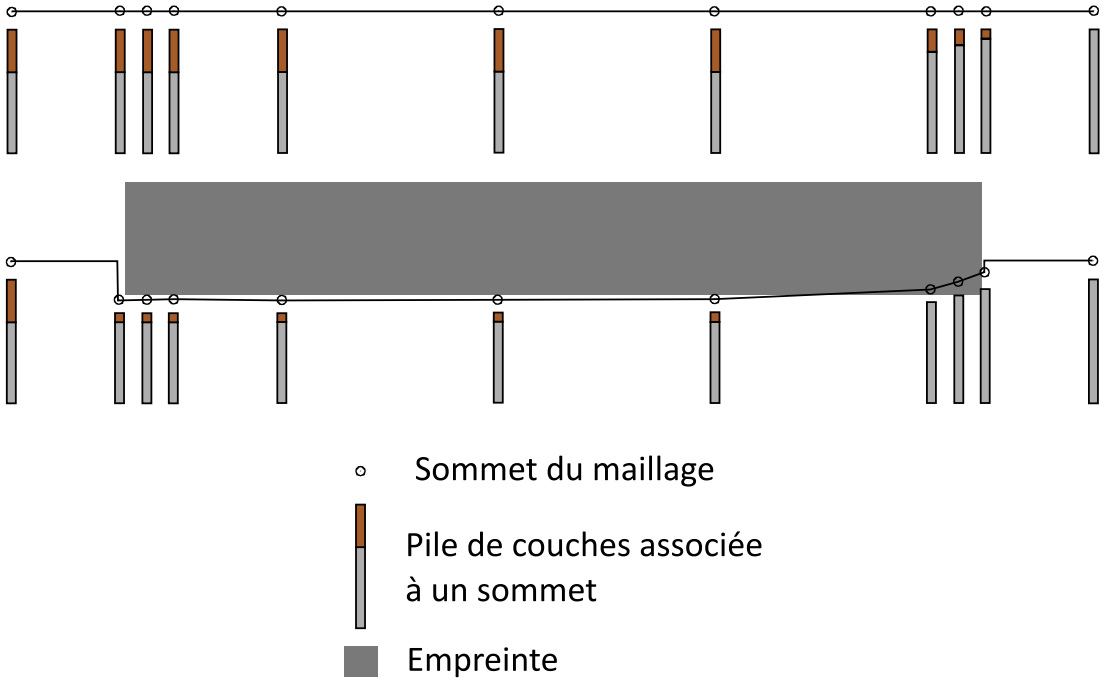
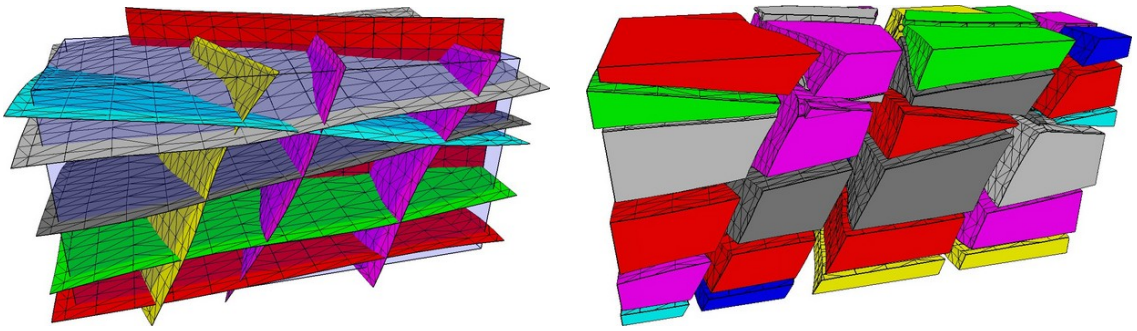


FIGURE 1.9 – Enfoncement d'une empreinte en tenant compte des couches de matériaux. En haut la structure du sol intacte, en bas la même structure altérée par une empreinte plate ([PTM+11]).



(a) (b)

FIGURE 1.10 – Exemple de construction d'un terrain en couches en utilisant les opérations booléennes ([BSP+04]).

Un modèle d'animation basé sur la topologie est présenté dans [LSM08]. Ce modèle utilise un mécanisme événementiel pour détecter les incohérences topologiques, l'animation de la partition de l'espace considérée est générée à partir du traitement séquentiel de tous les événements. Le traitement génère une carte généralisée représentant un ensemble de modifications topologiques simultanées et supposées instantanées. L'utilisateur fournit un scénario d'événements globaux associés à un contexte particulier. Par exemple, dans le cas d'une application à la géologie, des phases d'érosion ou de sédimentation, des créations de failles et des glissements de blocs géologiques. Ce modèle intègre également une description temporelle des évolutions : chaque entité géométrique possède un label propre qui est modifié lors des évolutions si l'entité subit des modifications. Ces labels permettent également de désigner les entités qui subissent des modifications lors de l'élaboration des scénarios. L'approche de Léon *et al.* [LSM08] est limitée à la 2D (Figure 1.11). Les incohérences topologiques et les modifications topologiques de la carte sont restreintes à cette dimension : intersection sommet / sommet, sommet / arête ou arête / arête. Le passage à la 3D augmente les cas à gérer (intersection sommet / sommet, sommet / arête, sommet / face, arête / arête, arête / face et face / face) et leurs solutions sont plus complexes à mettre en place. Nous proposons dans la section 3 une étude des opérations topologiques 3D permettant de gérer ces incohérences.

Les méthodes décrites dans cette partie permettent de représenter statiquement un terrain surfacique ou volumique. La géomorphologie implique des modifications du terrain au cours du temps. Bien que les méthodes basées sur les voxels ou en strates permettent de représenter aussi bien des arches, des grottes ou des lits de rivières, ces structures sont gourmandes en mémoire et sont limitées à la taille et à la résolution de la grille. Cependant, les méthodes utilisant des maillages surfaciques ne prennent pas en compte les modifications latérales du terrain. Nous avons vu que les cartes généralisées permettent de représenter le terrain en couches, et qu'il est possible de faire évoluer la géométrie et la topologie du terrain, que ce soit verticalement ou latéralement. Néanmoins, pour obtenir une simulation réaliste, l'évolution du terrain doit être cohérente avec les phénomènes physiques et chimiques que sont l'érosion, la sédimentation et le transport des sédiments. La partie suivante traite de la représentation du fluide ainsi que des méthodes d'interaction entre le fluide et le terrain, toutes deux inspirées de la physique.

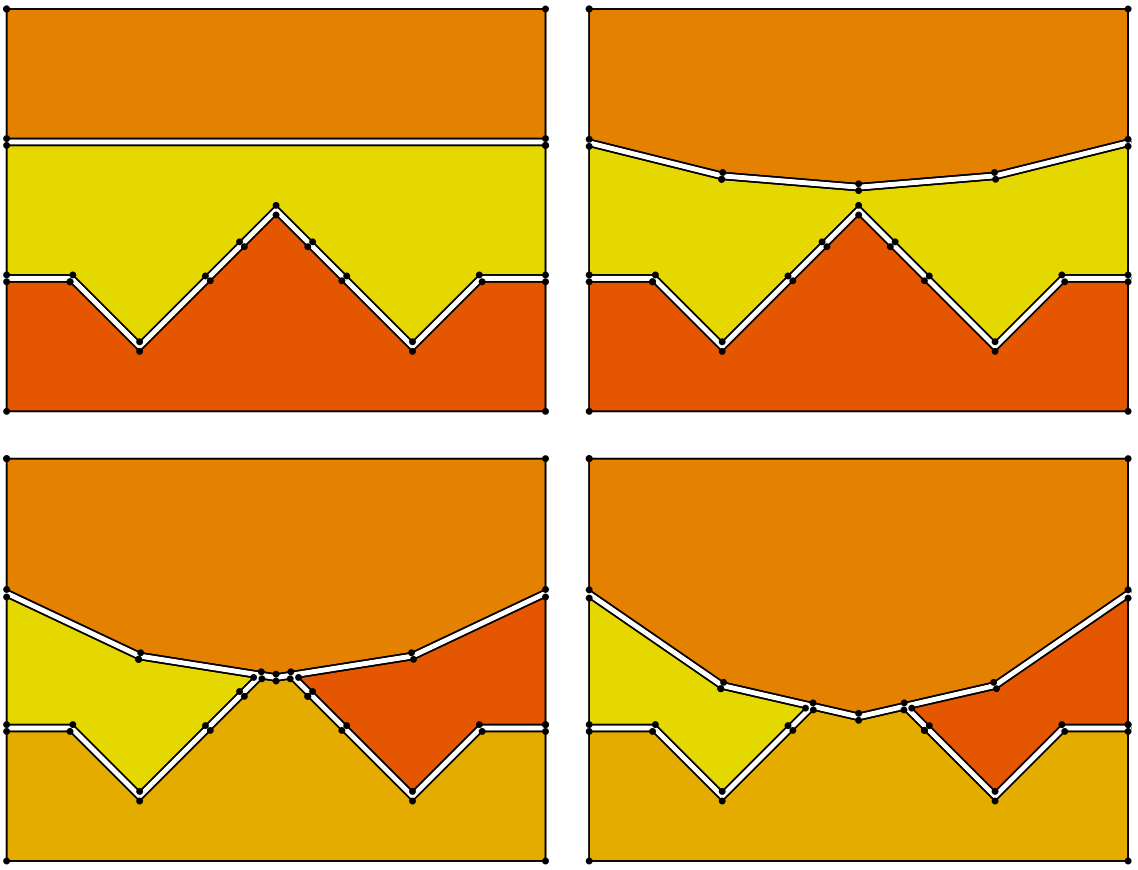


FIGURE 1.11 – Résultat obtenu par [LSM08] à partir d'un scénario décrivant l'état initial de la scène et son évolution au travers d'une succession de phénomènes géologiques : Sédimentation, Érosion, Création de failles et Glissement.

1.2 Méthodes inspirées de la physique

Les paysages que nous connaissons sont issus d'une succession de modifications physiques au cours du temps. Afin d'augmenter le réalisme visuel des terrains générés, plusieurs approches s'inspirent de la physique en simulant l'érosion du terrain, le transport et le dépôt des sédiments par un fluide. Ces approches s'appuient sur la mécanique des fluides afin d'obtenir un écoulement de fluide visuellement réaliste. Dans un premier temps, nous détaillons plusieurs méthodes de simulation de fluide et dans un second temps, nous décrivons des méthodes d'interaction entre le fluide et le terrain.

1.2.1 Simulation de fluide

L'animation d'un fluide est un problème important en synthèse d'images quand on souhaite simuler des phénomènes tels que la pluie, la boue, des vagues, *etc*. Nous décrivons dans cette partie deux catégories d'approches : l'une qui se base sur un modèle discret du fluide, et l'autre sur la mécanique des milieux continus. Nous expliquons également des méthodes de résolution de ces modèles.

1.2.1.1 Modèles

Discret

Le modèle « Discrete Element Method » (DEM) est principalement utilisé dans les domaines de simulation de matériaux granuleux et de la dynamique des molécules. Le fluide est discrétisé en particules. Leur mouvement est déterminé par le calcul des forces agissant entre chaque paire de particules. Ces forces peuvent être de différentes natures : force élastique f^{spring} modélisée par un ressort linéaire et force d'amortissement $f^{damping}$ qui dissipe l'énergie entre les particules. Ces forces sont calculées pour une particule i en collision avec une particule j en utilisant les équations :

$$f_i^{spring} = -k(d - |r_{ij}|)\frac{r_{ij}}{|r_{ij}|} \tag{1.1}$$

$$f_i^{damping} = \eta v_{ij} \tag{1.2}$$

où k, η, d, r_{ij} et v_{ij} sont respectivement les coefficients d'élasticité, d'amortissement, le diamètre de la particule, la position relative et la vélocité de la particule j par rapport à la particule i. L'équation 1.1 implique que deux particules sont en collision quand $|r_{ij}| < d$. Une force de cisaillement est modélisée comme une force proportionnelle à la vélocité tangentielle relative des particules v_{ij}^t :

$$f_i^{shear} = k^t v_{ij}^t \tag{1.3}$$

où k^t est une constante. La vélocité tangentielle relative est calculée par :

$$v_{ij}^t = v_{ij} - \left(v_{ij} \cdot \frac{r_{ij}}{|r_{ij}|}\right)\frac{r_{ij}}{|r_{ij}|} \tag{1.4}$$

La force qui s'applique sur une particule j est la somme de ces trois forces calculées pour chacune des particules i du voisinage de j :

$$F_j = \sum_i \left(f_i^{spring} + f_i^{damping} + f_i^{shear}\right) \tag{1.5}$$

Le potentiel de Lennard-Jones approxime l'interaction entre deux particules i et j selon la distance r_{ij} qui les sépare. Il a été utilisé pour modéliser un fluide dans [TPF89] et [MP89]. Il est décomposé en deux parties : un terme de répulsion forte et un terme d'attraction douce. Cette approche est simple : elle ne prend en compte que deux paramètres : σ la distance à laquelle le potentiel est à 0, et ε la valeur minimum du potentiel atteinte quand r_{ij} est égale à la distance d'équilibre r_0 (Figure 1.12). Cependant, la force de répulsion possède une amplitude très grande quand les deux particules sont trop proches, ce qui est mal supporté par les systèmes d'intégration numérique et crée une instabilité. Cette instabilité est accrue avec le nombre de particules qui composent le système.

$$\Phi(r_{ij}) = 4\varepsilon\left[\left(\frac{\sigma}{r_{ij}}\right)^{12} - \left(\frac{\sigma}{r_{ij}}\right)^{6}\right] \tag{1.6}$$

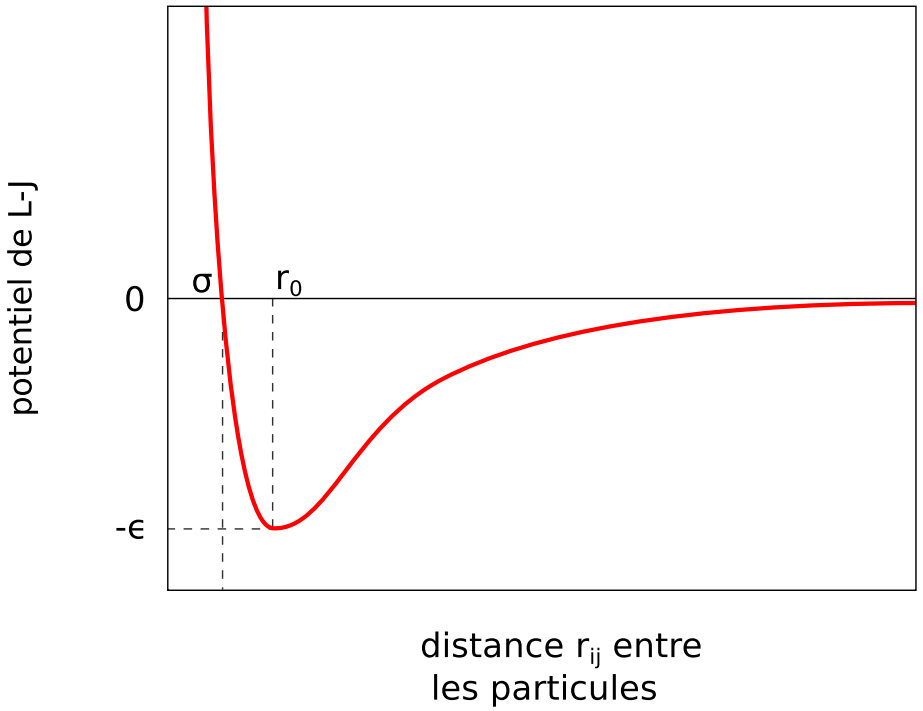


FIGURE 1.12 – Potentiel de Lennard-Jones.

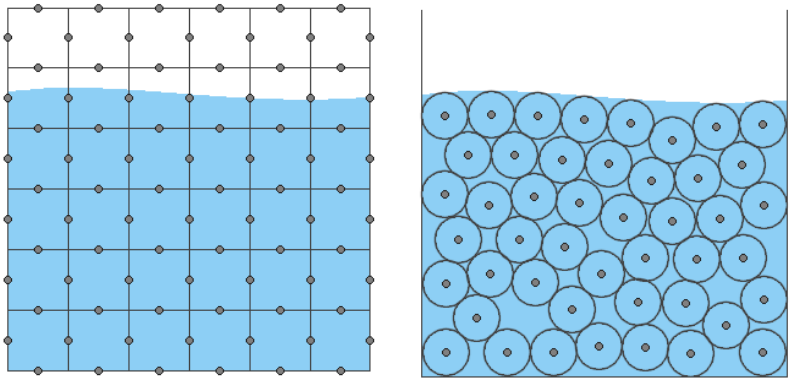
Milieux continus

D'autres modèles s'appuient sur la mécanique des milieux continus à travers la dynamique des fluides. Les équations de Navier-Stokes décrivent le mouvement d'un fluide incompressible au cours du temps t. Les principales quantités physiques d'un fluide visqueux et isotherme sont la vélocité u, la densité massique ρ et la pression p. Ces quantités sont considérées comme étant un champ continu dans le fluide.

$$\rho\left(\frac{\partial}{\partial t} + u \cdot \nabla\right) u = -\nabla p + \mu \nabla \cdot (\nabla u) + f_{ext} \tag{1.7}$$

$$\nabla \cdot u = 0 \tag{1.8}$$

où ∇ est l'opérateur différentiel *nabla*, μ est la viscosité du fluide et f_{ext} représente la somme des forces de densité volumique externes comme l'attraction gravitationnelle g. Cette formulation de l'écoulement du fluide est basée sur une grille. L'équation 1.7 décrit la conservation du moment et exprime la deuxième loi de Newton pour un fluide. Elle indique comment le fluide modifie sa vitesse à cause des forces qui s'exercent sur lui. Le membre droit de l'équation représente la somme des forces de densité volumique agissant sur le fluide et le membre gauche représente le produit de la densité massique et de la densité d'accélération. L'équation 1.8 décrit la conservation de la masse d'un fluide incompressible et indique que le fluide ne doit ni gagner ni perdre de masse au cours de son évolution.



(a) (b)

FIGURE 1.13 – (a) Un fluide représenté selon le point de vue eulérien. (b) Le même fluide représenté selon le point de vue lagrangien. (extrait de [Kel06]).

1.2.1.2 Méthodes de résolution

Les équations de Navier-Stokes en mécanique des fluides sont à la base des simulations de ces dernières années. Deux principales implémentations, issues de la géométrie analytique, existent pour résoudre ces équations : une méthode basée sur une grille, appelée méthode eulérienne, et une méthode basée sur des particules, appelée méthode lagrangienne.

Approche eulérienne

L'approche eulérienne est basée sur l'évaluation des équations de Navier-Stokes en des points fixes de l'espace répartis sur une structure de grille régulière. Les quantités d'un champ ne dépendent pas seulement du temps, mais aussi de la position de la grille, qui elle-même, dépend du temps. [FM96] de Foster *et al.* est l'un des premiers articles en informatique graphique à simuler un fluide en trois dimensions. Les auteurs utilisent une grille uniforme pouvant contenir des obstacles entourés par le fluide. Les équations de Navier-Stokes sont approchées en utilisant la méthode des différences finies sur une grille de faible résolution. La vélocité est définie au centre de chacune des faces de la cellule alors que la pression est définie au centre de la cellule. La nouvelle vélocité est calculée pour chaque cellule en utilisant les valeurs de vélocité et de pression de l'étape précédente, il faut ensuite vérifier que chaque cellule respecte l'équation 1.8 de conservation de masse. La pression à l'intérieur des cellules est ajustée selon la divergence des vélocités et la vélocité sur les faces des cellules est ajustée selon cette pression. Foster *et al.* font coïncider les obstacles avec les faces des cellules afin de simplifier la simulation et les calculs au niveau des interfaces liquide / obstacle.

D'autres articles comme [Sta99] présentent une méthode de simulation de fluide eulérienne permettant de simuler interactivement des fluides similaires à de la fumée. Cette fois, les quantités du fluide sont toutes discrétisées au centre des cellules. Il est possible d'interagir en temps interactif avec le fluide. Ces méthodes ne permettent pas de simuler des phénomènes d'éclaboussure comme une vague qui rencontre un rocher. La simulation de fluide est limitée à la grille uniforme : sa taille et sa résolution. Elle doit englober la

scène avec un niveau de détails fixe aussi bien dans les zones nécessitant un raffinement plus fin que dans les zones où le niveau de détails peut être plus grossier.

Approche lagrangienne

La méthode lagrangienne utilise des particules au lieu d'une grille pour représenter le fluide, les équations de Navier-Stokes en sont simplifiées. Le nombre de particules est constant et chaque particule possède une masse constante sur son volume : la conservation de la masse globale du fluide est donc garantie et la résolution de l'équation 1.8 peut être omise. L'incompressibilité doit être contrainte de manière locale. L'expression $\partial u/\partial t + u \cdot \nabla u$ de l'équation 1.7 peut être remplacée par la dérivée substantielle Du/Dt. Les particules composent le fluide, la dérivée substantielle est donc simplement une dérivée temporelle de la vélocité des particules. Ainsi, le terme $u \cdot \nabla u$ n'est pas nécessaire dans le cas d'un système de particules. Les forces de densité volumique présentes dans l'équation 1.7 modélisent les forces de pression $f^{pression}$, de viscosité $f^{viscosite}$ et les forces externes. La somme de ces forces $f = -\nabla p + \rho g + \mu \nabla^2 u$ détermine le changement du moment $\rho Du/Dt$ pour les particules. L'accélération d'une particule a_i se calcule ainsi :

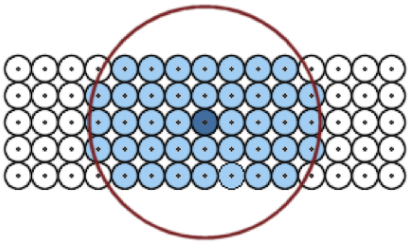
$$a_i = \frac{du_i}{dt} = \frac{f_i}{\rho_i} \tag{1.9}$$

où f_i est la somme des forces s'appliquant sur cette particule et ρ_i la densité de fluide représentée par cette particule.
Chaque particule est déplacée selon un pas de temps fixe Δt en utilisant l'accélération précédemment calculée, et la nouvelle position des particules est obtenue en intégrant l'accélération numériquement. Plusieurs méthodes d'intégration numérique existent comme *Euler explicite*, *Euler implicite*, *Leapfrog*, *Runge-Kutta*, *etc* [HNW93].

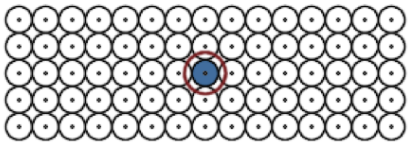
Smoothed Particles Hydrodynamics La méthode des Smoothed Particles Hydrodynamics (SPH) est à l'origine une méthode de simulation de phénomènes astrophysiques [Mon92]. Il s'agit d'une méthode d'interpolation basée sur des systèmes de particules, elle a été introduite dans le cadre de la simulation de fluide par Desbrun *et al.* [DGC96]. Chaque quantité d'un champ, seulement défini en des points discrétisés de l'espace peut être évaluée pour n'importe quel point de l'espace. Pour cela, la méthode évalue ces quantités à partir de chacune des particules du voisinage en utilisant des fonctions appelées noyaux d'interpolation. Une quantité scalaire A est interpolée à une position de l'espace r en effectuant une moyenne pondérée de la contribution de chacune des particules du voisinage :

$$A_S(r) = \sum_j V_j A_j W\left(r - r_j, h\right) \tag{1.10}$$

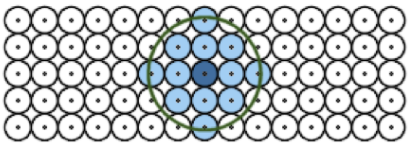
où j est une particule du voisinage, V_j son volume, r_j sa position et A_j la quantité scalaire à la position r_j. La fonction $W(r, h)$ est la fonction d'interpolation dont le rayon d'interaction h délimite la zone dans laquelle les particules ont une influence (Figure 1.14).



(a) Rayon d'interaction trop large.



(b) Rayon d'interaction trop petit.



(c) Rayon d'interaction correct.

FIGURE 1.14 – Une valeur correcte du rayon d'interaction est indispensable pour une simulation de fluide stable, la précision et les temps de calculs (extrait de [Kel06]).

Pour résoudre les équations de Navier-Stokes, nous avons également besoin du gradient et du laplacien de quantité. Le gradient d'une quantité A est évalué par :

$$\nabla A_S(r) = \sum_j V_j A_j \nabla W\left(r - r_j, h\right) \tag{1.11}$$

Le laplacien est évalué par :

$$\nabla^2 A_S(r) = \sum_j V_j A_j \nabla^2 W\left(r - r_j, h\right) \tag{1.12}$$

Évaluation de la densité de masse Le volume V d'une particule est obtenu par la relation suivante :

$$V = \frac{m}{\rho} \tag{1.13}$$

où m est la masse de la particule et ρ sa densité.
Alors que la densité d'une particule change au cours de la simulation, sa masse est constante. La densité doit donc être évaluée à chaque pas de temps en utilisant l'équation 1.10 :

$$\rho(r_i) = \sum_j m_j \frac{\rho_j}{\rho_j} W\left(r_i - r_j, h\right) = \sum_j m_j W\left(r_i - r_j, h\right) \tag{1.14}$$

où r_i est la position de la particule i et r_j la position de la particule j.

Évaluation de la force de pression En prenant en compte le fait que la force de pression $f^{pression}$ entre deux particules i et j est symétrique, Müller *et al.* [MCG03] proposent l'équation suivante à partir de l'équation 1.11 :

$$f_i^{pression} = -\sum_{j \neq i} \frac{p_i + p_j}{2} \frac{m_j}{\rho_j} \nabla W\left(r_i - r_j, h\right) \tag{1.15}$$

Un fluide comme l'eau doit garder une cohésion interne et chercher à atteindre une densité proche de sa densité au repos ρ_O, ainsi la pression p est évaluée en utilisant la loi des gaz parfaits :

$$p = k(\rho - \rho_O) \tag{1.16}$$

où k est une constante du gaz qui dépend de la température.

Évaluation de la force de viscosité Le terme responsable de la force de viscosité $\mu\nabla^2 v$ est évalué en utilisant l'équation 1.12. La force de viscosité $f^{viscosite}$ ne dépendant que des différences de vélocité entre les particules, il est possible de rendre l'équation symétrique de manière simple :

$$f_i^{viscosite} = \sum_j m_j \frac{v_j - v_i}{\rho_j} \nabla^2 W\left(r_j - r_i, h\right) \tag{1.17}$$

La particule i est accélérée dans la direction de la vélocité relative de ses voisines.

Fonction d'interpolation W La stabilité des SPH repose sur la fonction d'interpolation W. Cette fonction détermine la contribution d'une particule par rapport à une autre en fonction de la distance qui les sépare. Elle doit répondre à plusieurs contraintes : elle doit être normalisée, positive, symétrique et de valeur nulle au-delà du rayon de support h. Müller *et al.* [MCG03] proposent les fonctions suivantes pour les équations de densité (1.14), de pression (1.15) et de viscosité (1.17), respectivement :

$$W_{defaut}(r,h) = \frac{315}{64\pi h^9} \begin{cases} (h^2 - \|r\|^2)^3 & 0 \leq \|r\| \leq h \\ 0 & \|r\| \geq h, \end{cases} \tag{1.18}$$

$$\begin{aligned} W_{pression}(r,h) &= \frac{15}{\pi h^6} \begin{cases} (h - \|r\|)^3 & 0 \leq \|r\| \leq h \\ 0 & r \geq h, \end{cases} \\ \nabla W_{pression}(r,h) &= -\frac{45}{\pi h^6} \frac{r}{\|r\|} (h - \|r\|)^2 \end{aligned} \tag{1.19}$$

$$\begin{aligned} W_{viscosite}(r,h) &= \frac{15}{2\pi h^3} \begin{cases} -\frac{\|r\|^3}{2h^3} + \frac{\|r\|^2}{h^2} + \frac{h}{2\|r\|} - 1 & 0 \leq \|r\| \leq h \\ 0 & r \geq h, \end{cases} \\ \nabla^2_{viscosite}(r,h) &= \frac{45}{\pi h^6}(h - \|r\|) \end{aligned} \tag{1.20}$$

Pour que le fluide simulé à l'aide de la méthode des SPH soit stable, chaque particule de fluide doit posséder x voisines en fonction du rayon du support h (Équation 1.21).

$$x = \left(\frac{n}{V}\right) \frac{4}{3}\pi h^3 \tag{1.21}$$

avec n le nombre total de particules que contient la simulation et V le volume total de fluide que représentent les n particules.

Cela pose problème au niveau des bords du fluide, là où les particules n'ont pas assez de voisines pour garantir la stabilité de la méthode. De plus, il est difficile d'estimer l'interaction entre le fluide et un support sur lequel il s'écoule. Müller *et al.* [MST+04b] [MST04a] discrétisent le support en particules alors que Harada *et al.* [HKK07] modifient les fonctions d'interpolation pour estimer la contribution du support.

1.2.2 Interaction avec le terrain

L'interaction entre le fluide et le terrain détermine la position des modifications et leur intensité. Il est donc essentiel qu'elle soit cohérente. Elle doit tenir compte des représentations du terrain et du fluide qui influent également sur le type d'interaction. Dans une simulation physique d'évolution de terrain, le modèle de fluide doit prendre en compte les échanges de matériaux avec le terrain, par exemple pour simuler l'érosion et la sédimentation.

Musgrave *et al.* [MKM89] distinguent deux types d'érosion. La première est l'érosion hydraulique qui est causée par l'écoulement de l'eau. La seconde, "thermal weathering", regroupe tous les phénomènes non-hydrauliques qui causent la détérioration des roches et qui créent des pentes raides et des éboulis. Les auteurs utilisent une carte de hauteur créée à l'aide de fractales pour représenter le terrain. Sont associés à chaque sommet à l'instant t et une altitude donnée, un volume d'eau et une quantité de sédiment en suspension dans l'eau. L'eau et les sédiments en excès au niveau du sommet sont transférés à chaque sommet voisin ayant une altitude inférieure. Musgrave *et al.* introduisent une capacité maximale de transport de sédiment par unité d'eau, une constante de dépôt de sédiments et une constante de résistance du sol. En conséquence, l'eau et les sédiments des sommets les plus élevés sont transportés et déposés vers les zones les moins élevées. Pour simuler le second processus d'érosion, Musgrave *et al.* comparent la différence d'altitude avec celle de ses voisins. Si la pente est supérieure à un certain angle, alors un pourcentage de cette différence est déplacé sur le sommet voisin. Olsen utilise le même type d'érosion dans [Ols04] (Figure 1.15). Dans [Mus93], Musgrave rajoute un troisième type d'érosion : la diffusion latérale. Il s'agit d'un filtre spatial qui arrondit les angles formés par les crêtes et les vallées. Dans [BF01], Benes *et al.* adaptent l'érosion hydraulique à leur structure en couches. Ils ajoutent la notion de densité de la matière : lorsque de la roche est transformée en sédiment, la densité des sédiments diminue pour qu'ils puissent être transportés.

Chiba *et al.* [CMF98] simulent le ruissellement de l'eau sur un terrain défini par un modèle de cartes de hauteur. Chiba *et al.* utilisent le champ de vitesse de l'eau pour calculer l'énergie e issue de l'interaction du fluide et du terrain, et le débit du ruissellement Q. L'érosion, le transport et la sédimentation sont calculés en fonction de e, Q et les constantes définies par l'utilisateur K_e et K_Q, qui permettent de pondérer les calculs afin d'obtenir des résultats visuellement différents. La quantité maximale de sédiments transportable S_{max} par un fluide ayant un débit Q et la quantité de sédiments transportés S_t sont calculées à chaque pas de temps. Si $S_t > S_{max}$, alors la différence est ajoutée à l'élévation, sinon

FIGURE 1.15 – Résultats obtenus par la méthode de Olsen ([Ols04]).

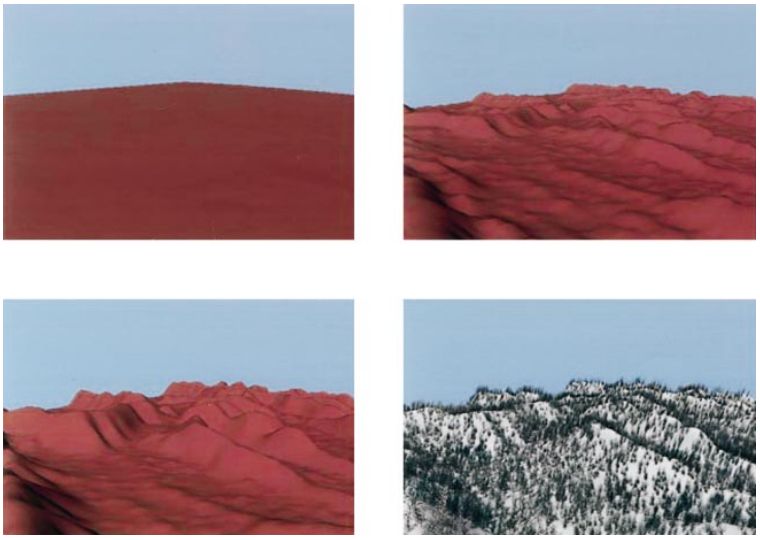


FIGURE 1.16 – Résultats obtenus par la méthode de Chiba *et al.* ([CMF98]).

une quantité de sédiments est transférée au fluide à partir de l'élévation du terrain. Cette méthode permet de générer des vallées et des crêtes (Figure 1.16).

Dans [BF02], Benes *et al.* reprennent leur modèle de [BF01] et divisent le processus d'érosion en quatre parties. L'eau est créée à partir de sources ou sous forme de pluie en augmentant la "couche d'eau" des piles de leur modèle par une quantité K_r. L'eau peut également être évaporée selon la température et le niveau d'eau en retirant une quantité K_e d'eau à la pile. L'eau érode le terrain et capture des sédiments selon les mêmes principes que [CMF98]. Puis l'eau transporte les sédiments capturés. Benes *et al.* utilisent un modèle de diffusion pour transférer l'eau entre les piles de couches (Figure 1.17). L'eau dépose les sédiments dissous sur le sol si elle atteint le taux maximal de saturation en sédiments. L'évaporation de l'eau joue ici un rôle : elle diminue le volume d'eau, la concentration en sédiments augmente et une partie est déposée sur le sol.

Neidhold *et al.* [NWD05] présentent une méthode combinant une simulation de fluide simplifiée à un algorithme d'érosion en temps interactif. En plus des données d'élévation inspirées de la méthode de [BF01], ils stockent pour chaque cellule de la grille des informa-

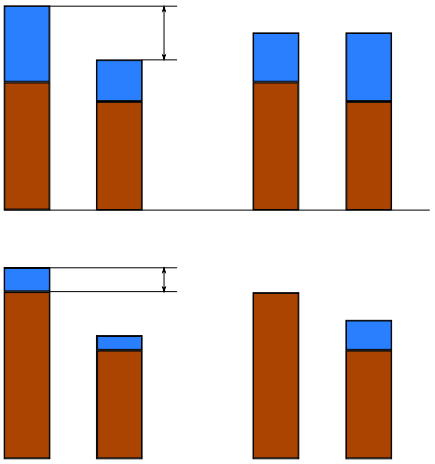
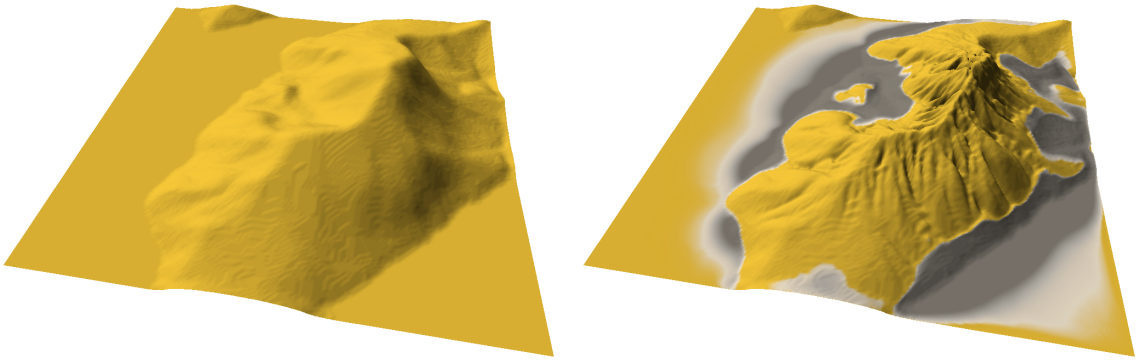


FIGURE 1.17 – Deux cas de diffusion de l'eau (haut et bas) de la méthode de Benes *et al.* ([BF02]). Les dessins de gauche montrent la situation avant et ceux de droite après la diffusion.

tions relatives à la simulation de fluide : la quantité de fluide, la vélocité et l'accélération, ainsi que la quantité de matière dissoute. La résolution des équations de Navier-Stockes faite par Neidhold *et al.* [NWD05] est suffisante pour la simulation d'érosion. L'accélération est calculée en fonction de la moyenne de la pente avec les cellules voisines moins élevées. Si la vélocité est inférieure ou proche de zéro, le fluide est seulement accéléré dans la direction de la pente la plus forte. Leur méthode d'érosion prend en compte la quantité de fluide et la quantité de sédiment dissoute dans ce fluide. Sans érosion, ces quantités sont respectivement ajoutées à l'élévation et à la quantité de matière dissoute de la cellule. Si l'érosion a lieu, la méthode de transfert est modifiée de manière à déposer ou à dissoudre une partie des sédiments depuis la cellule considérée vers la cellule de destination. Neidhold *et al.* utilisent un système de dépôt et de dissolution comparable à [CMF98] en utilisant la capacité de transport calculée à partir de la quantité de fluide et sa vélocité. Pour générer de la pluie, ils ajoutent une quantité de fluide aux cellules de la grille à intervalle constant ou aléatoire en utilisant un bruit pour définir les cellules arrosées. Dans le cas des sources, l'utilisateur définit ces dernières dans l'environnement de la simulation. Pour chaque source, une quantité d'eau est ajoutée à chaque cellule en utilisant une distribution gaussienne. L'utilisateur peut aussi "peindre" de l'eau interactivement sur la simulation. L'eau peut disparaître de deux manières : en sortant du domaine de la simulation ou par évaporation, la technique utilisée est celle de [BF02].

Benes *et al.* [BTHB06] se basent sur une grille régulière de voxels de type air, liquide et matière pour représenter le terrain. La résolution des équations de Navier-Stockes leur permet d'évaluer le champ de vitesse de chacune des faces des voxels ainsi que la pression exercée au centre des voxels. Si un voxel est rempli exclusivement d'eau, il est marqué comme FULL, s'il ne contient que de l'air, il est marqué comme EMPTY, et s'il contient de la matière il est marqué comme MAT. La pression et la vélocité dépendent de la viscosité de la matière et de la vélocité du liquide des voxels voisins. Benes *et al.* définissent un scalaire m. Dans le cas d'un voxel FULL, m représente la quantité de matière dissoute dans l'eau, et dans le cas d'un voxel MAT la quantité de matière contenue dans le voxel comprise entre 0 et 1. Un voxel d'eau ne peut contenir plus de sédiment que le taux de saturation



(a) (b)

FIGURE 1.18 – Résultats obtenus par Neidhold *et al.* ([NWD05]). (a) Chaine de montagne d'origine. (b) Chaine de montagne érodée.

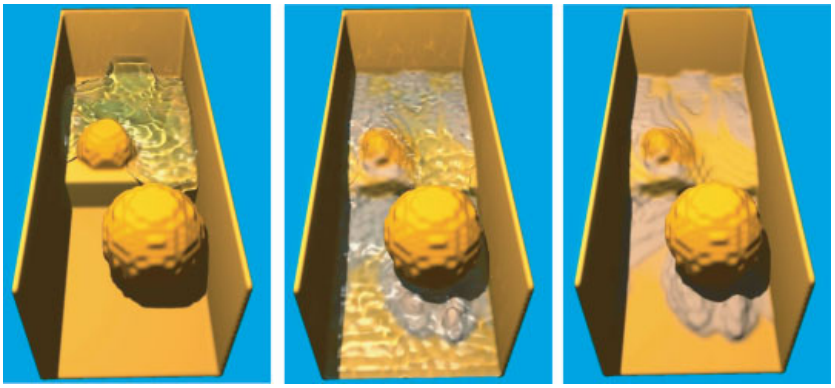


FIGURE 1.19 – Résultats obtenus par Benes *et al.* ([BTHB06]).

maximum c_{max}. Un voxel FULL doit satisfaire la condition $0 \leq m < c_{max}$. Un voxel peut changer d'état suivant deux transitions : FULL $\Rightarrow$ MAT décrit le dépôt de matière, MAT $\Rightarrow$ FULL décrit l'érosion. Le dépôt s'effectue toujours sur le bas du voxel.

Valette *et al.* [VPLL06] proposent une méthode de détérioration d'un terrain par la pluie à l'échelle du mètre. Ils utilisent le même type de représentation de terrain que [BTHB06], mais ils définissent un plus grand nombre d'état pour les voxels, ainsi que différentes règles de transitions d'un état à un autre. Ces règles simulent chacune un processus : la création et le transport de fragments de matière en simulant l'effet de *splash* provoqué par la chute des gouttes de pluie, l'évaporation, l'infiltration et le ruissellement de l'eau, ainsi que le dépôt des fragments de matière.

Mei *et al.* [MDH07] adaptent une méthode similaire à [CMF98], [BF02] et [NWD05] sur carte graphique, ce qui améliore les temps de calculs. La capacité de transport du fluide est liée à la géométrie du terrain et à la vélocité du fluide. Stava *et al.* [SBBK08] reprennent la méthode d'érosion de [NWD05] et [MDH07] sur GPU en intégrant un modèle de « shallow-water » : l'eau est discrétisée en colonnes qui communiquent entre elles. Quand une quantité de sédiments déposés n'est pas modifiée pendant une longue durée, les sédiments sont transformés en une matière plus résistante de la couche immédiatement

inférieure.

Krištof *et al.* [KBKv09] utilisent un fluide pour éroder, sédimenter et transporter le terrain représenté par un maillage (voir section 1.1.3). Ils résolvent les équations de Navier-Stockes à l'aide de la méthode des Smoothed Particle Hydrodynamics pour simuler l'écoulement de l'eau. Pour gérer les interactions entre le terrain et le fluide, ils utilisent des particules de bords. Les triangles du maillage sont discrétisés en utilisant l'algorithme *scanline* avec un espace fixe. Une force de "non-pénétration" est ajoutée pour que le fluide ne passe pas à travers les particules de bords. Pour repousser les particules de fluide du bord, ils utilisent la méthode de [Ama06] en fonction du vecteur vélocité de la particule de fluide, de la normale de la surface, d'un coefficient de rigidité et d'un coefficient d'amortissement. L'écoulement du fluide implique une force de cisaillement sur les bords du solide, cette force est parallèle à l'écoulement. Krištof *et al.* utilisent l'approche de [WCMT07] pour cette force en fonction de la vélocité relative entre les particules de fluide et de bords. La relation entre la force de cisaillement τ et la vitesse d'érosion ε est donnée par [Par65] et prend en compte la résistance à l'érosion de la matière τ_c

$$\varepsilon = K_\varepsilon(\tau - \tau_c)$$

où K_ε est un coefficient représentant la puissance de l'érosion.
Krištof *et al.* introduisent un modèle de donateur-accepteur. Une particule de fluide i peut être soit donateur soit accepteur dans chaque relation avec une autre particule j. À cause des transferts multiples simultanés entre les particules voisines, il est nécessaire de vérifier que la quantité de sédiments transportée par une particule de fluide n'excède pas la quantité maximale transportable. La méthode permet aussi de diffuser les sédiments à travers les particules de fluide en se basant sur [Jen08] qui décrit l'équation générale du processus de transport de sédiments dissous dans un fluide. Krištof *et al.* ont intégré cette équation dans celle de diffusion fournie par [Mon05]. Quand une particule est assez proche du sol, elle transfère les sédiments capturés vers les particules de bords. La quantité transférée est calculée à partir de la densité de la matière et du volume de fluide. Les modifications du terrain suite à l'érosion ont été détaillées dans la section 1.1.3.

Nous avons détaillé plusieurs méthodes d'altération du terrain. Elles apportent des techniques inspirées de la physique permettant au fluide d'éroder, de transporter et de déposer des sédiments. La méthode de Krištof *et al.* [KBKv09] reprend toutes ces approches pour les appliquer au système de particules en y ajoutant la diffusion des sédiments au sein de fluide. Comme nous l'avons montré, les systèmes de particules ont l'avantage de ne pas être limités à la taille et la résolution d'une grille, contrairement aux approches eulériennes. De plus, les systèmes de particules permettent de simuler un grand nombre de phénomènes hydrologiques comme les chutes d'eau. Pour ces raisons, nous nous sommes inspirés du modèle de fluide utilisé par Krištof *et al.* , combiné à la robustesse des cartes généralisées pour la représentation du terrain.
La partie suivante détaille le modèle des cartes généralisées et des opérations topologiques de base. Une approche théorique de l'érosion utilisant les opérations booléennes sur les cartes est également critiquée.

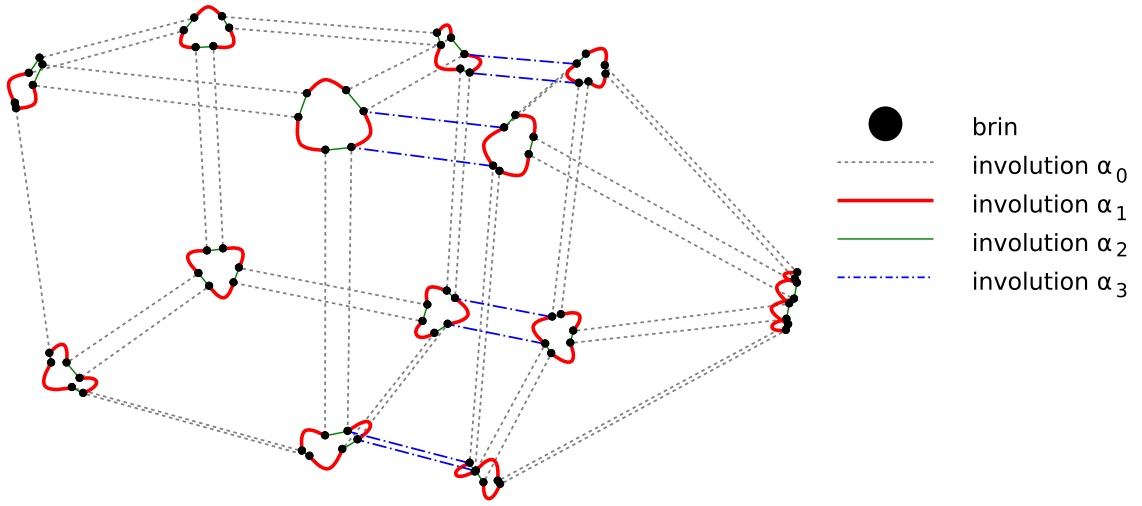


FIGURE 1.20 – Représentation éclatée d'une 3-G-Carte.

1.3 Cartes généralisées

Les cartes généralisées ou G-Cartes font partie des modèles de représentation par frontière (B-Rep pour Boundary Representation). Ce modèle, introduit par Pascal Lienhardt [Lie94], permet de représenter des subdivisions de $\mathbb{R}^n$ en cellules de dimension i (i-cellules), $i \in [0, n]$. Il s'agit d'une extension des cartes combinatoires [Tut84] (Figure 1.20).

1.3.1 Description du modèle

Une carte généralisée de dimension n (ou « n-G-Carte », abrégée encore par « G-Carte ») permet de représenter une quasi-variété cellulaire orientable ou non, avec ou sans bord. Une quasi-variété de dimension n est un objet de dimension n obtenu par assemblage de n-cellules le long de $(n-1)$-cellules. Une $(n-1)$-cellule ne peut pas appartenir au bord de plus de deux n-cellules. Une G-Carte se compose d'éléments de base appelés brins liés entre eux par des involutions.

Définition 1. *Une carte généralisée de dimension n $G = (B, \alpha_0, \alpha_1, \ldots, \alpha_n)$ est définie par :*

- *B un ensemble de brins,*
- *$(\alpha_i)_i = 0, \ldots, n$ des bijections de B dans B telles que $\forall i, \alpha_i$ est une involution ($\alpha_i \circ \alpha_i = id$),*
- *$\forall i, j$ tels que $j \geq i + 2$ alors $\alpha_i \circ \alpha_j$ est une involution.*

Dans la suite du document, nous noterons α_{ij} la composition des involutions α_i et α_j :

$$\alpha_{ij}(b) = \alpha_j \circ \alpha_i(b) = \alpha_j(\alpha_i(b))$$

La notion de i-cellule peut être retrouvée dans une n-G-Carte par l'intermédiaire de la notion d'orbite. Celle-ci permet d'obtenir un ensemble de brins par composition d'un ensemble d'involutions.

Définition 2. *Soit $G = (B, \alpha_0, \alpha_1, \ldots, \alpha_n)$ une n-G-Carte et $b \in B$ un brin. On note $< \alpha_{i_0}, \ldots, \alpha_{i_k} > (b), 0 \leq i_0 < \cdots < i_k \leq n$ l'orbite de b par rapport à $\alpha_{i_0}, \ldots, \alpha_{i_k}$. Ceci correspond à l'ensemble des brins accessibles depuis b par une composition quelconque de ces applications.*

Une i-cellule correspond à une orbite contenant l'ensemble des involutions excepté α_i. Les i-cellules les plus courantes en dimension 3 sont :

0-cellule (le sommet) incidente au brin b est définie par l'orbite $< \alpha_1, \alpha_2, \alpha_3 > (b)$,

1-cellule (l'arête) incidente au brin b est définie par l'orbite $< \alpha_0, \alpha_2, \alpha_3 > (b)$,

2-cellule (la face) incidente au brin b est définie par l'orbite $< \alpha_0, \alpha_1, \alpha_3 > (b)$,

3-cellule (le volume) incidente au brin b est définie par l'orbite $< \alpha_0, \alpha_1, \alpha_2 > (b)$

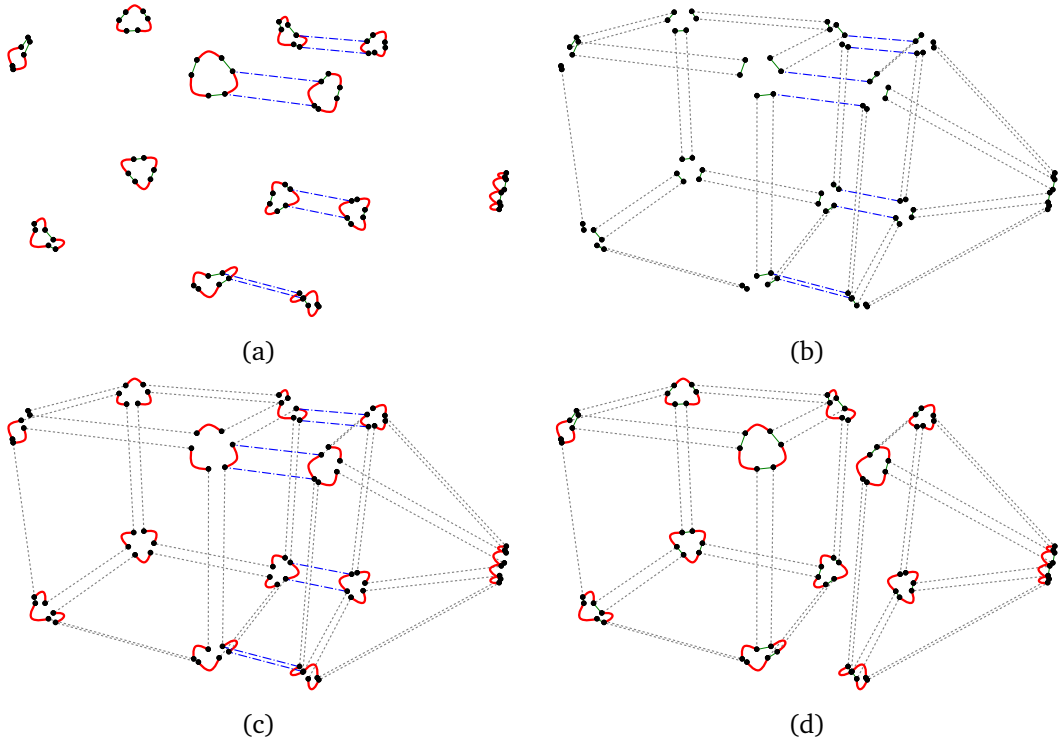


FIGURE 1.21 – Représentation des différentes orbites d'une 3-G-Carte : (a) sommet, (b) arête, (c) face et (d) volume.

Une n-G-Carte peut couvrir une partie ou la totalité de l'espace dans lequel celle-ci est définie. On parle alors respectivement de G-Carte *avec* ou *sans* bord. Les bords d'une n-G-Carte sont facilement identifiables en repérant les brins ayant un point fixe par α_n, c'est-à-dire que $\alpha_n(b) = b$. Seule cette involution est autorisée à avoir des points fixes. À l'inverse, une carte généralisée $G = (B, \alpha_0, \alpha_1, \ldots, \alpha_n)$ est une n-G-Carte sans bord si $\forall i \in [0, n]$ et $\forall b \in B$, $\alpha_i(b) \neq b$.

Une n-G-Carte peut être construite à partir de deux opérations de base. La première est l'ajout de brins qui sont invariants pour toutes les involutions. La seconde est la couture par α_i de deux orbites $< \alpha_0, \ldots, \alpha_{i-2}, \alpha_{i+2}, \ldots, \alpha_n >$. Par exemple, il est possible de créer une 3-G-Carte par assemblage de volumes (orbite $< \alpha_0, \alpha_1, \alpha_2 >$). Intuitivement, cela

consiste à assembler deux faces possédant la même structure, c'est-à-dire à coudre par α_3 deux orbites $< \alpha_0, \alpha_1 >$ isomorphes.

Définition 3. *Soit $G = (B, \alpha_0, \alpha_1, \ldots, \alpha_n)$ une n-G-Carte et $b, b' \in B$ deux brins tels que b et b' sont invariants par α_i. La couture par α_i de b et b' peut être réalisée si et seulement si $< \alpha_0, \ldots, \alpha_{i-2}, \alpha_{i+2}, \ldots, \alpha_n > (b)$ est isomorphe à $< \alpha_0, \ldots, \alpha_{i-2}, \alpha_{i+2}, \ldots, \alpha_n > (b')$ par une application ϕ. Le résultat de cette opération est donc une n-G-Carte $G' = (B, \alpha_0, \ldots, \alpha_{i-1}, \alpha'_i, \alpha_{i+1}, \ldots, \alpha_n)$ telle que :*

- $\forall b'' \in < \alpha_0, \ldots, \alpha_{i-2}, \alpha_{i+2}, \ldots, \alpha_n > (b)$ *et*
  $\forall b''' \in < \alpha_0, \ldots, \alpha_{i-2}, \alpha_{i+2}, \ldots, \alpha_n > (b')$,
  $b''\alpha'_i = b''\phi$ *et* $b'''\alpha'_i = b'''\phi^{-1}$,
- $\forall b'' \notin < \alpha_0, \ldots, \alpha_{i-2}, \alpha_{i+2}, \ldots, \alpha_n > (b) \cup < \alpha_0, \ldots, \alpha_{i-2}, \alpha_{i+2}, \ldots, \alpha_n > (b')$,
  $b''\alpha'_i = b''\alpha_i$.

Pour réaliser cette opération de couture, l'opération n'a besoin en paramètre que d'un brin de chaque orbite. Ainsi, les orbites des cellules à coudre sont parcourues en reliant chacun des brins qui les composent. Dans la suite du document, nous noterons `coudre`$_i$(*b,b'*) l'opération de couture par α_i des brin b et b'. À l'issue de cette opération, nous obtenons $\alpha_i(b) = b'$ et $\alpha_i(b') = b$. Nous noterons également `découdre`$_i(b)$ l'opération de décousure par α_i du brin b. À l'issue de cette opération le brin b est dit « libre » par α_i, nous obtenons $\alpha_i(b) = b$.

1.3.2 Plongements

Une G-Carte ne représente que la topologie des objets et non leur forme géométrique. Sans information géométrique, il n'est pas possible de représenter les brins dans l'espace. Pour cela, nous associons aux brins ou bien aux orbites des informations qui peuvent être de différentes natures (coordonnées cartésiennes, couleurs, données géologiques [PTM+11], etc.). Ces informations portent le nom de plongements et sont le plus souvent associées aux différentes cellules d'une G-Carte.

1.3.3 Opérations topologiques de base

Les modèles topologiques permettent de simplifier des opérations de base sur un maillage tout en conservant sa cohérence topologique. Nous détaillons ici certaines de ces opérations, que nous avons utilisées par la suite pour créer des opérations plus complexes (voir chapitre 3).

Insertion Commençons tout d'abord par l'opération la plus simple : l'insertion d'un sommet sur une arête. Pour cela, nous partons d'un Brin b de cette arête (Figures 1.22). Par défaut, le plongement du nouveau sommet correspond au milieu du segment formé par les deux extrémités de l'arête. La Figure 1.23 illustre l'algorithme 1.1 pour chaque brin appartenant à l'orbite $< \alpha_2, \alpha_3 >$ de b.

L'insertion d'une arête nécessite un brin de chaque orbite sommet que l'on souhaite relier. Nous supposons que les faces sont fermées. La Figure 1.24 montre un exemple d'insertion d'une arête sur une face appartenant à deux volumes.

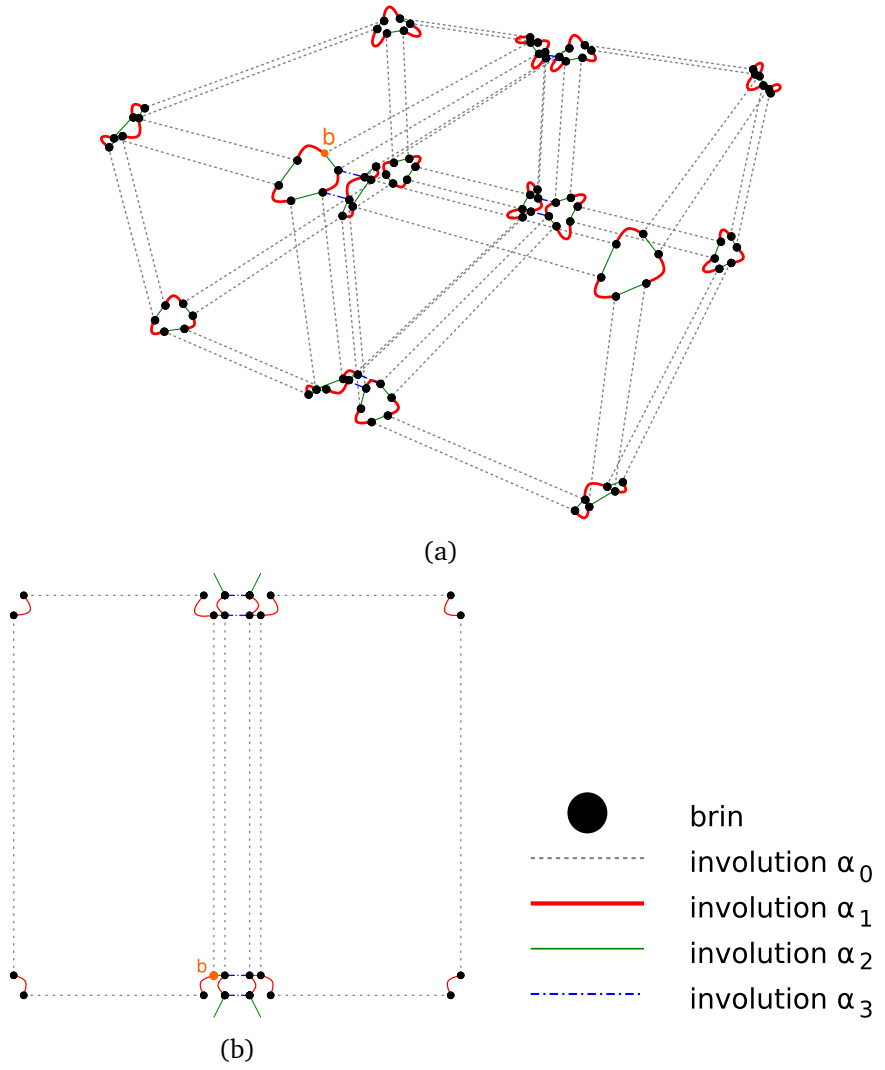
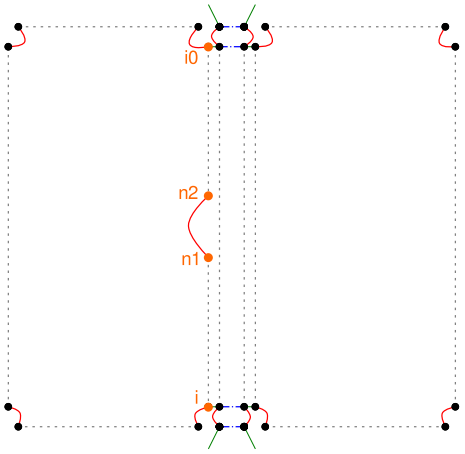
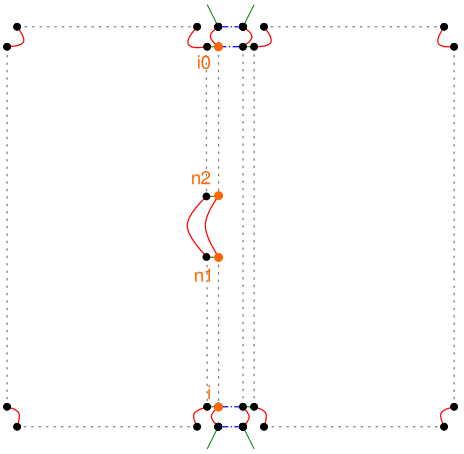


FIGURE 1.22 – Exemple d'insertion d'un sommet sur l'arête à laquelle appartient le brin b. L'arête appartient à deux volumes liés entre eux par α_3. (a) vue 3D, (b) vue de dessus.

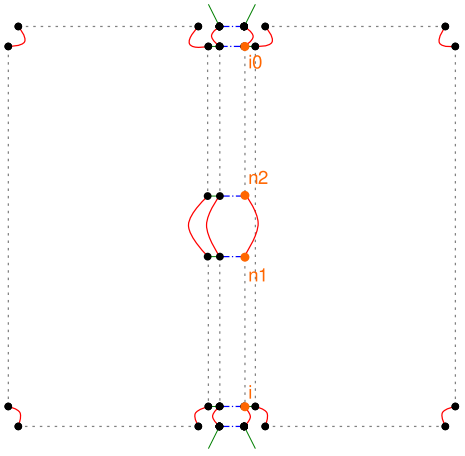
Dans le cas où les brins appartiennent à la même orbite face $< \alpha_0, \alpha_1, \alpha_3 >$, nous supposons également qu'ils ne sont pas déjà reliés par une arête. L'opération a pour conséquence de couper la face en deux.
Dans le cas où les brins n'appartiennent pas à la même orbite face, l'opération a pour conséquence de relier deux faces : l'une de ces faces représente alors un « trou » dans l'autre face.



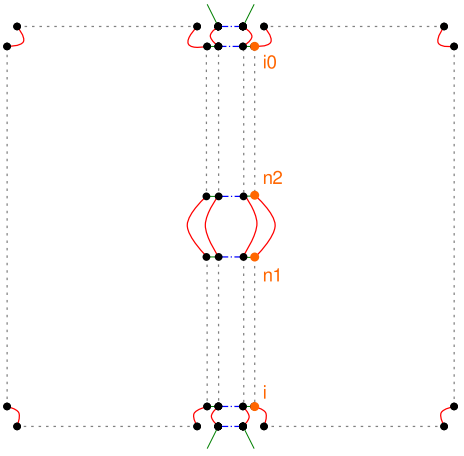
(a) Première itération : i vaut b. i n'est pas libre par α_2 mais $\alpha_2(i)$ n'a pas été traité.



(b) Deuxième itération : i vaut $\alpha_2(b)$. i n'est pas libre par α_2 et $\alpha_2(i)$ a été traité. On lie les brins adéquats par α_2. i n'est pas libre par α_3 mais $\alpha_3(i)$ n'a pas été traité.



(c) Troisième itération : i vaut $\alpha_{23}(b)$. i n'est pas libre par α_2 mais $\alpha_2(i)$ n'a pas été traité. i n'est pas libre par α_3 et $\alpha_3(i)$ a été traité. On lie les brins adéquats par α_3.



(d) Quatrième itération : i vaut $\alpha_{232}(b)$. i n'est pas libre par α_2 et $\alpha_2(i)$ a été traité. On lie les brins adéquats par α_2.

FIGURE 1.23 – Insertion d'un sommet sur une arête simple.

Algorithme 1.1 : `InsertionSommet(Brin b)`

Entrée : Un Brin *b* appartenant à l'arête.
Pour chaque *Brin i de l'orbite* $< \alpha_2, \alpha_3 > (b)$ **Faire**
  Brin i0 $\leftarrow \alpha_0(i)$;
  Brin n1, n2;
  `coudre`$_0$(*i, n1*);
  `coudre`$_0$(*i0, n2*);
  `coudre`$_1$(*n1, n2*);
  Si *i est libre par* α_2 *et* $\alpha_2(i)$ *a été traité* **Alors**
    `coudre`$_2$(*n1,* $\alpha_{20}(i)$);
    `coudre`$_2$(*n2,* $\alpha_{21}(n1)$);
  FinSi
  Si *i est libre par* α_3 *et* $\alpha_3(i)$ *a été traité* **Alors**
    `coudre`$_3$(*n1,* $\alpha_{30}(i)$);
    `coudre`$_3$(*n2,* $\alpha_{31}(n1)$);
  FinSi
  Marquer i comme traité;
FinPour
Démarquer tous les brins de l'orbite $< \alpha_2, \alpha_3 > (b)$;
Retourne $\alpha_0(b)$;

Algorithme 1.2 : `InsérerArête(Brin b1, Brin b2)`

Entrée : Un Brin *b1* appartenant à l'orbite du premier sommet.
Entrée : Un Brin *b2* appartenant à l'orbite du second sommet.
`// Voir Figure 1.24a`
Brin dd1 $\leftarrow \alpha_1$(b1);
Brin dd2 $\leftarrow \alpha_1$(b2);
Brin n1, n2;
`coudre`$_0$(*n1,n2*);
`// Voir Figure 1.24b`
`coudre`$_1$(*b1,n1*);
`coudre`$_1$(*b2,n2*);
`// Voir Figure 1.24c`
Brin nn1, nn2;
`coudre`$_0$(*nn1,nn2*);
`coudre`$_2$(*n1,nn1*);
`coudre`$_2$(*n2,nn2*);
`coudre`$_1$(*dd1,nn1*);
`coudre`$_1$(*dd2,nn2*);
Si *b1 n'est pas libre par* α_3 **Alors**
  `// Voir Figure 1.24d`
  Brin n31, n32;
  `coudre`$_0$(*n31,n32*);
  `coudre`$_3$(*n31,n1*);
  `coudre`$_3$(*n32,n2*);
  `coudre`$_1$($\alpha_3(b1)$,*n31*);
  `coudre`$_1$($\alpha_3(b2)$,*n32*);
  `// Voir Figure 1.24e`
  Brin nn31, nn32;
  `coudre`$_0$(*nn31,nn32*);
  `coudre`$_3$(*nn31,nn1*);
  `coudre`$_3$(*nn32,nn2*);
  `coudre`$_1$($\alpha_3(dd1)$,*nn31*);
  `coudre`$_1$($\alpha_3(dd2)$,*nn32*);
  `coudre`$_2$(*n31,nn31*);
  `coudre`$_2$(*n32,nn32*);
FinSi
Retourne $\alpha_1(b1)$

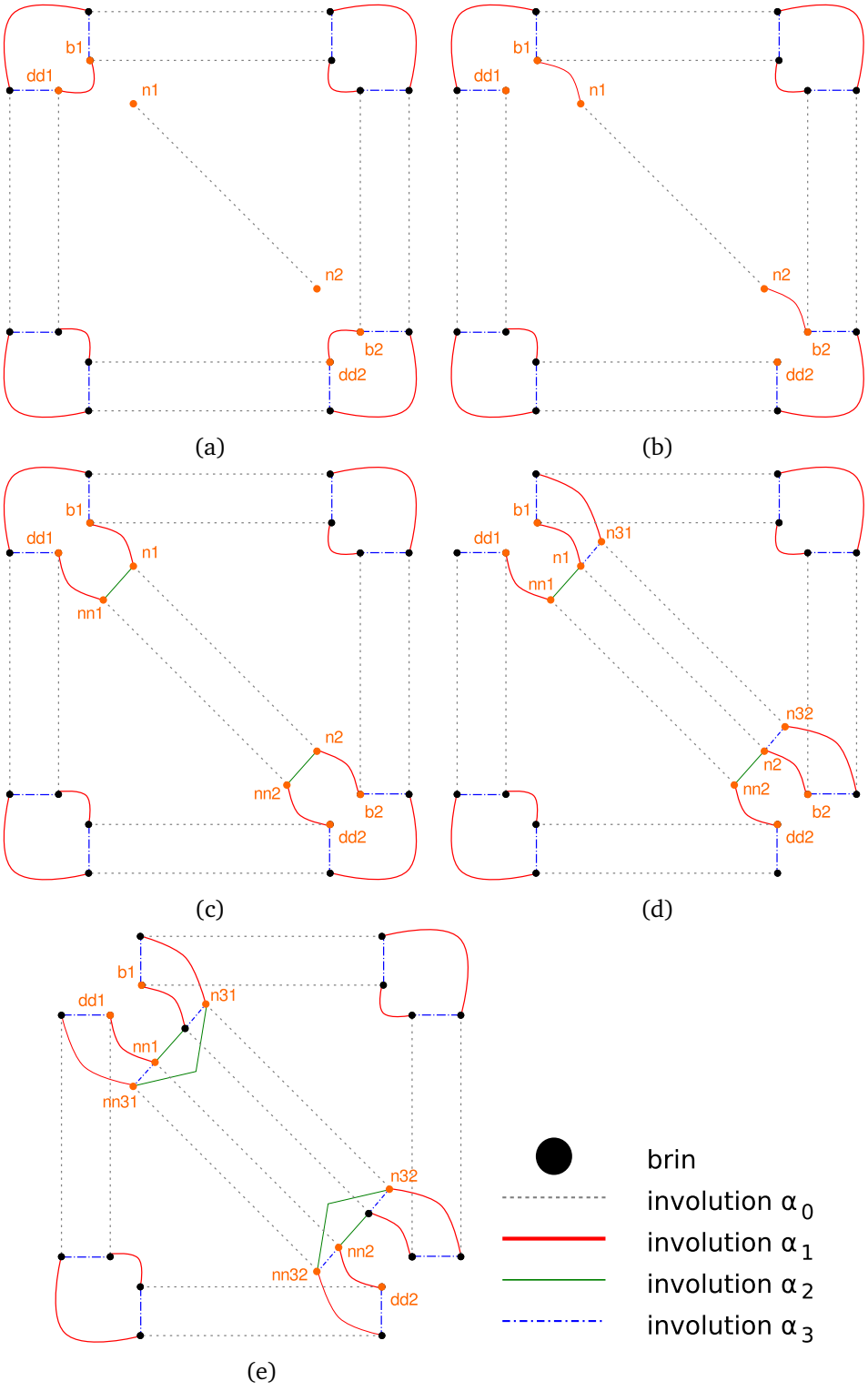


FIGURE 1.24 – Exemple d'insertion d'une arête entre deux sommets suivant l'algorithme 1.2.

Duplication La duplication permet de cloner un ensemble de brins ainsi que leurs liens. Un brin et l'orbite que l'on souhaite copier sont nécessaires à l'opération. Les plongements des sommets dupliqués sont égaux à leurs originaux par défaut.
Nous notons `dupliquer`$_{i_1 i_2 \ldots i_k}(b)$ l'opération de duplication de l'orbite $< \alpha_{i_1}, \alpha_{i_2}, \ldots, \alpha_{i_k} >$ du Brin b, avec $0 \leq i_1 \leq i_2 \leq \cdots \leq i_k \leq 3$, qui retourne le dupliqué de b. Par exemple, `dupliquer`$_{01}(b)$ duplique les brins de l'orbite $< \alpha_0, \alpha_1 > (b)$ en conservant les involutions α_0 et α_1 entre les brins dupliqués.

Contraction La contraction d'arête prend en paramètre un brin de l'arête que l'on souhaite contracter. Nous nous plaçons dans le cas où toutes les faces incidentes à l'arête sont fermées, ce qui simplifie les opérations en évitant plusieurs vérifications. La figure 1.25 illustre l'algorithme 1.3.

Algorithme 1.3 : `ContractArête(Brin b)`

Entrée : Un Brin b appartenant à l'arête à contracter.
Pour chaque *Brin i de l'orbite* $< \alpha_2, \alpha_3 > (b)$ **Faire**
  Brin d1 $\leftarrow \alpha_1$(i);
  Brin d2 $\leftarrow \alpha_{01}$(i);
  `// Voir Figure 1.25a`
  `découdre`$_1$(*d1*);
  `découdre`$_1$(*d2*);
  `coudre`$_1$(*d1,d2*);
  `// Voir Figure 1.25b`
FinPour
`// Voir Figure 1.25c`
Supprimer les brins de l'orbite $< \alpha_2, \alpha_3 > (b)$;
`// Voir Figure 1.25d`

Dans le cas d'une arête appartenant à une face triangulaire, le résultat de l'opération est une face dégénérée (Figure 1.26). Pour supprimer cette face, nous la contractons à l'aide de l'algorithme 1.4.

Triangulation de face Nous distinguons ici deux types de triangulations de face : sans insertion de sommet et avec insertion de sommet. Dans le premier cas, nous utilisons les méthodes de [FM84], [CI84], [CTVW88]. Ces triangulations se basent sur l'insertion d'arêtes entre les sommets composant la face (Figure 1.27b). Elles sont applicable aux faces convexes ou concaves, mais il faut ajouter des contraintes pour les faces « trouées ». La triangulation de Delaunay contrainte permet de trianguler de telles faces. Cette méthode maximise le plus petit angle de l'ensemble des angles des triangles, évitant les triangles trop allongés. Delaunay intègre pour cela une condition : le cercle circonscrit d'un triangle ne doit contenir aucun autre sommet que les siens. Dans le second cas, un sommet est inséré au sein de la face. Les triangles sont créés en utilisant l'opération de cône pour chaque arête composant la face d'origine (Figure 1.27c). Cette méthode est limitée car elle ne fonctionne pas dans le cas de face concave ou trouée (Figure 1.28), il faut alors insérer

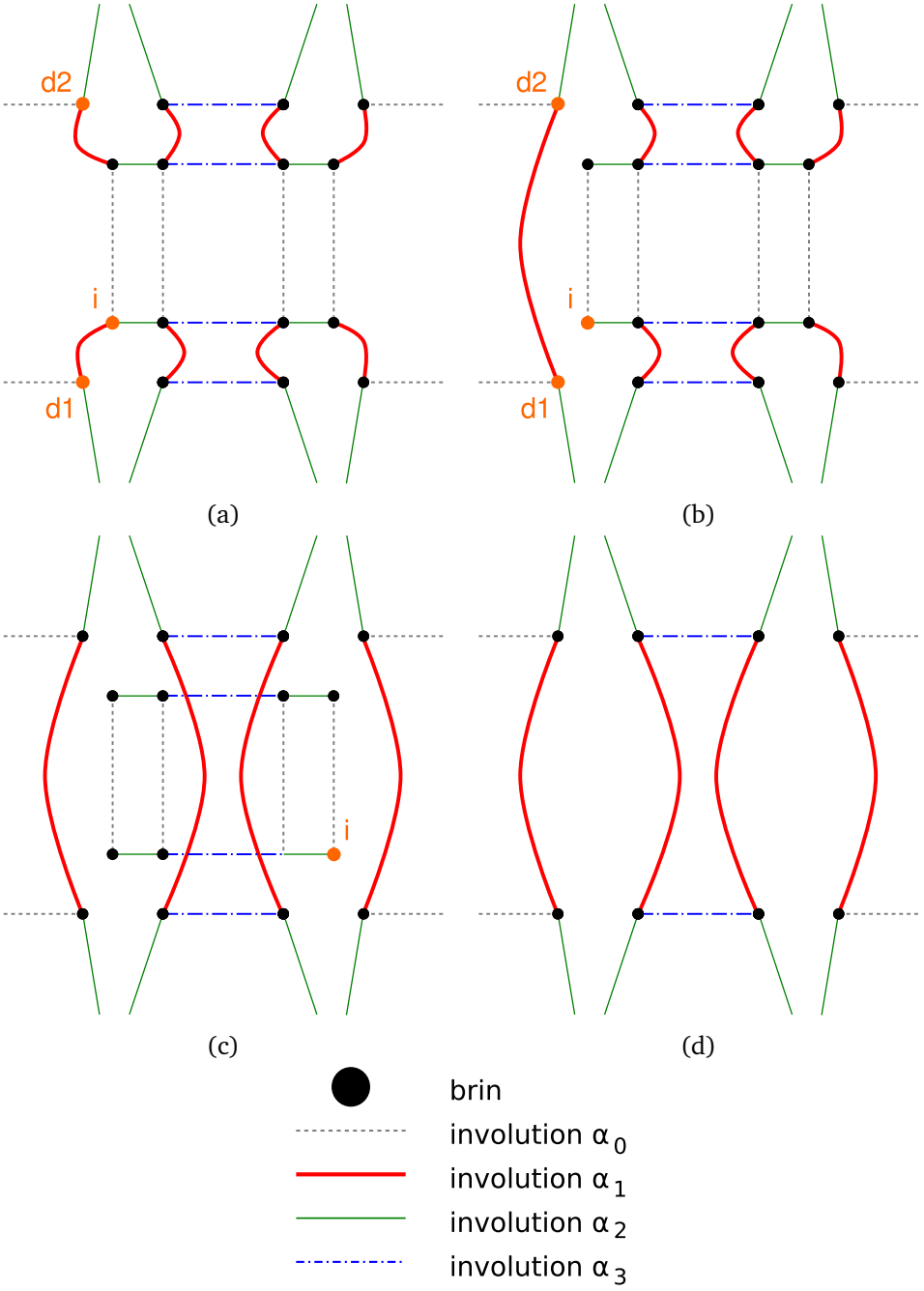


FIGURE 1.25 – Exemple de contraction d'une arête suivant l'algorithme 1.3.

Algorithme 1.4 : `ContractFaceDégénérée(Brin b)`

Entrée : Un Brin b appartenant à la face à contracter.

`// Voir Figure 1.26b`

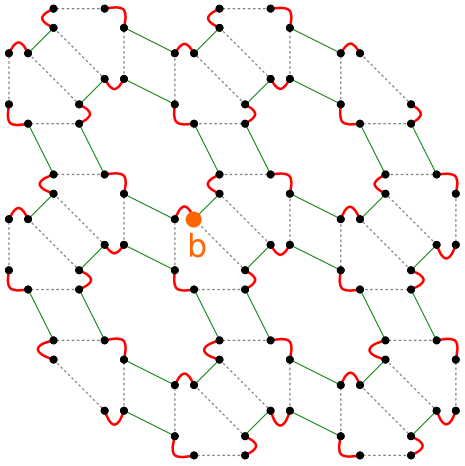
Pour chaque *Brin i de l'orbite* $< \alpha_2, \alpha_3 > (b)$ **Faire**

Brin d1 $\leftarrow \alpha_2$(i);
Brin d2 $\leftarrow \alpha_{12}$(i);
`découdre`$_2$(*d1*);
`découdre`$_2$(*d2*);
`coudre`$_2$(*d1,d2*);

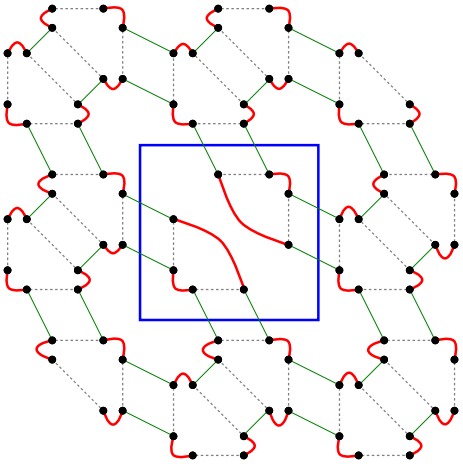
FinPour

Supprimer les brins de l'orbite $< \alpha_0, \alpha_1, \alpha_3 > (b)$;

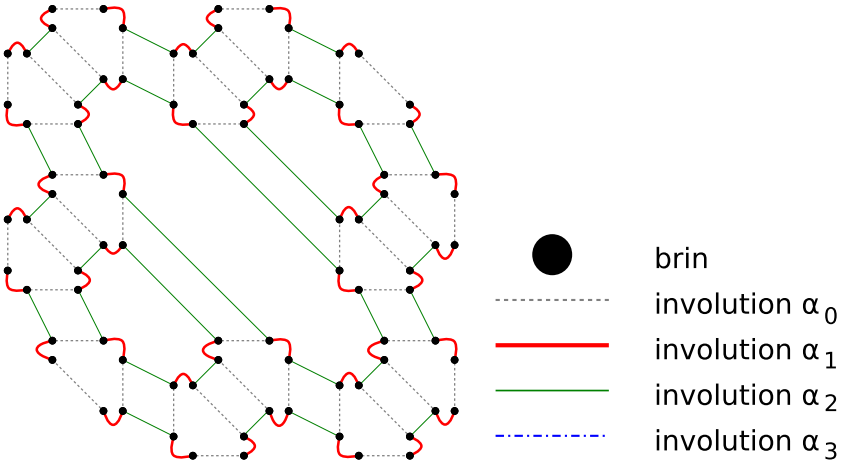
`// Voir Figure 1.26c`



(a) Contraction de l'arête incidente à b à l'aide de l'algorithme 1.3.



(b) Résultat de l'opération de contraction d'arête. Deux faces sont dégénérées (encadrées en bleu) : elles ne sont composées que de deux arêtes.



(c) Contraction des faces dégénérées ce qui a pour effet de les supprimer.

FIGURE 1.26 – Exemple de contraction d'arête menant à une face dégénérée.

plusieurs sommets ce qui complexifie le maillage.

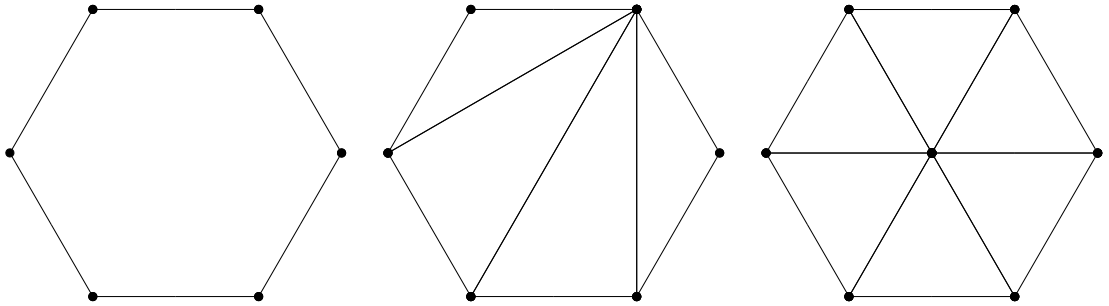


FIGURE 1.27 – Différentes triangulations d'une face convexe. (a) Face convexe à trianguler, (b) Triangulation sans insertion de sommet, (c) Triangulation avec insertion de sommet.

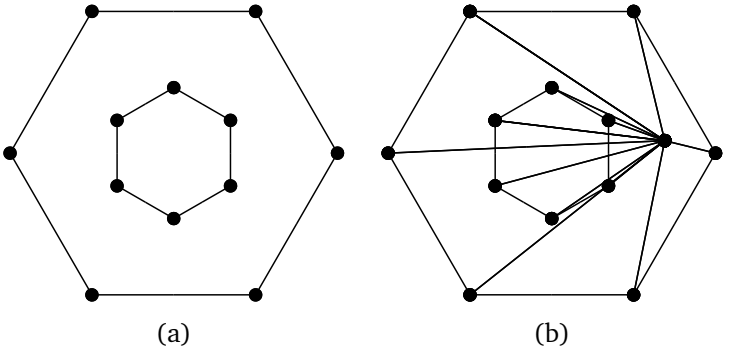


FIGURE 1.28 – Triangulation d'une face trouée. (a) Face trouvée à trianguler, (b) Triangulation avec insertion de sommet : le trou est comblé et les triangles générés s'intersectent.

Opération de cône pour une face L'opération de cône permet de créer une $(i+1)$-cellule à partir d'une i-cellule et d'un sommet S, tels que les sommets de la $(i+1)$-cellule construite regroupent les sommets de la i-cellule et le sommet S. Par exemple, le cône d'un sommet est une arête dont les bords sont le sommet original et S ; le cône d'une arête est une face triangulaire ; le cône d'une face est un volume dont les faces du bord regroupent la face originale et les faces qui résultent du cône des bords de la face originale, etc. L'algorithme 1.5 décrit l'opération de cône pour une face, il prend en paramètre un brin b appartenant à la face et les coordonnées du sommet S. Cette opération utilise les opérations de base de duplication et de triangulation de face avec création de sommet.

Fusion topologique La fusion de deux $i-$cellules peut se faire si elles sont liées par une $(i-1)$-cellule. L'opération consiste à supprimer cette $(i-1)$-cellule pour obtenir une nouvelle $i-$cellule.

1.3.4 Opérations booléennes par co-raffinement

Dans le cadre d'un terrain sur lequel s'écoule un fluide, on peut simplifier les modifications du terrain en deux étapes : le volume d'eau retire de la matière au terrain par

Algorithme 1.5 : `ConeFace(Brin b, Sommet S)`

Entrée : Un Brin b appartenant à i-cellule.
Entrée : Un Sommet S.
`// On duplique la face incidente à` b.
Brin d $\leftarrow$ `dupliquer`$_{013}(b)$;
`// On coud par` α_2 `les deux faces entre elles.`
Pour chaque *Brins i de l'orbite* $< \alpha_0, \alpha_1, \alpha_3 > (b)$ ***et*** *j de l'orbite* $< \alpha_0, \alpha_1, \alpha_3 > (d)$ *correspondant* **Faire**
  `coudre`$_2(i,j)$;
FinPour
`// On triangule la face incidente à` d `avec insertion de sommet. Le sommet inséré est` S.
`triangulationAvecSommet`(d, S);

endroit, et rajoute de la matière à d'autres. On peut également facilement imaginer ces deux étapes basées sur l'opération booléenne de différence. Prenons l'exemple d'un terrain représenté par un ensemble de volumes et la rivière représentée par un volume. Lors de l'érosion, les sommets du volume d'eau sont déplacés puis on effectue une différence entre le volume d'eau et l'ensemble des volumes représentant le terrain. Le résultat est celui escompté : le terrain est bien érodé.

Les opérations booléennes sont à la base de la géométrie de construction de solides (CSG). Le mécanisme permettant d'obtenir le résultat d'une opération booléenne se décompose en plusieurs phases. Tout d'abord, il faut détecter les cellules de chaque objet qui sont en intersection puis les découper mutuellement afin de modéliser leur intersection. Il est ensuite nécessaire de déterminer, parmi les découpes obtenues, lesquelles appartiennent à l'objet final en fonction de l'opération booléenne choisie, de supprimer les parties d'objets inutiles et de recoller celles restantes afin d'obtenir le bon résultat. Les algorithmes [CD96] [Caz97] s'appuient sur une opération unique appelée co-raffinement permettant de calculer les intersections entre deux subdivisions de l'espace afin d'en obtenir une nouvelle formée des deux précédentes. Le résultat de cette opération est constitué de différentes régions correspondant aux résultats obtenus par les opérations booléennes. Le co-raffinement ne se limite pas au traitement d'objets pleins : il peut gérer n'importe quel type de subdivision comme par exemple des maillages (Figures 1.29).

Cette opération a été appliquée aux cartes combinatoires [CD97] et aux G-Cartes [Gui06] dans le but de construire des modèles géologiques 3D.

L'opération booléenne ne prend pas en compte les propriétés géologiques des volumes, il faut donc tenir compte de la résistance à l'érosion d'une couche de terrain lorsque l'on déplace les sommets du volume d'eau. Lors de la sédimentation, les sommets du terrain sont déplacés afin de faire gagner de la matière aux volumes. On effectue ensuite une différence entre l'ensemble des volumes du terrain et le volume d'eau pour que la rivière *épouse* correctement le terrain. En théorie, cette méthode fonctionne correctement, mais les déplacements des sommets doivent être conséquents. En effet, l'opération booléenne

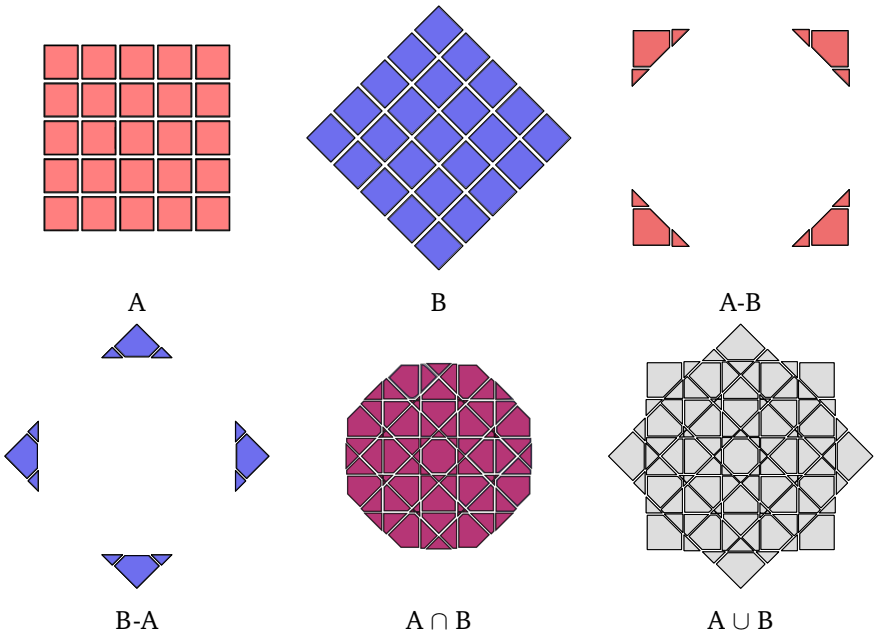


FIGURE 1.29 – Résultats des différentes opérations booléennes à partir du co-raffinement de deux maillages A et B (extrait de [Gui06]).

calcule des intersections entre les faces du terrain et du fluide, or si les déplacements des sommets sont trop faibles, les faces sont détectées comme coplanaires et l'opération booléenne échoue. À l'inverse, si les déplacements de sommets sont trop conséquents, l'écoulement du fluide est modifié de manière trop brutale et crée des incohérences.

1.3.5 Implantations

Les cartes combinatoires et cartes généralisées sont basées sur une structure de brins simples. Des modeleurs utilisent ces structures et implantent des opérations de base, dont celles décrites précédemment, et d'autres plus complexes. CGoGN [1], Combinatorial and Geometric modeling with Generic N-dimensional Maps, a été développé au laboratoire ICUBE de l'Université de Strasbourg. Il s'agit d'un moteur utilisant les cartes combinatoires, basé sur l'indexation des brins. Le moteur a atteint sa maturité récemment, c'est l'une des raisons pour laquelle nous avons privilégié un autre moteur au début de la thèse en 2009 : Moka. Moka [2] a été développé par l'équipe XLIM-SIC de l'Université de Poitiers. Ce moteur utilise les 3-G-cartes, est orienté modélisation et possède une interface graphique en Qt/C++ permettant de créer des objets 3D à l'aide des opérations de base. Moka est sous licence GPL, il est possible d'y ajouter de nouvelles opérations. Il est à noter qu'aujourd'hui, une implantation de CGOGN gère les cartes généralisées. Notre code est donc facilement portable sur ce support.

1. http://cgogn.u-strasbg.fr/
2. http://moka-modeller.sourceforge.net/

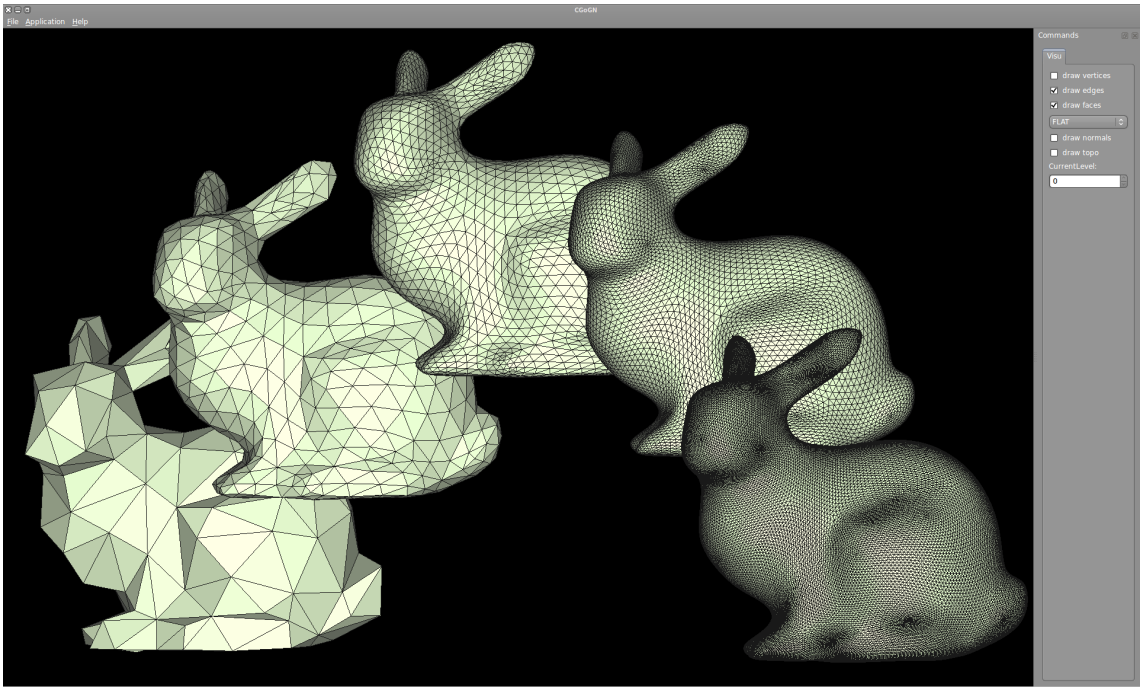


FIGURE 1.30 – Impression écran de l'actuelle interface graphique de CGoGN.

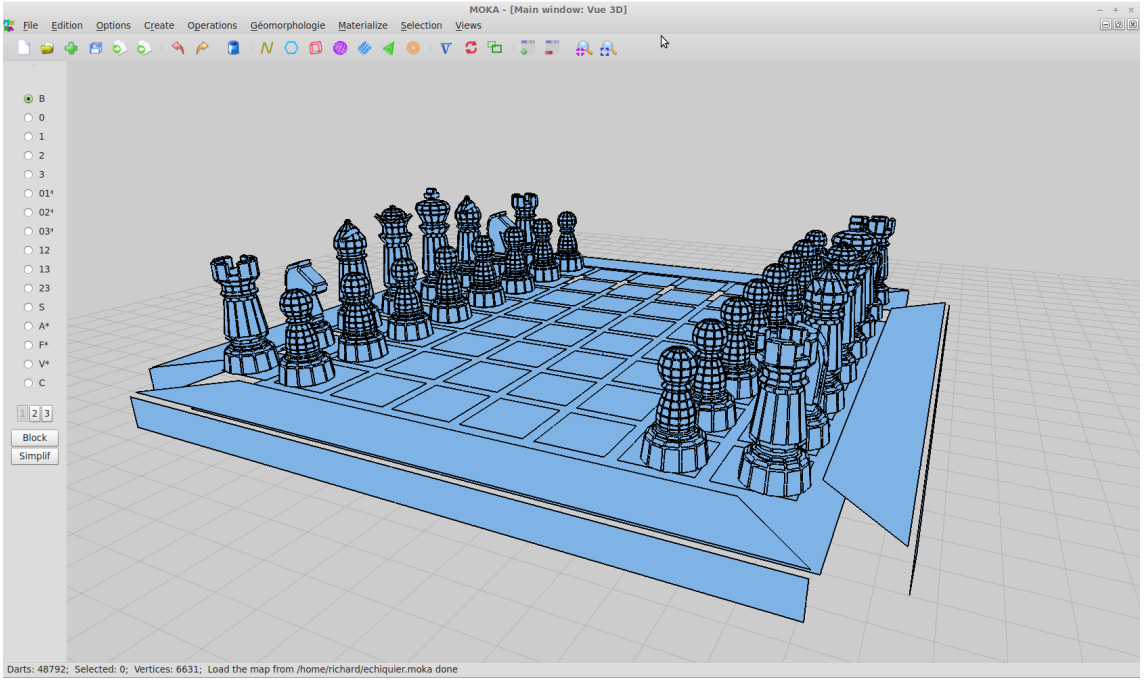


FIGURE 1.31 – Impression écran de l'actuelle interface graphique de Moka.

1.4 Synthèse

Notre objectif est de simuler l'évolution d'un terrain subissant des modifications dues à l'écoulement d'un fluide. Nous avons différencié la construction du terrain et l'évolution du terrain. Les méthodes de construction surfaciques stochastiques ne prennent pas en compte la structure en couches du terrain et ne permettent pas d'effectuer des modifications prenant en compte les caractéristiques géologiques des couches. Nous avons également constaté que les méthodes de construction utilisant les voxels et de strates restent limitées à la résolution et à la taille d'une grille. Il est également nécessaire d'ajouter une phase intermédiaire pour effectuer le rendu. Les méthodes volumiques basées sur un modèle topologique permettent de créer un terrain en couches mais pas de le faire évoluer en 3D au cours du temps. Les méthodes inspirées de la physique simulent la détérioration du terrain par un écoulement de fluide. Les systèmes de particules permettent de simuler un plus large éventail de phénomènes hydrologiques que les modèles basés sur des voxels. Ils ne sont également pas limités à la résolution et la taille d'une grille.

Seuls quelques modèles que nous avons détaillés permettent à la fois de représenter la structure du terrain et de le faire évoluer au cours du temps à l'aide d'une simulation de fluide. Mais les modifications effectuées sont contraintes afin d'éviter les problèmes de cohérence de la structure de terrain.

Nous avons également décrit le modèle des n-G-cartes que nous utilisons par la suite pour créer le terrain en couches. Nous avons critiqué une approche basée sur les opérations booléennes pour effectuer les modifications du terrain.

Dans les chapitres suivants, nous étudions l'interaction entre un fluide et le terrain ainsi que l'ensemble des opérations topologiques permettant de résoudre la problématique posée. Nous proposons dans le chapitre 2, un modèle d'interaction entre un fluide basé sur un système de particules et la surface d'un terrain représentée par un maillage. Nous critiquons ce modèle et tentons de l'améliorer. Dans le chapitre 3, nous décrivons les opérations topologiques adaptées à un modèle de terrain volumique en couches permettant de résoudre les incohérences décrites dans la partie 1.1.3.

Chapitre 2

Modèle interactif d’érosion et de sédimentation

Sommaire

Chapitre 2

Modèle interactif d'érosion et de sédimentation

Introduction

Notre objectif est de créer un modèle de simulation physique de fluide permettant de transporter de la matière afin de pouvoir déterminer la localisation et l'intensité des modifications à effectuer au niveau du terrain. La modification du terrain par écoulement de fluide est issue de trois phénomènes : l'érosion, le transport et le dépôt de sédiments. Ils peuvent être de différentes natures : mécanique ou chimique. Par exemple, le calcaire est érodé, transporté par dissolution dans l'eau puis déposé par précipitation : il s'agit d'une interaction chimique. Un grain de sable est érodé puis transporté par saltation ou par charriage : il s'agit ici d'une interaction mécanique.
Ces phénomènes résultent de l'interaction, chimique ou mécanique, entre le fluide et le terrain et doivent être simulés afin d'obtenir une évolution de terrain cohérente et réaliste. Nous l'avons vu dans l'état de l'art, les méthodes inspirées de la physique permettent non seulement d'améliorer le réalisme du terrain, mais aussi de le faire évoluer au cours du temps. La méthode s'approchant le plus de notre objectif, décrite par Krištof *et al.* [KBKv09], est assimilée à une détérioration chimique du terrain : les sédiments dissous dans le fluide sont diffusés. Ils ne prennent pas en compte l'érosion mécanique due à l'écoulement du fluide.
L'interaction entre le fluide et le solide est déterminante pour notre modèle. Nous ne prenons pas en compte l'interaction entre le fluide qui s'infiltre dans le terrain mais seulement l'interaction du fluide qui s'écoule sur le terrain. Ainsi, nous avons besoin d'une représentation surfacique du terrain. Ensuite, nous souhaitons que notre modèle puisse représenter un terrain de taille et de complexité quelconques et que le fluide puisse s'écouler sur la totalité du terrain si nécessaire. Nous souhaitons également obtenir des simulations en temps interactif, notre modèle doit pour cela être parallélisable. Comme nous l'avons montré dans la section 1.2.1, les systèmes de particules permettent de simuler un fluide sans être limité à la taille et la résolution d'une grille. De plus, les systèmes de particules sont parallélisables et permettent d'obtenir des simulations en temps interactifs avec un grand nombre de particules sur les cartes graphiques actuelles (plusieurs centaines de milliers

de particules).

Les résultats présentés dans ce chapitre ont fait l'objet d'une présentation à International Conference on Computer Vision and Graphics [BPC+10] et d'une publication dans la revue internationale Machine GRAPHICS & VISION [BPC+11].

2.1 Approche mécanique

Cette première approche reproduit l'impact physique d'un volume de fluide sur un terrain en simulant la saltation des sédiments sur le terrain. Nous utilisons un modèle particulaire de fluide à base de DEM (Discrete Element Method, voir section 1.2.1.1) et un terrain représenté par un maillage triangulaire raffiné selon un critère d'élévation et de pente. Les particules de fluide qui entrent en collision avec un triangle du terrain prélèvent une quantité de matière qu'elles vont transporter et déposer en aval.

Notre modèle se basant sur l'écoulement du fluide sur un terrain, nous devons gérer les collisions entre les particules en mouvement et les triangles composant la surface du terrain. Afin de simplifier le problème, nous considérons lors de la détection de collisions que seules les particules de fluide sont en mouvement. En effet, une méthode de résolution considérant à la fois les mouvements des particules et les mouvements des triangles est consommatrice en temps de calcul. Le modèle doit également gérer le transport de sédiments ainsi que leur dépôt. L'influence des particules de fluide isolées doit être minimisée tout en permettant au fluide de s'écouler de manière libre. Enfin, le terrain doit être modifié en fonction de l'érosion et de la sédimentation qu'il subit et ce, sans provoquer d'incohérence dans l'écoulement du fluide.

Nous avons établi l'algorithme suivant pour respecter ces contraintes :

- Mise à jour de la position des particules de fluide
- Calcul des collisions avec le maillage
- Évaporation et absorption de particules de fluide
- Génération, transport et dépôt des sédiments
- Mise à jour de la position des sommets du maillage

Le déplacement des particules de fluide et les calculs de collision s'effectuent à chaque pas de temps, alors que l'érosion et la sédimentation représentées par le déplacement des sommets du maillage sont effectuées seulement à certains pas de temps. Ce délai permet aux déplacements dus à l'érosion et à la sédimentation d'être suffisamment conséquents pour avoir un intérêt pour la simulation. En effet, déplacer ces sommets à chaque itération impliquerait de recalculer la grille les stockant également, ce qui serait coûteux en temps de calcul. L'évaporation et l'absorption s'appliquent seulement aux particules répondant à certains critères que nous décrivons dans la suite.

L'ensemble de ces étapes est détaillé dans les sections suivantes.

2.1.1 Modèle d'érosion hydraulique

Le modèle de fluide que nous avons utilisé est basé sur la simulation gravitationnelle de n-corps fournie dans le kit de développement de la technologie GPGPU CUDA [Ngu08]. Notre méthode propose un modèle d'interaction entre des particules et des triangles, nous avons donc choisi d'utiliser un système de particules simple basé sur le DEM. En effet, les interactions entre les particules prennent en compte les forces de friction, d'attraction et de cisaillement entre les particules, ce qui permet de reproduire un comportement qui reproduit assez convenablement un ensemble de molécules d'eau.

2.1.1.1 Érosion

Afin d'obtenir des interactions réalistes entre le fluide et le terrain, nous devons d'abord détecter et gérer les collisions entre les particules et les triangles du terrain. Notre méthode est inspirée de [Bad90] ; nous calculons les intersections entre les triangles et les segments formés par les positions successives p_t et p_{t+1} des particules de fluide. La méthode de calcul d'intersection entre un segment et un plan est connue. On utilise l'équation du plan et l'équation d'une droite dans l'espace pour obtenir l'équation suivante :

$$t = -\frac{d + \vec{n} \cdot O}{\vec{n} \cdot \vec{D}} \tag{2.1}$$

où $\vec{n}$ est la normale du plan, O un point sur la droite, $\vec{D}$ le vecteur directeur normalisé de la droite et d, la distance du plan à l'origine, obtenue par le produit scalaire d'un point du plan et de sa normale. Si $t \geq 0$, il y a intersection entre la droite et le plan. Le point d'intersection P est calculé en utilisant l'équation paramétrique de la droite. Dans le cas d'un triangle, la position du point P est calculée ainsi (Figure 2.1b) :

$$\overrightarrow{V_0P} = \alpha\overrightarrow{V_0V_1} + \beta\overrightarrow{V_0V_2} \tag{2.2}$$

où α et β sont deux réels. Le point P est dans le triangle $(\triangle V_0V_1V_2)$ si $\alpha \geq 0, \beta \geq 0$, et $\alpha + \beta \leq 1$.

La vélocité $\vec{v}$ de la particule est décomposée en deux vecteurs $\vec{v_t}$ et $\vec{v_n}$, représentant respectivement la vélocité tangentielle et la vélocité normale relatives au vecteur normal $\vec{n}$. Après collision, la vélocité normale $\vec{v'}_n$ est atténuée pour simuler la force de friction, dépendante du type de sol. Un second paramètre d'amortissement, spécifié par l'utilisateur, est défini par un scalaire compris entre 0 et 1. La nouvelle vélocité $\vec{v}_{t+1}$ de la particule à l'instant $t+1$ est obtenue en faisant la somme de la vélocité tangentielle $\vec{v_t}$ et la vélocité normale *atténuée* $\vec{v'}_n$. Ce système force les particules à « glisser » sur la surface comme illustré sur la figure 2.2a.
Quand elle entre en collision avec un triangle, une particule érode une certaine quantité de matière C (en reprenant [NWD05]) :

$$C = K_c \cdot sin\alpha \cdot |\vec{v}| \tag{2.3}$$

où K_c est le taux d'érosion et α est l'angle d'inclinaison du triangle. La prise en compte de l'inclinaison de la face nous permet d'effectuer une érosion en fonction de la pente. Cela

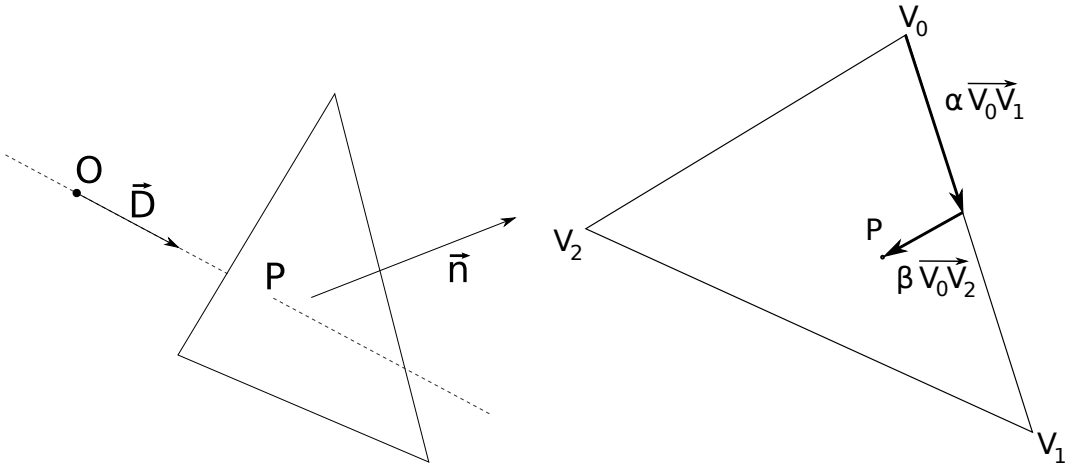


FIGURE 2.1 – (a) Calcul de l'intersection entre une droite et un triangle (Équation 2.1). (b) Représentation paramétrique du point P (Équation 2.2).

signifie qu'un sol horizontal sera moins érodé qu'un sol pentu. Afin d'appliquer l'érosion, nous déplaçons les sommets du triangle vers le bas. Si les trois sommets sont déplacés uniformément d'une même valeur $C/3$, la pente est conservée mais le résultat n'est pas réaliste (Figure 2.2b) : l'érosion d'un terrain doit avoir pour effet d'aplanir celui-ci. Nous avons donc choisi d'appliquer un déplacement pondéré en fonction de la hauteur des sommets : $2/3$ pour le sommet le plus haut, $1/6$ pour les deux autres. La pente du triangle est adoucie, comme montré sur la figure 2.2c. Le déplacement final appliqué à un sommet est la somme des déplacements calculés pour chaque triangle auquel il appartient.

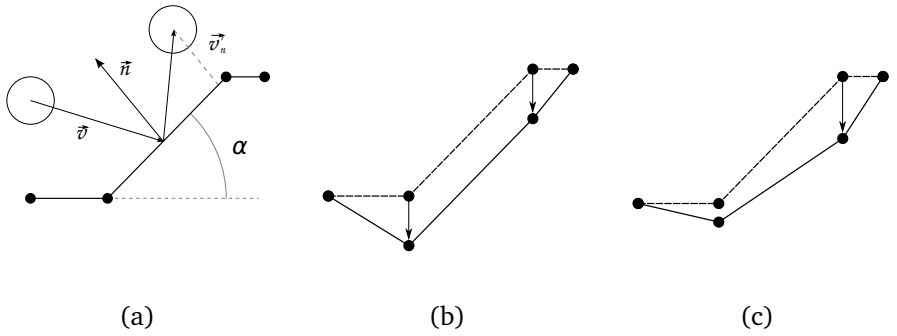


FIGURE 2.2 – (a) Collision entre une particule et un triangle (représenté par un segment 2D). (b). Déplacement vertical uniforme des sommets. (c). Déplacement vertical pondéré par la hauteur.

2.1.1.2 Transport et sédimentation

Nous dérivons la quantité s de sédiments érodée depuis un sommet donné à partir du volume « perdu » par érosion comme montré sur la figure 2.3. Le fluide transporte ces sédiments et finalement les dépose au bout d'un certain temps dont nous détaillons le calcul ci-dessous.

Le processus de transport/dépôt est complexe : plusieurs paramètres sont à prendre en

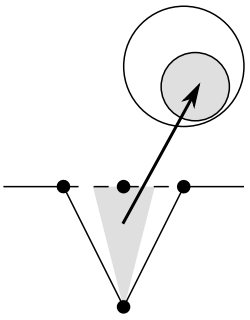


FIGURE 2.3 – Volume érodé depuis un sommet (en gris) et transféré à la particule de fluide la plus proche.

compte comme la taille des sédiments, leur composition et la vitesse du fluide.

Une méthode de transport de sédiments possible est d'ajouter un nouveau type de particules à la simulation qui gère les sédiments : lors de l'érosion, des sédiments sont générés et transportés par les particules de fluide avec lesquelles elles interagissent. Il est alors possible d'obtenir le déplacement de chaque sédiment et de déterminer si le sédiment est déposé sur un triangle en fonction de sa vitesse. Les interactions entre particules de fluide et de sédiment doivent être pondérées par leur masse : une particule de fluide n'a pas la même force d'attraction ou de répulsion qu'un sédiment. Cette méthode complexifie la simulation puisqu'elle implique de gérer plus de types d'interactions. Nous avons donc simplifié le problème en attachant les sédiments directement aux particules de fluide (Figure 2.3) : cela nous permet de gérer un type d'élément seulement. Lors de la phase d'érosion, nous ajoutons une quantité de sédiments à la particule de fluide la plus proche du sommet érodé.

Ensuite, nous estimons le *temps de transport* ε d'un sédiment de manière empirique, exprimé en nombre de pas de temps et dépendant de la taille s du sédiment :

$$\varepsilon = \lceil \varepsilon_M \left[1 - \left(\frac{s}{s_M} \right)^{\frac{1}{s_M}} \right] \rceil \tag{2.4}$$

où ε_M représente le temps maximal de transport et s_M la taille maximale d'un sédiment.

À chaque pas de la simulation, le temps de transport ε de chaque particule est décrémenté : quand il atteint zéro, le sédiment tombe au fond du lit de la rivière. Comme représenté sur la figure 2.4, l'équation 2.4 assure que les sédiments d'une taille proche de zéro sont transportés pendant au maximum ε_M pas de temps avant d'être déposés, alors que des sédiments d'une taille plus importante, proche de s_M, sont déplacés pendant seulement quelques pas. Lorsque la taille du sédiment est égale à s_M, ε est égale à 0. Le sédiment n'est alors pas transporté. Le seuillage permet d'obtenir un pas de temps entier. Si le fluide ralentit, les sédiments sont alors transportés sur une plus courte distance. Notre méthode simule ainsi le phénomène de saltation : les sédiments se déplacent par sauts successifs tant que la vitesse du fluide le permet.

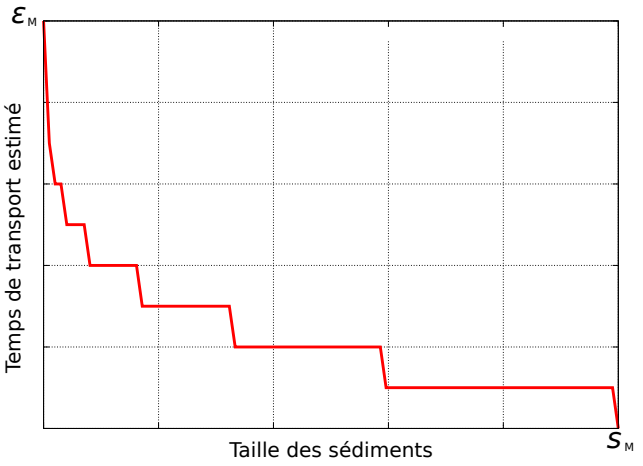


FIGURE 2.4 – Temps de transport estimé d'un sédiment en fonction de sa taille d'après l'équation 2.4.

Dans notre modèle, la sédimentation consiste à transférer les sédiments depuis les particules de fluide vers les triangles quand leur temps de transport atteint zéro. Nous posons que le volume d'un sédiment n'est distribué qu'à un seul triangle afin de simplifier les calculs. Les sommets du triangle sont cette fois déplacés vers le haut, en utilisant, ici aussi, différents « poids » (une règle similaire est utilisée dans [KBKv09]). Les poids associés sont de $2/3$ pour le sommet le plus bas et $1/6$ pour les deux autres sommets.

2.1.1.3 Évaporation et absorption

L'évaporation et l'absorption du fluide par le terrain sont deux phénomènes physiques à prendre en compte pour une simulation réaliste. Ces deux processus permettent également de retirer des particules de fluide de la simulation dans certaines conditions, comme par exemple quand un groupe de particules isolées forme une flaque d'eau. Après un certain nombre de pas de temps, les particules isolées peuvent être « évaporées » (i.e. désactivées) car elles n'érodent plus le terrain.

Une particule de fluide est considérée comme isolée si le nombre de ses particules voisines descend en dessous d'un certain seuil pendant une durée définie par l'utilisateur. Nous prenons également en compte une vélocité minimale v_{min} afin de supprimer les particules de fluide stagnantes.

Pour simuler l'absorption du fluide par le terrain, nous avons intégré un compteur au niveau des triangles : les n premières particules qui entrent en collision avec un triangle sont « absorbées » (i.e. désactivées). Pour qu'un ensemble de triangles forme un nouveau lit de rivière par érosions successives, le fluide doit « fournir » un nombre de particules supérieur au seuil *minCollisions* : cela évite un élargissement du lit trop brutal ainsi qu'une ramification du lit trop importante qui donnerait naissance à une multitude de bras de rivière.

2.1.2 Implantation

Notre méthode est en temps interactif. Pour atteindre cette performance, nous avons choisi d'utiliser la puissance des cartes graphiques en utilisant la technologie GPGPU développée par Nvidia : CUDA (Compute Unified Device Architecture). L'architecture de ces cartes graphiques est de type SIMD [1].

Alors que la puissance des processeurs n'augmente pas autant qu'on pouvait l'espérer d'après les conjonctures de Moore [Moo65], la tendance est à multiplier le nombre de cœurs disponibles pour effectuer des opérations en parallèle. On peut constater d'après la figure 2.5 que les processeurs graphiques offrent une plus grande puissance de calcul que les processeurs standard. Cela permet de simuler le comportement d'un plus grand nombre de particules et ainsi obtenir des résultats plus plausibles. Les cartes graphiques ont, elles, une architecture spécifiquement conçue pour effectuer des calculs en parallèle. Nvidia a créé la technologie CUDA permettant d'utiliser cette capacité de manière plus intuitive sur ses cartes graphiques. Une autre technologie, OpenCL [Std], basée sur l'expérience de CUDA, est utilisable aussi bien sur les cartes graphiques de Nvidia que de AMD-ATI. Elle est moins mature et plus restreinte que CUDA, par exemple au niveau des opérations atomiques qui limitent les accès concurrents à la mémoire. La programmation se fait en C sous forme de noyaux, plus communément appelés `kernels`, qui sont exécutés en parallèle et ont accès à plusieurs types de mémoires.

Les systèmes de particules sont bien adaptés à la parallélisation. En effet, les données concernant une particule à l'instant t sont relatives aux données de ses voisines à l'instant $t-1$. On stocke ces informations à l'instant t et à l'instant $t-1$ pour calculer correctement les forces appliquées aux particules. Chaque particule est gérée par un `thread`. Un `thread` est un processus léger qui exécute un `kernel`. Les cartes graphiques lancent plusieurs `threads` simultanément, exécutant le même `kernel` avec des données d'entrées différentes. Des structures de recherche, comme une grille de hachage, peuvent être utilisées afin d'accélérer les calculs.

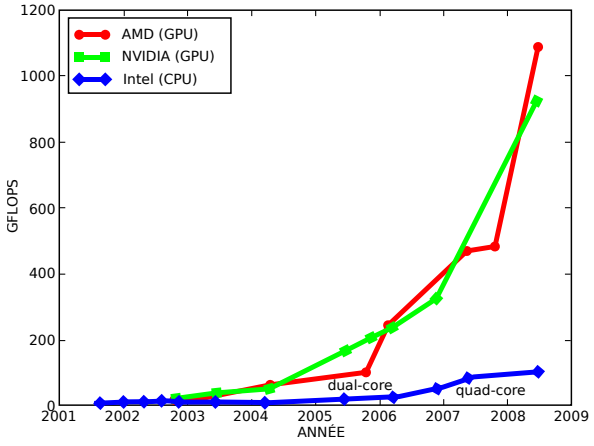


FIGURE 2.5 – Évolution des puissances de calcul en GFlops des processeurs AMD-ATI, Nvidia et Intel de 2001 à 2009. (extrait de http://air.imag.fr/mediawiki/index.php/GPGPU)

1. Single Instruction on Multiple Data, ou SIMD, est une architecture où la même instruction est appliquée simultanéement à plusieurs données pour produire plusieurs résultats.

2.1.2.1 Structures de données

Une particule possède une position dans l'espace et une vélocité. Ces deux attributs sont stockés dans des tableaux séparés qui sont transmis à la carte graphique. L'intérêt de séparer les données est de permettre à un `kernel` d'utiliser uniquement les informations nécessaires à son calcul (par exemple déterminer la position dans une grille régulière - *cf.* ci-dessous) dès qu'elles sont disponibles. Nous stockons également la position des particules à l'instant $t-1$, ainsi que la somme des forces appliquées à chaque particule.

Les particules sont placées dans une grille régulière de hachage pour accélérer la recherche des voisines. Nous calculons la clé de hachage d'une particule à partir de sa position dans la grille et nous créons un couple clé-index pour chaque particule. Nous trions alors les particules selon leur clé, ce qui revient à les trier selon leur position, permettant ainsi d'accéder directement à leurs voisines.

Le calcul de la clé décorrèle la résolution de la grille de hachage des dimensions de la simulation. En effet, le calcul de la position d'une particule dans la grille de hachage est modulo la taille de la grille. Deux particules peuvent avoir la même position dans la grille de hachage, mais ne pas être voisines. Il faut donc, lors des calculs d'interaction entre les particules, vérifier que la distance entre celles-ci est bien inférieure au rayon d'interaction. En conséquence, la résolution de la grille et la taille de cellule influent sur les performances de la méthode. Plus la résolution est élevée et la taille de cellule est petite, moins il y a de particules par cellule mais le coût mémoire est important. À l'inverse, plus la résolution est petite et la taille de cellule élevée, plus il y a de particules par cellule mais le coup mémoire est plus faible.

Nous avons privilégié lors de nos tests une taille de cellule égale au diamètre de la particule, ce qui nous permet d'obtenir des résultats en temps interactif sur notre matériel. Nous avons réservé la dernière cellule de la grille de hachage pour les particules désactivées. Lorsque nous désactivons une particule, par évaporation, par absorption ou lorsqu'elle sort du domaine de la simulation, nous modifions l'index de cette particule dans la grille de hachage vers `0xffffffff` (valeur maximale d'un `unsigned int`) qui correspond à l'index de la dernière cellule de la grille. Grâce à ce procédé, lors du tri des particules, les particules désactivées se retrouvent automatiquement en fin de tableau.

Il nous faut calculer le nombre de particules désactivées à chaque itération. Pour cela, nous utilisons une marque supplémentaire au niveau des particules que nous fixons à 1 si la particule est désactivée. Par réduction du tableau de marques permettant de faire la somme de tous les éléments du tableau, nous obtenons le nombre de particules désactivées à partir de l'ensemble des marques.

Des émetteurs de particules servent de sources, à chaque pas de temps nous connaissons le nombre exact de particules qui sont ajoutées à la simulation. Nous connaissons donc à chaque itération le nombre de particules émises et le nombre de particules désactivées. Grâce à ces informations, nous pouvons optimiser le nombre de `threads` utilisés par notre simulation. En effet, au début, la simulation comporte peu de particules, il est alors inutile d'utiliser l'ensemble des `threads` disponibles. Plus la simulation avance, plus le nombre de particules augmente et plus nous avons besoin de `threads`. À noter que le nombre de `threads` lancés est égal au nombre de particules arrondi à la puissance de 2

supérieure, il s'agit d'une optimisation conseillée par Nvidia.

Concernant le terrain, nous utilisons un tableau stockant la position des sommets et un tableau de triplets d'indices pour les triangles. Nous utilisons également deux tableaux *indexParticle* et *indexVertex* (voir Tableau 2.1) pour déterminer quelle particule entre en collision avec quel sommet. Un tableau permet de stocker le déplacement de chaque sommet. Nous utilisons une seconde grille régulière de hachage pour stocker les triangles et accélérer la recherche de collision potentielle avec les particules. Nous nous sommes inspirés de [KS09] pour construire cette grille sur GPU. En effet, la construction de la grille intervient à chaque fois que les sommets des triangles sont déplacés et cela ne doit pas avoir d'impact sur le caractère interactif de la simulation.

Idéalement, une particule de fluide devrait interagir avec l'ensemble des triangles proches en utilisant un rayon d'action. Cette méthode impliquerait l'utilisation d'une nouvelle structure de données permettant de stocker toutes les relations (*i.e.* collisions) particules/triangles. À chaque itération, le nombre de relations est différent et limité par la taille du tableau qui les stocke. Il faut alors utiliser un critère pour déterminer les collisions que l'on prend en compte et celles que l'on ignore. Cela peut entraîner des incohérences importantes dans la simulation : certaines zones ne seraient pas modifiées parce que la simulation a atteint le nombre maximal d'interactions qu'elle peut gérer.

Nous considérons qu'une particule de fluide ne peut éroder ou sédimenter qu'un seul triangle à chaque itération, soit trois sommets. Ainsi, aucune zone d'interaction n'est omise, mais en contrepartie toutes les collisions ne sont pas prises en compte. Notre problème est de déterminer quelle particule contribue au déplacement de quel sommet. Pour cela, nous nous sommes inspirés des travaux présentés dans [Gre08], basés sur trois étapes successives utilisant des opérations atomiques.

Le tableau 2.1 décrit un exemple où les particules 0 et 2 entrent en collision avec les triangles formés par les sommets $(4, 6, 8)$ et $(2, 6, 8)$, comme illustré dans la figure 2.6. Lors de l'étape **1**, nous stockons les collisions dans les tableaux *indexParticle* et *indexVertex*, une particule peut entrer en collision avec un triangle, soit 3 sommets. Les trois premières entrées dans les tableaux correspondent à la collision entre la particule 0 et le triangle formé par les sommets $(4, 6, 8)$.
Les deux tableaux *indexParticle* et *indexVertex* sont triés ensuite selon les indices croissants des sommets (étape **2**). Nous ajoutons pour une meilleure compréhension un tableau *indexCollision* représentant l'index des collisions. Une fois les tableaux triés, la collision entre la particule 0 et le triangle formé par les sommets $(4, 6, 8)$ est stockée aux indices 1,2 et 4.
Le tableau *cellStart* est rempli en utilisant l'*indexCollision* de la première apparition de chaque sommet dans *indexVertex*. Le tableau *cellEnd* est quant à lui rempli en utilisant l'*indexCollision* de la case suivant la dernière apparition de chaque sommet dans *indexVertex* (étape **3**). Le sommet 6 apparaît dans les cases 2 et 3 dans le tableau *indexVertex* de l'étape **2**, on stocke alors 2 dans la sixième case du tableau *cellStart* et 4 dans la sixième case du tableau *cellEnd*.
Finalement, les particules qui contribuent au déplacement d'un sommet d'*indexVertex* v sont obtenues en considérant le tableau *indexParticle* trié de la case *cellStart[v]* à la case

cellEnd[v]-1 (étape **4**). Pour le sommet 6, ce sont les particules 0 et 2 qui contribuent à son déplacement, et sont stockées en 2 et 3 dans le tableau *indexParticle* trié.

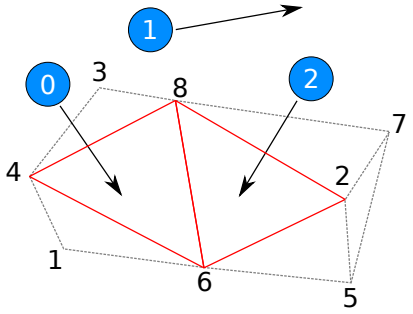


FIGURE 2.6 – Collision entre deux particules de fluide (marquées 0 et 2) et deux triangles formés des sommets (4,8,6) et (2,6,8).

Étape **1**

indexParticle	0	0	0	1	1	1	2	2	2	···
indexVertex	4	6	8	/	/	/	2	6	8	···

Étape **2**

indexCollision	0	1	2	3	4	5	6	7	8	···
indexVertex	2	4	6	6	8	8	/	/	/	···
indexParticle	2	0	0	2	0	2	/	/	/	···

Étape **3**

indexVertex	0	1	2	3	4	5	6	7	8	···
cellStart	/	/	0	/	1	/	2	/	4	···
cellEnd	/	/	1	/	2	/	4	/	6	···

Étape **4**

indexVertex	0	1	2	3	4	5	6	7	8	···
indexParticle			2		0		0,2		0,2	···

TABLE 2.1 – Recherche des particules qui contribuent aux déplacements des sommets.

2.1.2.2 Génération d'un maillage adaptatif

Nous savons que l'eau s'écoule des points élevés vers les points les plus bas. Nous avons donc choisi de raffiner les zones susceptibles d'être érodées afin de prévenir des artefacts visuels dus à la résolution du maillage originel. Ce problème connu [VHB87] implique une subdivision récursive des triangles voisins. Nous ne pouvons toutefois pas raffiner le maillage à la volée sans compromettre le caractère interactif de la simulation : cela implique d'intégrer une nouvelle phase pour raffiner le terrain en utilisant là aussi CUDA [SS09].

Nous avons décidé de raffiner le maillage en pré-traitement afin de générer plus de détails là où l'érosion doit s'appliquer. En conséquence, le maillage n'est pas modifié là

où l'érosion est faible. Nous utilisons deux critères pour déterminer les zones susceptibles d'être érodées, la pente et la profondeur des régions : plus une zone est pentue et/ou profonde, plus nous la raffinons. Le maillage est généré à partir d'une carte de hauteur, la profondeur se traduit par une couleur noire et la pente par un dégradé allant d'une couleur claire vers une couleur plus sombre (Figure 2.7). Il est possible de déterminer le gradient minimal que doit avoir un triangle pour être raffiné. Puisque nous utilisons une carte de hauteur, nous calculons la différence d'intensité de gris entre les sommets d'un triangle. Si cette différence est supérieure au seuil défini, nous raffinons le triangle. Afin d'éviter de « manquer » un dénivelé entre deux sommets, nous interpolons des points sur la surface du triangle pour obtenir une nouvelle intensité de gris. Plus le maillage est raffiné, plus il est coûteux en mémoire et calculs, mais plus la simulation est précise.

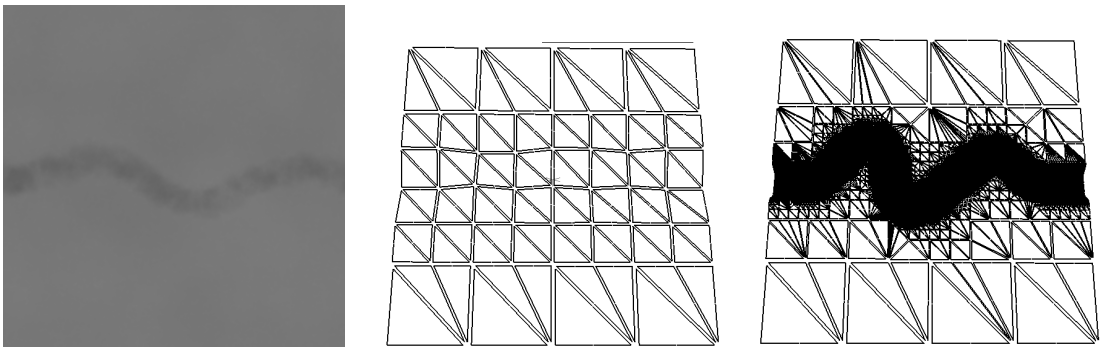


FIGURE 2.7 – Différents niveaux de subdivision du maillage à partir de la carte de hauteur (à gauche).

Les émetteurs de particules sont disposés sur le terrain, chacun est défini par une position, une largeur et une hauteur. L'utilisateur peut modifier le nombre de pas de temps entre deux émissions de particules interactivement au cours de la simulation, pour modifier le débit de la source.

2.1.3 Résultats

Après que le terrain a été généré en utilisant la méthode précédente, et que l'utilisateur a disposé les émetteurs de particules, la simulation commence. L'utilisateur peut modifier plusieurs paramètres en temps réel (pas de temps, taux d'érosion, temps de transport maximal, etc.), comme on peut le voir sur la figure 2.8. Les valeurs par défaut de ces paramètres sont listées dans le tableau 2.2. Ces paramètres ne possèdent pas d'unité.

Les impressions écrans présentées dans la figure 2.8 montrent notre application interactive gérant 130k particules et 80k triangles à approximativement 20 images par seconde sur une carte graphique NVIDIA Quadro FX 3700M. En comparaison, [KBKv09] génère une image en 1,38 secondes pour un même nombre de particules, mais ce temps inclut un rendu basique de la surface du fluide. Ici, le maillage utilisé est régulier et non adaptatif, en effet le fluide peut s'étaler sur toute la surface du terrain et générer des modifications. De plus, utiliser notre méthode de génération du maillage dans ce cas-là ne présente pas d'intérêt : les zones montagneuses ont un fort gradient, et les plaines se trouvent à basse altitude.

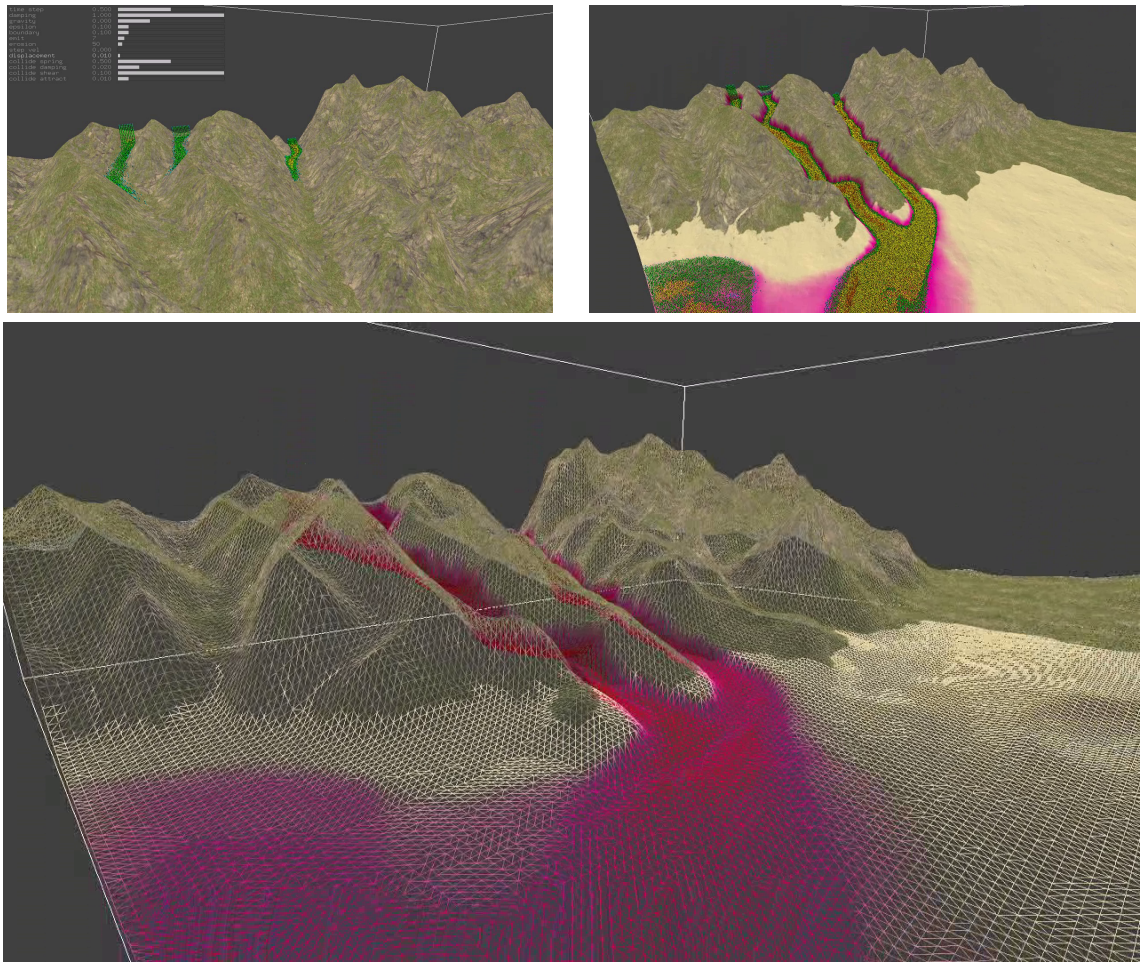


FIGURE 2.8 – Ruisseau au sommet de la montagne représentée par un maillage texturé (en haut à gauche), dévalant la pente (en haut à droite) et érodant graduellement le terrain (en bas). Les zones érodées sont en violet. Des potentiomètres peuvent être affichés pour chaque paramètre modifiable interactivement de la simulation.

Paramètre	Définition	Valeur par défaut
r	Rayon d'une particule	0.015625 (2^{-6})
deltaEmission	Nombre d'itérations entre deux émissions de particules	4
timestep	Pas de temps entre deux images de l'animation	0.5
K_c	Taux d'érosion	0.01
damping	Amortissement appliqué aux particules entrant en collision avec un triangle	0.2
v_{min}	Vélocité minimale nécessaire à une particule pour éroder un terrain	0.001
s_M	Taille maximale d'un sédiment	0.1
ε_M	Temps de transport maximal d'un sédiment attaché à une particule d'eau	25
$erosionstep$	Nombre d'itérations entre deux opérations d'érosion	50
neighborRadius	Distance maximale pour définir une particule comme voisine pour l'évaporation	$8 \times r$
v	Nombre d'itérations pour évaluer l'évaporation d'une particule	50
minNeighbor	Nombre minimal de voisines au cours de v itérations pour empêcher l'évaporation	5
minCollisions	Nombre de particules absorbées par un triangle avant qu'il ne puisse être érodé	5

TABLE 2.2 – Valeurs par défaut des paramètres utilisés dans notre simulation.

Les figures 2.9 et 2.10 montrent un autre exemple, avec un maillage initial présenté sur la figure 2.7 (à droite) creusé par une rivière, rendu utilisant POV-Ray en post-traitement, sur un supercalculateur avec 3 processeurs NVIDIA Tesla C1060 à approximativement 9 secondes par image pour 20k triangles et 130k particules. Cet exemple illustre le caractère itératif de notre méthode. Dans cette simulation, le taux d'érosion et le temps de transport maximal sont plus élevés parce que nous nous intéressons plus à l'érosion du terrain qu'à la sédimentation.

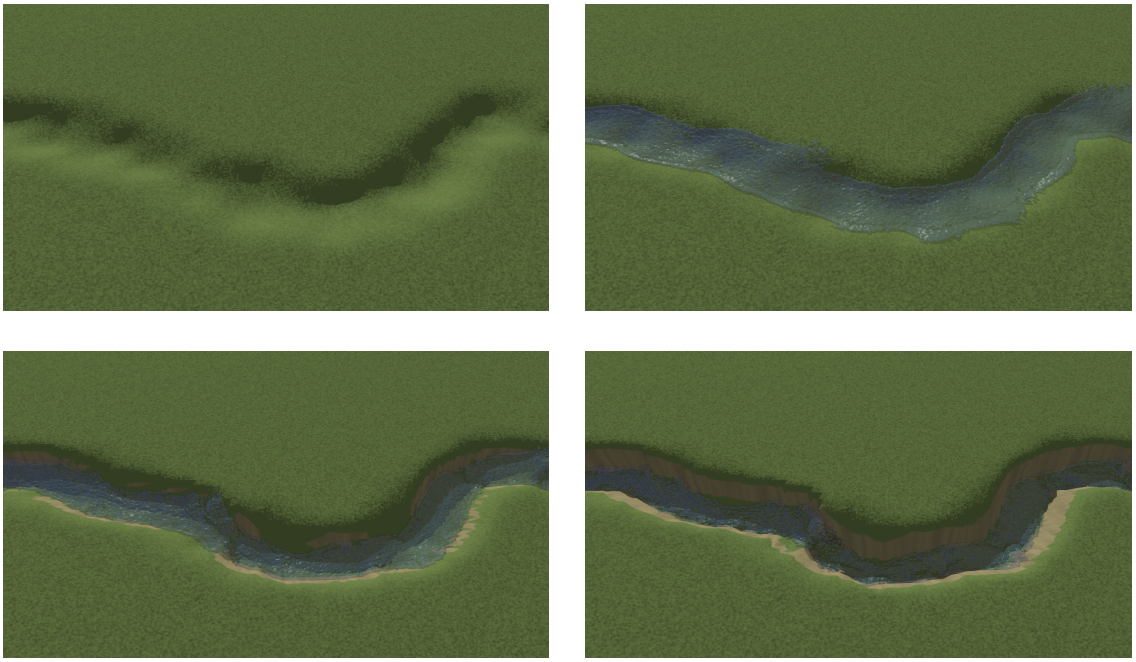


FIGURE 2.9 – Canyon creusé itérativement par une rivière.

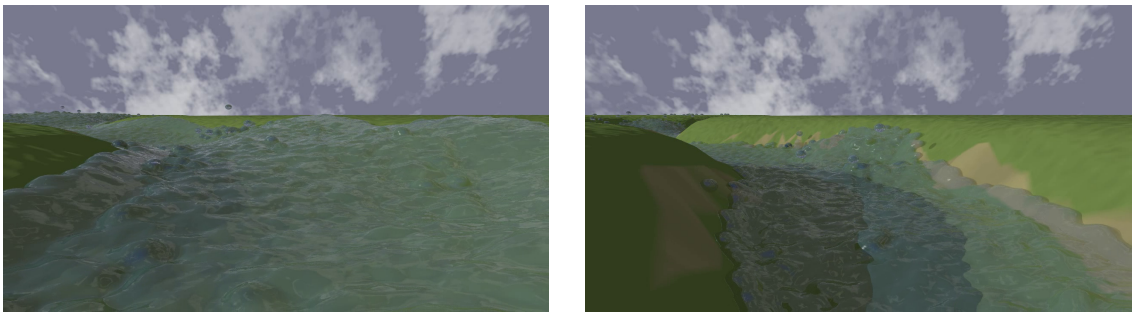
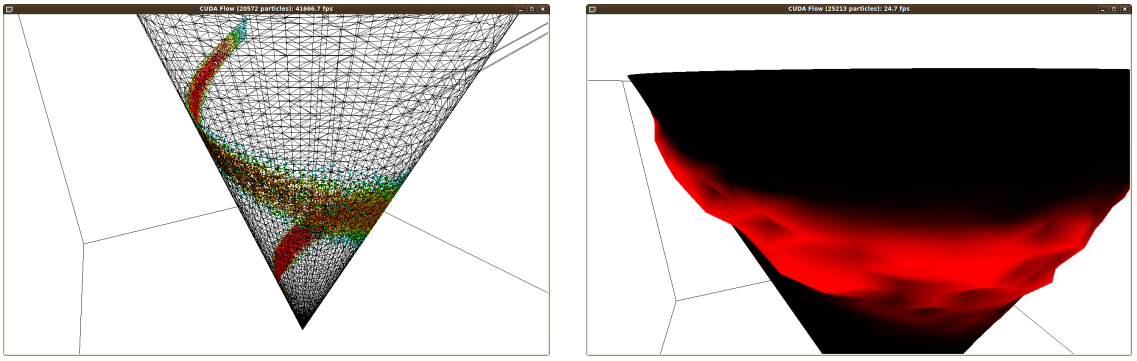


FIGURE 2.10 – Vue de l'intérieur du canyon.

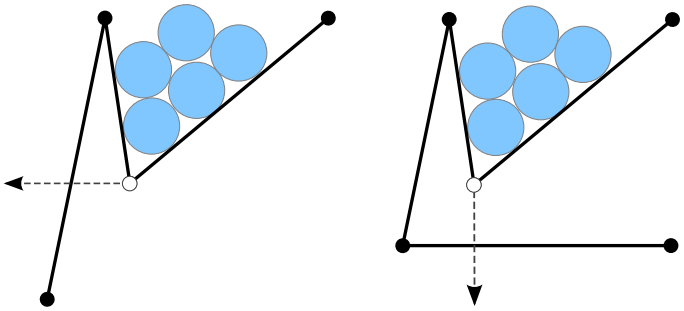
L'érosion et la sédimentation présentées dans les parties précédentes impliquent un déplacement vertical (vers le haut ou vers le bas), alors qu'on peut observer une *érosion latérale* dans des gorges ou canyons par exemple. Pour simuler cet effet, nous avons choisi de déplacer les sommets selon leur normale au lieu de l'axe vertical. Ce principe est illustré par la figure 2.11, il s'agit d'un cas théorique où les particules glissent à l'intérieur d'un cône. La figure 2.11a représente une vue de l'extérieur du cone après déplacement des sommets par érosion. Nous pouvons distinguer la formation d'irrégularités sur les parois,

ce qui correspond à la réalité (voir figure 2.11a). Néanmoins, cette méthode peut conduire à la création d'incohérences géométriques et topologiques. La figure 2.12a montre une coupe en 2D d'une simulation où un fluide érode latéralement un triangle, poussant un sommet à travers la paroi : il s'agit d'une auto-intersection du maillage. La même chose peut en fait se produire avec un surplomb dans le cas d'un déplacement vertical (Figure 2.12b). Nous inhibons les déplacements de sommets qui peuvent conduire à ce type d'incohérence, ce qui peut conduire à des résultats non réalistes visuellement. Les incohérences sont étudiées et traitées dans le chapitre suivant.



(a) (b)

FIGURE 2.11 – Test où un fluide érode latéralement l'intérieur d'un cône.



(a) (b)

FIGURE 2.12 – (a) Érosion latérale sur un sommet provoquant une auto-intersection. (b). L'érosion verticale peut aussi générer des auto-intersections si le terrain originel possède des concavités telles que des caves ou des tunnels.

2.1.4 Critiques

Cette première approche permet de simuler l'évolution d'un lit de rivière en temps interactif, à partir d'un terrain représenté par un maillage triangulaire et un système de particules. Cette méthode, comme plusieurs autres présentées dans l'état de l'art, peut être vue comme *physiquement inspirée*, puisque nous relions des principes physiques et des approximations. Notre simulation fonctionne en temps interactif, pour cela nous avons dû négliger des phénomènes comme la dissolution, l'infiltration de l'eau dans le sol, l'évaporation saisonnière, etc. Notre approche raffine le terrain au préalable là où il est susceptible

d'être modifié. Cela implique de raffiner entièrement le terrain dans le cas d'une simulation intégrant la pluie, ce qui est coûteux en mémoire. Une solution évoquée est de raffiner le terrain au fur et à mesure de la simulation, mais cette technique augmente le coût des temps de calculs. De plus, plusieurs paramètres doivent être ajustés pour obtenir un résultat plausible, ce qui est complexe car certains n'ont pas de rapport avec la physique. Ensuite, le transport des sédiments est trop approximatif et ne simule que la saltation alors que le transport de sédiments dissous n'est pas pris en compte. Enfin, le fluide suit la pente par gravité, mais d'autres forces internes au fluide doivent être prises en compte pour qu'une simulation sur un terrain avec une pente faible soit plausible.

2.2 Approche basée sur l'abaque Hjülstrom

Pour prendre en compte à la fois l'érosion mécanique et chimique, nous nous sommes basés sur le diagramme de Hjülstrom (voir Figure 2.13). Hjulström, hydrologiste suédois, propose en 1935 une représentation sous forme d'un graphe permettant de déterminer si une rivière érode, transporte ou dépose des sédiments [Hju35]. Le diagramme obtenu prend en compte la taille des sédiments en mm et la vélocité de l'eau en cm/s. Il est découpé en trois parties. La courbe supérieure représente la *vitesse critique d'érosion* v_{ec} et la courbe inférieure la *vitesse limite de sédimentation* v_{sed}. Si la vélocité du fluide est supérieure à v_{ec}, le fluide érode des sédiments ; si la vélocité du fluide est inférieure à v_{sed}, le fluide dépose des sédiments et si la vélocité du fluide se situe entre les deux, le fluide transporte les sédiments.

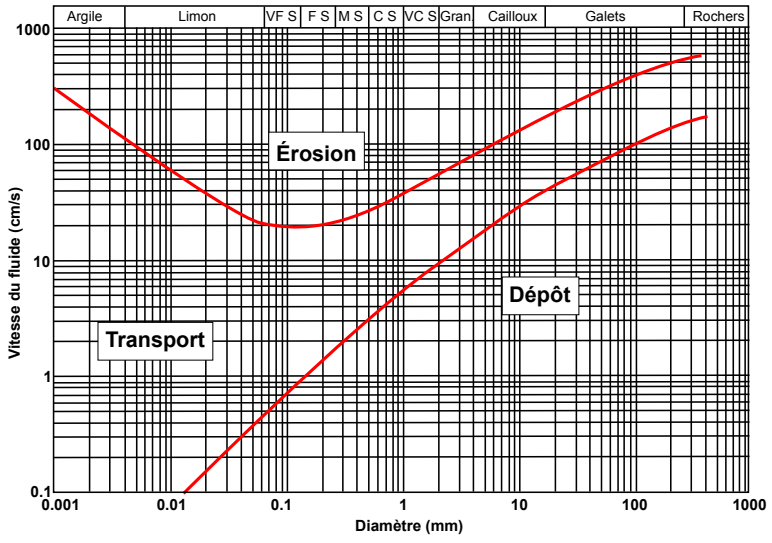


FIGURE 2.13 – Diagramme de Hjulström

L'interaction entre le fluide et le terrain n'est, ici, pas basée uniquement sur l'impact entre les particules et le terrain (distance entre les éléments nulle). En effet, les particules de fluide interagissent avec le terrain par interpolation de la distance qui les séparent. L'approche précédente ne fait interagir les particules et le terrain que lors du contact et ne correspond donc plus à notre cas de figure. Nous avons choisi d'utiliser un autre système

de particules basé sur les SPH (voir section 1.2.1.2). L'utilisation des SPH augmente le temps de calcul et le nombre de paramètres de la simulation, mais ceux-ci sont inspirés de la mécanique des fluides, ce qui rend la simulation plus facile à paramétrer : les paramètres ont des grandeurs connues (en mètre pour les distances, en gramme pour les masses, etc.). Ainsi, il est possible d'utiliser les mêmes paramètres de fluide dans différentes simulations.

Nous ne considérons plus des interactions particule/triangle mais uniquement des interactions particule/particule en discrétisant le maillage en particules dites de bords. Cela permet de distinguer la simulation de fluide, qui a déjà été implantée sur GPU, et l'évolution du terrain. La phase de modification du terrain intègre un raffinement des faces sur CPU en fonction de l'intensité des modifications qu'elles subissent.

2.2.1 Interaction avec le terrain

La solution développée dans la première approche pour simuler les collisions entre le fluide et le terrain ajoute une force externe aux particules sans tenir compte du voisinage. Cela peut impliquer des incohérences au niveau de la simulation du fluide, et les particules de fluide semblent parfois « rebondir » sur le terrain au lieu de s'écouler sur celui-ci. Les SPH utilisent des interpolations sur la distance pour calculer les forces qui s'appliquent sur les particules. Nous avons vu dans la section 1.2.1.2 qu'il faut que chaque particule de fluide ait un nombre x de voisines pour que la simulation soit stable. Cela pose problème au niveau des bords du fluide, notamment avec le terrain. Pour pallier ce problème, nous avons choisi d'utiliser la méthode décrite par Müller *et al.* [MST+04b] qui proposent d'ajouter des particules de bords sur le maillage. Les particules de fluide interagissent avec ces particules selon une méthode inspirée du potentiel de Lennard-Jones (voir section 1.2.1.1). Ce potentiel attire les particules de fluide vers la surface du terrain, ce qui permet au fluide d'y adhérer, mais les empêche de pénétrer à travers.

Les particules de bords doivent recouvrir l'ensemble du terrain en contact avec le fluide. Cette méthode nécessite de discrétiser le maillage en particules avec pour contrainte d'espacement le rayon d'interaction des particules de fluide. En effet, les particules de bords ne doivent pas être espacées d'une distance supérieure à ce rayon afin d'empêcher le fluide de passer à travers le terrain. Cependant, une discrétisation trop fine du terrain a un coût mémoire important. Plusieurs types de discrétisation de surface peuvent être utilisées, par exemple le Poisson Disk Sampling [GZWW11]. Nous avons choisi d'utiliser une discrétisation face par face linéaire qui est la plus simple à mettre en place. Nous admettons que l'échantillonnage est plus important que nécessaire, mais il nous permet de lier les particules à la face de laquelle elles sont issues (Figure 2.14).

Le maillage étant discrétisé en particules, il n'est plus nécessairement triangulé. Néanmoins, nous avons conservé des faces triangulaires pour la surface du terrain, pour permettre un rendu et un raffinement plus simple, et également certifier la planéité des faces.

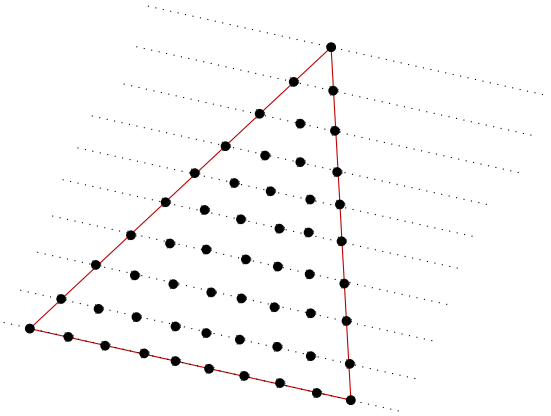


FIGURE 2.14 – Échantillonnage linéaire en particules d'un triangle.

2.2.2 Prise en compte du modèle de Hjulström

Le diagramme de Hjulström met en relation la vitesse d'écoulement du fluide et la taille des sédiments pour déterminer si le fluide érode, transporte ou dépose des sédiments. En utilisant la vélocité des particules de fluide, exprimée en m/s par le modèle de SPH et le diagramme de Hjulström, nous déterminons les échanges entre les particules de fluide et les particules de bords. Nous utilisons le même procédé que dans notre précédente approche : les sédiments contenus dans le fluide sont « attachés » aux particules de fluide. Pour qu'une particule de fluide et une particule de bords interagissent, elles doivent se trouver à une distance inférieure au rayon d'interaction.

À partir de ce rayon d'interaction, on peut assimiler une particule de fluide à une sphère qui représente un volume d'eau. Il est donc possible de calculer la capacité maximale de matière dissoute dans ce volume. Si une particule de fluide transporte déjà une quantité maximale de sédiment, elle ne peut pas éroder le terrain. Les particules de bords n'ont pas de capacité maximale : elles peuvent transmettre ou recevoir une quantité infinie de sédiments.
Lorsqu'une particule de fluide a une vélocité inférieure à v_{sed}, elle transmet une partie des sédiments qu'elle transporte à la particule de fluide la plus proche en prenant en compte la capacité de transport de celle-ci. Si la particule de fluide est voisine d'une particule de bords, elle lui transmet l'intégralité de ses sédiments. Cet échange est traité en priorité par rapport à un échange particule de fluide vers particule de fluide.

Une fois la phase d'interaction entre le fluide et le terrain terminée, nous calculons pour chaque face la différence entre la somme des sédiments érodés et la somme des sédiments déposés par le fluide. Si cette différence est positive, la face est érodée, sinon elle est sédimentée. Nous utilisons la méthode de poids, décrite dans les parties 2.1.1.1 et 2.1.1.2, sur les sommets pour déterminer le déplacement.

2.2.3 Implantation et tests

Pour comprendre le comportement des sédiments décrit par le diagramme de Hjulström, nous avons effectué plusieurs simulations.

Nous avons basé notre approche sur l'implantation sur GPU des SPH faite par Herault *et al.* [HBD10]. Nous avons ajouté une structure simple aux particules composée de deux `float` et un `unsigned int` représentant les informations relatives aux sédiments et à la géologie.

```
struct geodata
{
 unsigned int type_matiere;
  float qte_erod;
  float qte_sed;
 unsigned int face;
}
```

Le `type_matiere` représente le type de sédiment, il s'agit d'un index relatif à un tableau stockant les informations concernant les différents types de matériaux (voir Tableau 2.3). Nous assumons avoir défini une seule valeur de vitesse critique d'érosion et une seule vitesse limite de sédimentation pour chaque type de matériau. Une particule de fluide utilise la variable `qte_sed` pour stocker la quantité de sédiment transportée. Pour une particule de bords, `qte_sed` sert à stocker la quantité de sédiment reçue par sédimentation et `qte_erod` stocke la quantité de sédiment prélevée par érosion. La variable `face` est l'index de la face à laquelle appartient la particule de bords.

Dénomination	Type	Taille du sédiment (mm)	Vitesse critique d'érosion (m/s)	Vitesse limite de sédimentation (m/s)
Argile	0	0.003	1	0
Limon	1	0.02	0.3	0.2
Sable	2	1	2.3	0.05
Caillou	3	10	1	0.3
Galet	4	50	3	0.8
Rocher	5	500	6	1.5

TABLE 2.3 – Comportement des différents types de sédiments étudiés par Hjulström.

Nous avons également ajouté un premier *kernel* effectuant l'échange de matière par érosion, ainsi qu'un second *kernel* effectuant l'échange de matière par sédimentation : entre particules de fluide et depuis les particules de fluide vers les particules de bords. Nous utilisons des opérations atomiques afin de prévenir les pertes de données dues aux accès concurrents aux données.
Le premier *kernel* effectue l'échange de matière depuis les particules de bords vers les particules de fluide. Il peut donc y avoir un accès concurrent aux données au niveau des sédiments transportés par une particule de fluide. Par exemple, sur la figure 2.15a, il y a un accès concurrent à la donnée qte_{sed} de la particule de fluide `F2` de la part des particules de bords `T1` et `T2`. Nous utilisons alors une opération atomique pour ajouter une quantité x de matière à la variable qte_{sed} de F2. Cette opération prend également en compte la

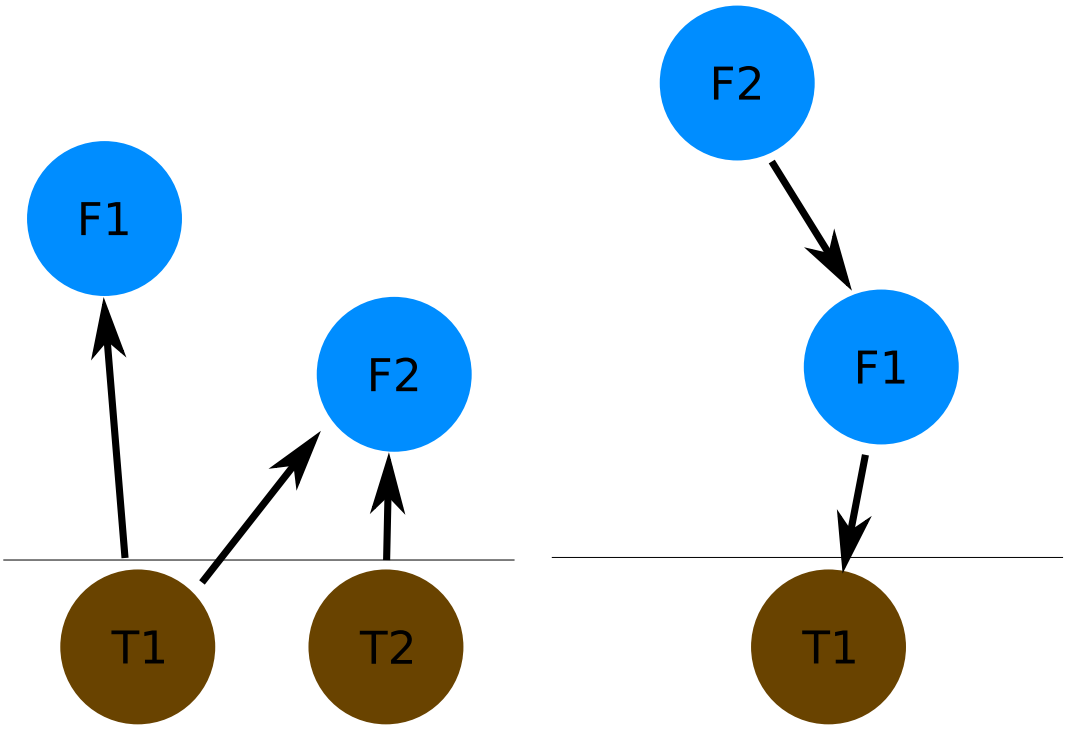


FIGURE 2.15 – F1 et F2 sont des particules de fluide, T1 et T2 sont des particules de bords. Les flèches représentent les échanges de matière. (a) Dans le cas de l'érosion, (b) Dans le cas de la sédimentation.

capacité de transport du fluide et retourne la quantité de matière érodée effective que nous ajoutons à la variable qte_{erod} de la particule de bords. Ainsi, si la particule de fluide a déjà atteint sa capacité de transport maximale, elle n'érode pas de matière.
Le second *kernel* effectue l'échange de matière depuis une particule de fluide vers une particule de bords ou une particule de fluide. Il peut y avoir un accès concurrent aux données au niveau des sédiments transportés par une particule de fluide, mais également au niveau de la quantité de sédiment reçue par une particule de bords. Par exemple sur la Figure 2.15b, la particule de fluide `F1` transmet et reçoit des sédiments en même temps. Nous assumons que le traitement est effectué dans un ordre aléatoire selon que la particule `F1` ou `F2` est traitée en premier par le GPU. Nous effectuons l'échange en deux temps : dans un premier temps, nous ajoutons une portion de sédiments transportés par `F2` à `F1` à l'aide de la même opération atomique que pour l'érosion. Et dans un second temps, nous soustrayons la quantité transmise effective à la particule `F2` à l'aide d'une opération atomique. Nous effectuons le même traitement pour l'échange entre une particule de fluide `F1` et une particule de bords `T1`, sauf que l'on transmet la totalité des sédiments transportés. Il est à noter qu'une particule de bords peut recevoir tous les sédiments transportés par les particules de fluide avoisinantes.

Test : vitesse critique d'érosion

La courbe de Hjulström (Fig. 2.13) montre la vitesse critique du fluide pour qu'il érode un certain type de terrain. Afin de tester cette propriété, nous avons simulé un écoulement

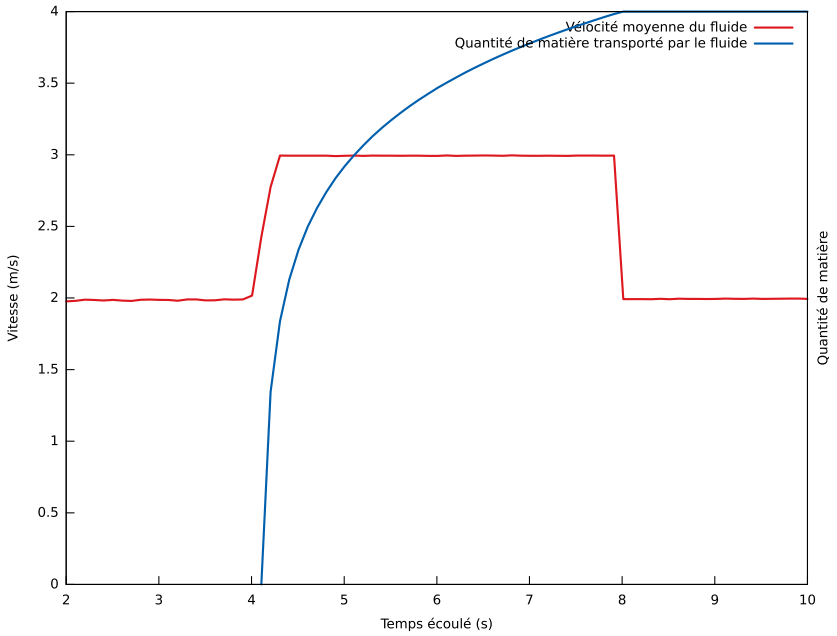


FIGURE 2.16 – Évolution de la quantité de sédiments présents dans le fluide par rapport à sa vélocité moyenne.

de fluide sur une surface érodable composée de sable et représentée par un plan incliné. Nous avons spécifié que la vitesse critique d'érosion du sable est de $2,3m/s$ et inhibé les modifications géométriques du terrain pour conserver un écoulement linéaire. Pour pouvoir contrôler la vitesse du fluide, nous avons également majoré la vitesse des particules et nous faisons varier cette vitesse maximale. Nous avons obtenu les résultats décrits sur la figure 2.16. Jusqu'à $4s$ de simulation, la vitesse moyenne du fluide est de $2m/s$, la quantité de matière érodée est nulle puisque la vitesse critique d'érosion est supérieure. De $4s$ à $8s$, la vitesse moyenne du fluide monte à $3m/s$, la quantité de matière érodée augmente. Finalement, nous établissons une vitesse moyenne d'écoulement de $2m/s$, la quantité de matière érodée est stable. Notre modèle prend en compte la vitesse critique d'érosion.

Test : vitesse critique de sédimentation

Il nous reste à tester la vitesse critique de dépôt du fluide ainsi que sa propriété de transport de la matière. Pour cela, nous avons créé une seconde simulation en « circuit fermé » : nous avons placé un volume de fluide dans un cube dont la partie inférieure est constituée d'un plan représentant du sable. Les particules de fluide en rouge sur la figure 2.17 possèdent une quantité x de matière. Nous avons fixé la vitesse critique de dépôt du sable à $0.05m/s$. La vitesse moyenne du fluide une fois stabilisé est nulle. Les sédiments sont transmis par gravité des particules de fluide du haut vers le bas jusqu'à atteindre le terrain où ils sont transmis aux particules de bords. La figure 2.18 montre la progression de la quantité de matière présente dans le fluide et de la quantité de matière captée par le terrain au cours du temps. Nous remarquons la complète symétrie des deux courbes. Nous observons également qu'en fin de simulation, la totalité des sédiments ont été captés par les particules de bords. Nous montrons donc que notre simulation conserve bien les quantités de sédiments présents.

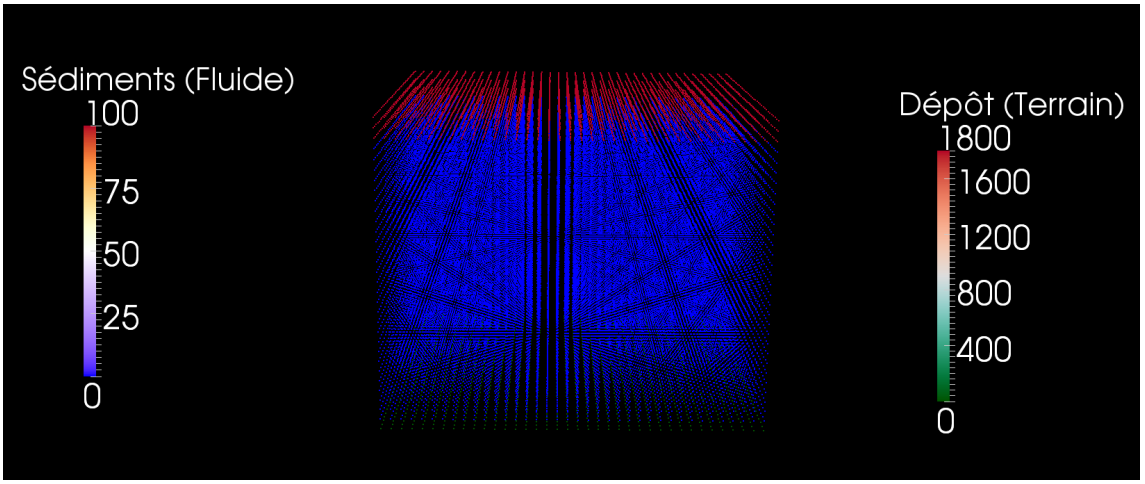


FIGURE 2.17 – Configuration initiale du fluide stationnaire.

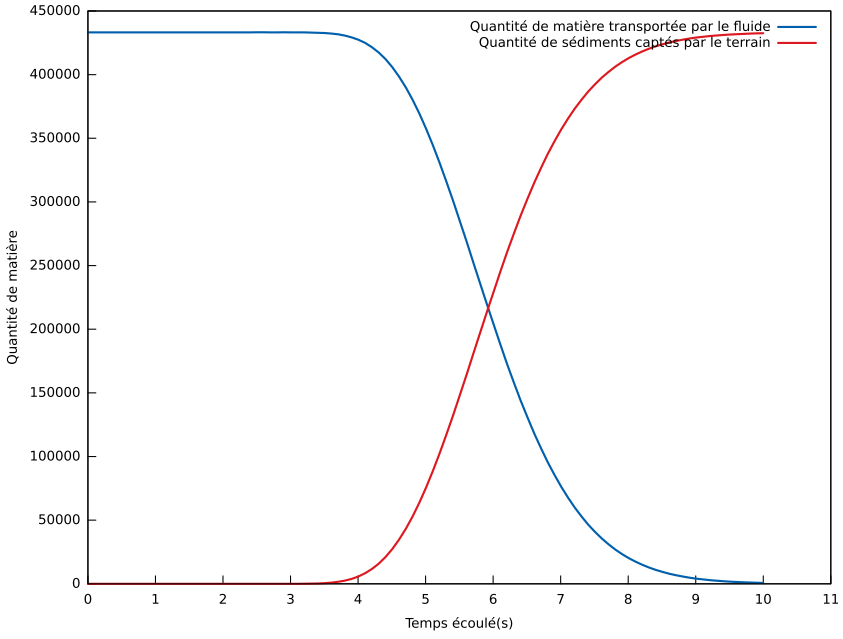
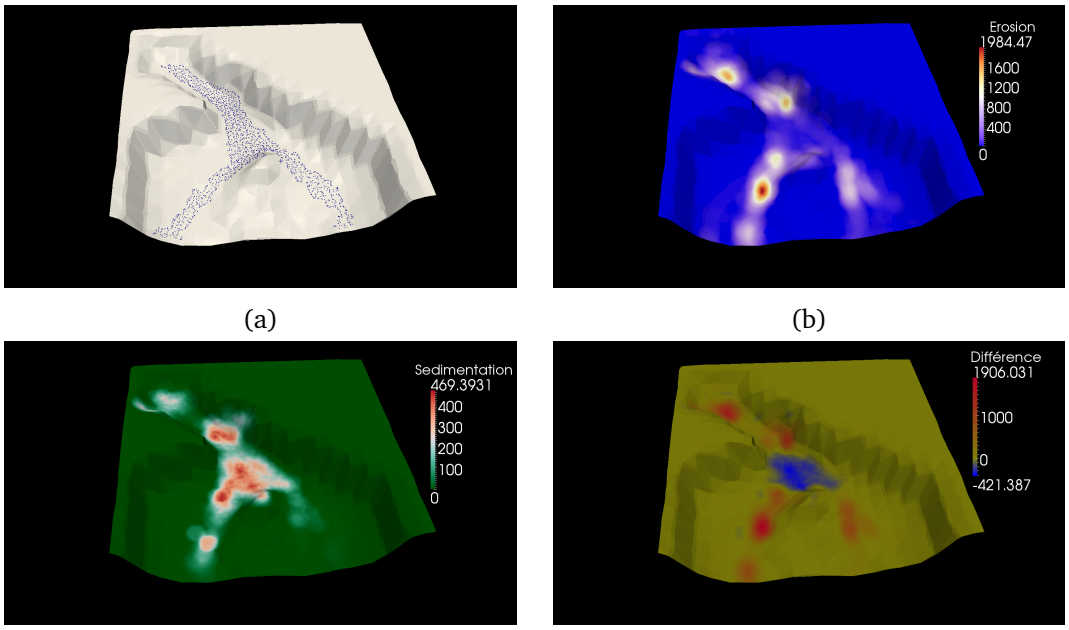


FIGURE 2.18 – Transfert de sédiments dans un fluide stationnaire.



(a) (b)

(c) (d)

FIGURE 2.19 – Visualisation des zones d'influence du fluide sur le terrain. (a) Zone d'écoulement du fluide discrétisé en particules, (b) Zone d'érosion, (c) Zone de sédimentation, (d) Différence entre la zone d'érosion et la zone de sédimentation. (Images réalisées avec Paraview)

Écoulement d'une rivière

Le terrain est composé d'une zone avec une pente abrupte découlant sur une plaine. La rivière en s'écoulant rencontre un obstacle au niveau de la plaine et se ramifie (Figure 2.19a). Nous pouvons constater que l'érosion est importante au niveau de la pente abrupte (Figure 2.19b), alors que la sédimentation est plus importante lorsque le fluide rencontre l'obstacle (Figure 2.19c), ce que confirme la figure 2.19d qui représente la différence entre les sédiments prélevés (érosion) et les sédiments déposés (sédimentation) : une zone bleue représentant la zone de sédimentation la plus importante est clairement visible au niveau de l'obstacle.

2.3 Conclusion

À travers cette section, nous avons montré qu'il était possible d'obtenir un fluide basé sur un système de particules ayant une capacité d'érosion, de transport et de dépôt plausible. Dans un premier temps, nous avons utilisé un DEM pour simuler le fluide et une interaction basée sur la collision entre les particules et les triangles pour simuler l'interaction avec le terrain. Cette méthode s'est révélée difficile à régler, notamment du fait que les valeurs utilisées n'ont pas d'unité. De plus, elle ne permet de simuler que la saltation des sédiments. Notre seconde approche basée sur les SPH a permis de pallier ce problème et également d'appuyer notre simulation sur le diagramme de Hjulström, permettant de justifier les échanges de sédiments entre le fluide et le terrain. La physique plus précise de notre seconde approche permet de définir des obstacles qui influent à la fois sur les zones

d'érosion et de sédimentation. En effet, en nous basant sur le diagramme de Hjulström, nous pouvons maintenant affirmer qu'il y a sédimentation dans les zones où le fluide ralentit, c'est à dire lorsque le fluide rencontre un obstacle ou quand la pente diminue. Cela n'est pas forcément le cas avec la première solution puisque la simulation utilise un compte à rebours afin de déposer les sédiments.

Nous avons également pris en compte les déplacements de matière en modifiant un terrain représenté de manière surfacique et en le raffinant en conséquence. Notre modèle de fluide suit ces modifications grâce aux particules de bords.

Cependant, la représentation d'un terrain seulement par sa surface ne permet pas de prendre en compte les différentes couches sous-jacentes et leurs propriétés (type de roche, résistance à l'érosion, etc). Pour pallier ce problème, nous pouvons utiliser un maillage similaire à celui utilisé dans [PTM+11]. Néanmoins, cette solution ne permet pas de gérer les incohérences topologiques et géométriques engendrées par le déplacement des sommets. Pour cela, nous avons développé une solution basée sur le modèle des 3-G-cartes (voir section 1.3) afin de gérer les différents cas et ainsi de maintenir la cohérence du maillage. Nous exposons cette solution dans le chapitre suivant.

Chapitre 3

Modèle d'évolution topologique 3D pour la géomorphologie

Sommaire

Chapitre 3

Modèle d'évolution topologique 3D pour la géomorphologie

Introduction

La simulation de fluide permet de déterminer le déplacement des sommets représentant la surface du terrain. Cependant, si l'on souhaite que ce déplacement soit cohérent avec la réalité, et donc pas limité à une seule direction, la représentation du terrain par sa surface n'est pas suffisante. Un terrain est composé d'un ensemble de couches de matière ayant des caractéristiques et des propriétés différentes. Par exemple, une couche constituée de sable n'est pas érodée de la même manière qu'une couche d'argile. Notre modèle de terrain doit prendre en compte cette structure en couches et permettre de modifier le terrain de manière cohérente en fonction de l'érosion et/ou de la sédimentation subies, mais également des comportements d'évolutions différents liés à la nature des couches.

Nous avons vu dans le chapitre 1 plusieurs types d'approches représentant les couches du terrain :

- des méthodes utilisant un maillage surfacique, ne permettant pas les modifications latérales ;
- des méthodes basées sur les voxels ou des strates, lourdes en mémoire et limitées à la taille et la résolution d'une grille ;
- des méthodes de représentation utilisant un maillage volumique, mais nécessitant des opérations de modifications topologiques et géométriques complexes.

Nous nous sommes basés sur cette dernière représentation, utilisant les 3-G-cartes que nous avons décrites dans la section 1.3. Néanmoins, le modèle présenté ne permet pas de gérer les incohérences géométriques et topologiques générées par l'évolution des couches, comme celle illustrée par la figure 3.1.
Nous analysons dans ce chapitre les incohérences géométriques et topologiques nécessitant des modifications pour obtenir une évolution cohérente du terrain. Nous décrivons ensuite le modèle permettant de gérer les événements basés sur les déplacements de sommets. Nous détaillons par la suite les opérations traitant les incohérences détectées. Enfin, nous mettons ce modèle à l'épreuve à travers divers tests et montrons plusieurs résultats

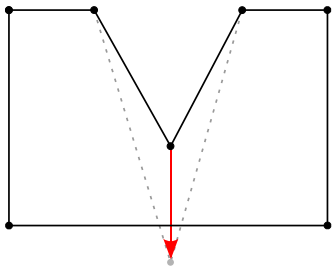


FIGURE 3.1 – Exemple 2D d'incohérence topologique : un sommet se déplace (flèche rouge) et traverse un côté du polygône.

dans le contexte de la géomorphologie.

Les résultats présentés dans ce chapitre ont fait l'objet d'une présentation aux Journées de l'AFIG 2011 [BCS+11a] et d'une publication dans Virtual Reality Interaction and Physical Simulation [BCS+11b].

Rappel
Dans la suite du document, nous noterons α_{ij} la composition des involutions α_i et α_j :

$$\alpha_{ij}(b) = \alpha_j \circ \alpha_i(b) = \alpha_j(\alpha_i(b))$$

3.1 Modèle d'animation

Le modèle géométrique que nous utilisons représente uniquement le plongement et la topologie des objets. Pour le faire évoluer au cours du temps, nous utilisons le modèle d'animation présenté par PF. Léon *et al.* dans [LSM08] que nous avons étendu à la 3D. Nous décomposons le mouvement global d'un objet à un ensemble de mouvements distincts de ses sommets. Ces mouvements conduisent l'objet à changer de forme, ce qui peut entraîner des incohérences aussi bien géométriques que topologiques.

Définition 4. *Nous appelons* `événement` *la collision d'une i-cellule en mouvement avec une j-cellule, avec i et $j \in \{0, 1, 2\}$.*

Ces collisions peuvent générer des incohérences. Nous devons les détecter pour ensuite les traiter afin d'assurer la stabilité du modèle topologique. Nous utilisons la dimension des cellules concernées par un événement pour les classifier dans la suite du chapitre.

3.1.1 Détection des événements

Les objets que nous traitons sont en trois dimensions. Les collisions possibles sont au nombre de neuf :

- sommet / sommet,

- sommet / arête,
- sommet / face,
- arête / sommet,
- arête / arête,
- arête / face,
- face / sommet,
- face / arête,
- face / face.

Les collisions ne sont pas symétriques. En effet, afin de simplifier notre modèle, nous considérons les déplacements des sommets comme indépendants. Par exemple, si une arête est en mouvement, nous déplaçons indépendamment un des sommets, puis l'autre. C'est pour cette raison qu'une collision sommet / arête n'est pas traitée comme une collision arête / sommet. Les types de collisions sont étudiés dans la suite de cette section.

Les déplacements de sommet sont considérés comme rectilignes uniformes et les faces des volumes comme planes. Pour chaque déplacement, nous vérifions si un événement a lieu. Bien que la détection des collisions se fasse de manière géométrique, nous distinguons également les collisions en fonction de la topologie : une collision entre deux sommets quelconques n'est pas le même événement qu'une collision entre deux sommets liés par une arête.
Notre méthode de détection d'événement effectue des tests entre l'ensemble des cellules. Cette méthode est basique, simple à mettre en place mais peu efficiente. La méthode décrite par Jund *et al.* [JCD09] [JCD10] est basée sur la détection de collisions à l'aide de particules. Elle permet de calculer les collisions entre un ensemble de particules en mouvement et un environnement composé de polyèdres convexes. Elle retourne l'instant, la position de la collision, le type de cellule rencontrée (sommet, arête, face) et la normale à la surface. Cette méthode n'était pas disponible au moment de l'étude de notre approche, mais elle pourrait être intégrée dans nos futurs travaux.

Dans les cas suivants, nous appelons respectivement A, $\vec{\Delta}$ et A', le sommet étudié, son vecteur de déplacement au cours du pas de temps dt et le sommet issu de la translation de A par $\vec{\Delta}$. Le déplacement d'un sommet est considéré comme rectiligne uniforme. Nous pouvons assimiler le déplacement de A au segment de droite $[AA']$.

3.1.1.1 Collision sommet-sommet

Nous appelons B le second sommet avec lequel nous effectuons le test de collision. Une collision a lieu si B appartient à la droite $[AA']$. L'instant t de la collision est calculé de la manière suivante :

$$t = \frac{\|AB\|}{\|AA'\|}$$

B appartenant au segment $[AA']$, $0 \leq t \leq dt$.

Si A et B entrent bien en collision, nous distinguons 4 cas :

- A et B sont liés par une arête (Fig 3.2a),
- A et B appartiennent à la même face (Fig 3.2b),
- A et B appartiennent au même volume (Fig 3.2c),

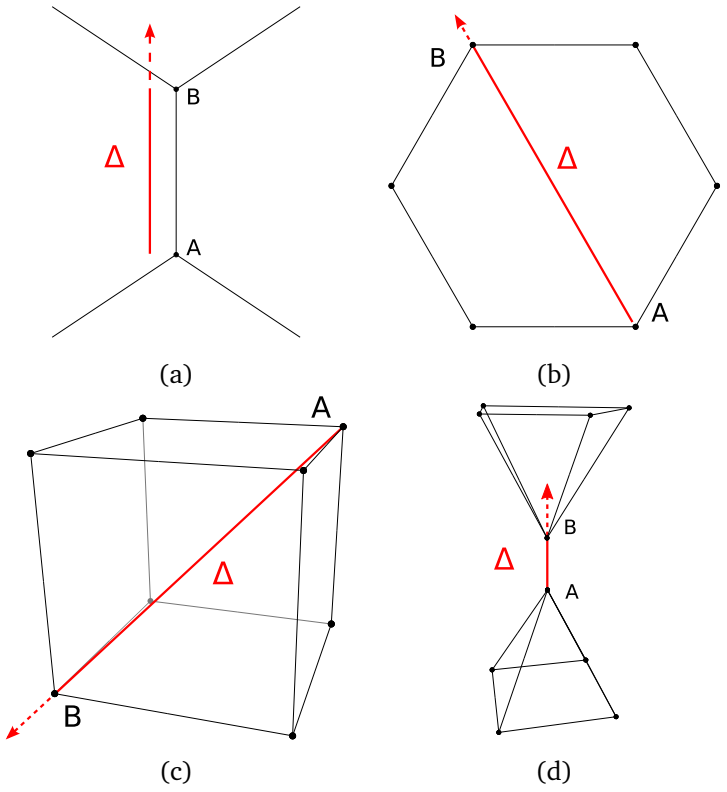


FIGURE 3.2 – Différents cas de figures de la collision sommet-sommet. Le sommet A a pour vecteur de déplacement $\vec{\Delta}$, nous testons une collision sommet-sommet avec B.

- A et B n'appartiennent pas au même volume (Fig 3.2d).

3.1.1.2 Collision sommet-arête

Nous appelons B_1 et B_2 les sommets de l'arête avec laquelle nous effectuons le test de collision. Une collision a lieu s'il y a intersection entre les segments $[AA']$ et $[B_1B_2]$. Nous appelons I le point d'intersection. L'instant t de collision est calculé de la manière suivante :

$$t = \frac{\|AI\|}{\|AA'\|}$$

Si A et l'arête entrent bien en collision, nous distinguons 3 cas différents :

- A et l'arête appartiennent à la même face (Fig 3.3a),
- A et l'arête n'appartiennent pas à la même face (Fig 3.3b),
- A et l'arête n'appartiennent pas au même volume (Fig 3.3c).

Nous ne vérifions pas si le sommet A est un sommet de l'arête, en effet ce cas est déjà traité par le test sommet-sommet.

3.1.1.3 Collision sommet-face

Nous appelons F la face avec laquelle nous effectuons le test. F appartient au plan P. Une collision a lieu quand les conditions suivantes sont réunies :

- il y a une intersection entre $[AA']$ et (P), nous appelons I le point d'intersection,

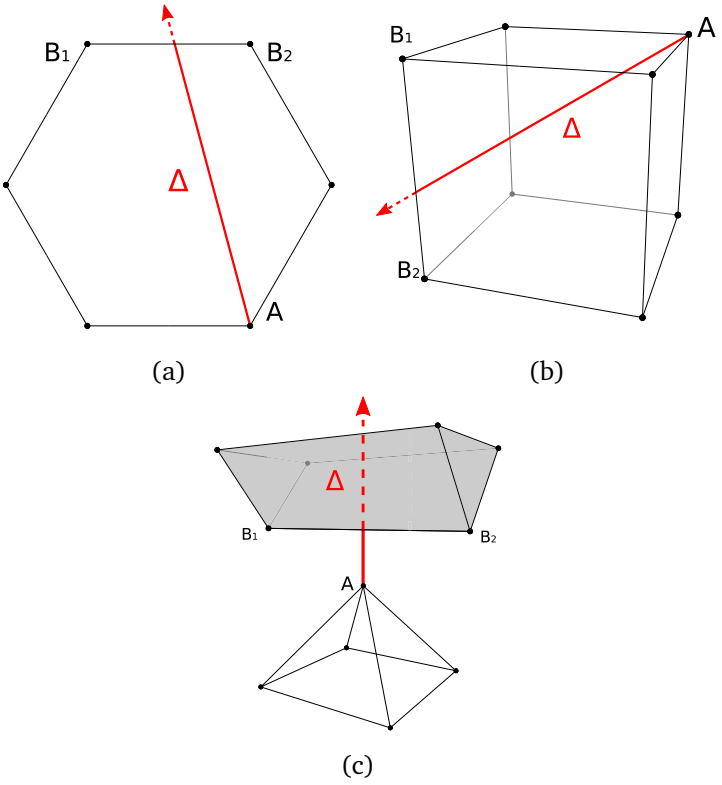


FIGURE 3.3 – Différents cas de figures de la collision sommet-arête. Le sommet A a pour vecteur de déplacement $\vec{\Delta}$, nous testons une collision sommet-arête avec le segment $[B_1B_2]$.

- I est à l'intérieur de la face F.

Le calcul de l'instant de collision t est le même que pour la collision sommet-arête.

Si A et F entrent bien en collision, nous distinguons 2 cas différents :

- A et F appartiennent au même volume (Fig 3.4a),
- A et F n'appartiennent pas au même volume (Fig 3.4b).

Dans les cas suivants, nous rappelons que nous déplaçons indépendamment les sommets de l'arête. Nous appelons A et B les sommets de l'arête, $\vec{\Delta}$ le mouvement du sommet A, A' le point issu de la translation de A par $\vec{\Delta}$ et P le plan auquel appartiennent A, A' et B.

3.1.1.4 Collision arête-sommet

Nous appelons S le sommet avec lequel nous effectuons le test. Une collision a lieu s'il y a une intersection entre les segments $[AA']$ et $[BS]$. Nous appelons I le point d'intersection. Le calcul de l'instant de collision t est le même que pour la collision sommet-arête.

Si l'arête et S entrent bien en collision, nous distinguons 3 cas différents :

- l'arête et S appartiennent à la même face,
- l'arête et S n'appartiennent pas à la même face,

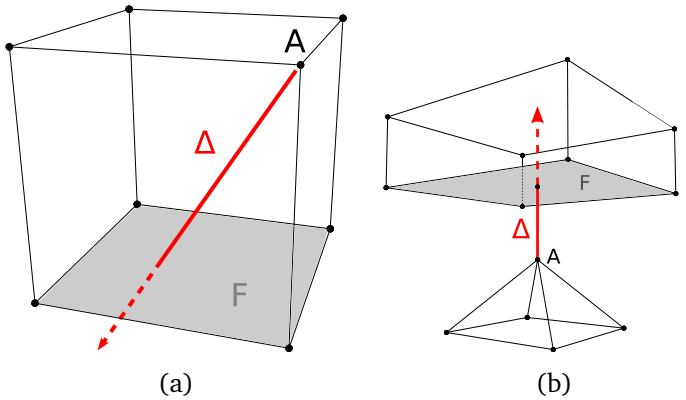


FIGURE 3.4 – Différents cas de figures de la collision sommet-face. Le sommet A a pour vecteur de déplacement $\vec{\Delta}$, nous testons une collision sommet-arête avec la face F.

- l'arête et S n'appartiennent pas au même volume.

3.1.1.5 Collision arête-arête

Nous appelons S_1 et S_2 les sommets de l'arête avec laquelle nous effectuons le test. Une collision a lieu quand les conditions suivantes sont réunies :

- il y a une intersection entre le segment $[S_1S_2]$ et le plan P, nous appelons I le point d'intersection,
- I appartient au triangle $AA'B$.

Le calcul de l'instant de collision t est le même que pour la collision sommet-arête.

Si les deux arêtes entrent bien en collision, nous distinguons 2 cas différents :

- les deux arêtes appartiennent au même volume (Fig 3.5a),
- les deux arêtes n'appartiennent pas au même volume (Fig 3.5b).

3.1.1.6 Collision arête-face

Nous appelons F la face avec laquelle nous effectuons le test. Nous rappelons que dans notre contexte, nous déplaçons les sommets indépendamment. Une collision arête-face ne peut donc avoir lieu que si un des sommets de l'arête appartient à la face F et si l'autre sommet entre en collision avec la face F (voir Figure 3.6). Supposons que B appartient à la face F. Alors, une collision a lieu s'il y a une intersection sommet-face entre A et F. L'instant t de collision est celui de la collision sommet-face.

Le déplacement d'un sommet A d'une face ne provoque un changement géométrique qu'au niveau de ses deux voisins directs B et C. Nous considérons donc seulement les collisions triangle-sommet. Soit $\vec{\Delta}$ le vecteur de déplacement du sommet A au cours du pas de temps dt.

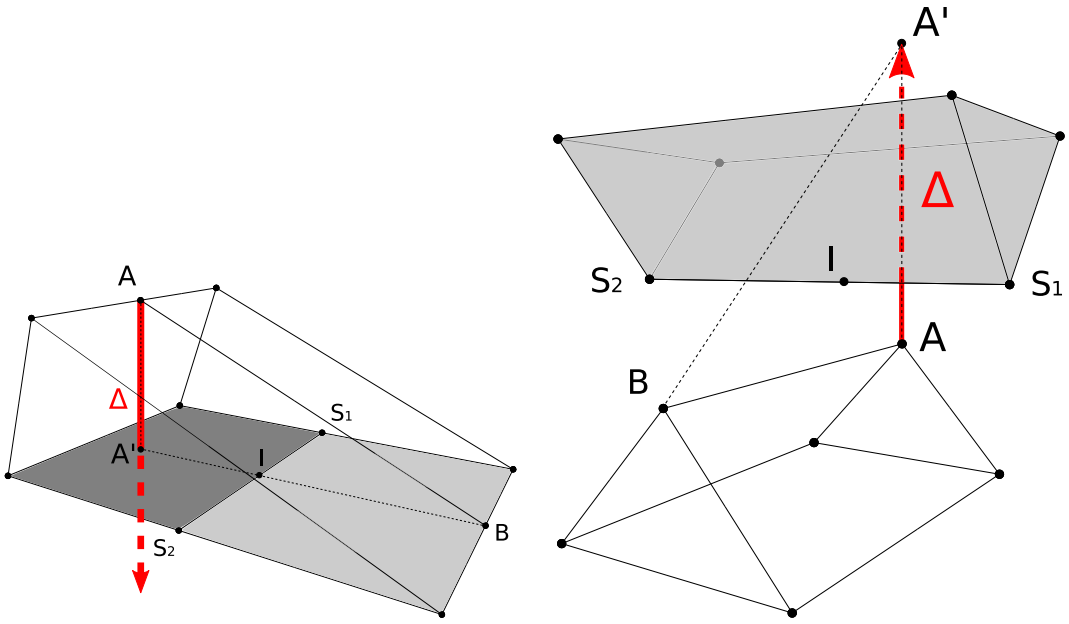


FIGURE 3.5 – Différents cas de figures de la collision arête-arête. Le sommet A a pour vecteur de déplacement $\vec{\Delta}$, nous testons une collision arête-arête avec le segment $[S_1S_2]$.

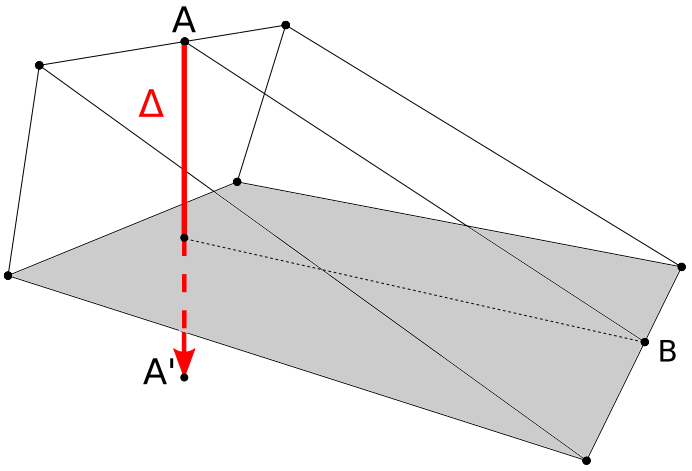


FIGURE 3.6 – Cas de figure de la collision arête-face. Le sommet A a pour vecteur de déplacement $\vec{\Delta}$, nous testons une collision arête-face avec le face grisée.

3.1.1.7 Collision face-sommet

Nous appelons M le sommet avec lequel nous effectuons le test de collision. Afin de détecter une collision entre la face BCA et M, nous cherchons l'instant t auquel les points $A(t)$, B, C et M sont coplanaires et nous vérifions ensuite si M appartient au triangle $BCA(t)$.

Nous calculons t à partir de l'intersection entre le plan (BCM) et la droite passant par A et de vecteur directeur $\vec{\Delta}$. Nous appelons I le point d'intersection.

$$t = \frac{\|AI\|}{\|\vec{\Delta}\|}$$

Il y a intersection si $0 \leq t \leq dt$ et si $M \in (BCA(t))$.

3.1.1.8 Collision face-arête

Nous appelons S_1 et S_2 les sommets formant l'arête avec laquelle nous effectuons le test. Comme pour la collision arête-face, ce cas de figure survient quand une extrémité de l'arête appartient à la face ABC. Supposons que S_2 appartient à ABC, il y a alors collision entre la face et l'arête s'il y a une collision face-sommet entre ABC et S_1. L'instant t de la collision est celui de la collision face-sommet.

3.1.1.9 Collision face-face

Dans notre contexte de déplacement indépendant des sommets, une collision face-face ne peut se produire que si les deux faces possèdent au moins une arête commune (voir Figure 3.7). Nous appelons F_2 la face avec laquelle nous effectuons le test. Il y a collision entre ABC et F_2 si :

- tous les sommets de F_2 entrent en collision avec ABC,
- ou si tous les sommets de ABC entrent en collision avec F_2.

Nous utilisons le test face-sommet pour déterminer s'il y a collision ou non entre une face et un sommet. Nous n'effectuons pas le test pour les sommets communs aux deux faces.

3.1.2 Gestion des collisions

Notre modèle d'animation se base sur le déplacement continu des sommets au cours d'un laps de temps. Nous vérifions la cohérence topologique et géométrique de nos objets au cours de ce laps de temps à l'aide des tests décrits précédemment. Néanmoins, il nous faut gérer ces événements dans un certain ordre. Nous traitons l'événement le plus proche au cours du laps de temps : cela implique en général des modifications topologiques pour respecter la cohérence, comme par exemple la création ou la suppression de nouveaux sommets. Ces modifications doivent être prises en compte, nous effectuons à nouveau les tests de collisions et traitons l'événement le plus proche, ainsi de suite jusqu'à ce que tous les événements ayant lieu au cours du laps de temps soient traités.

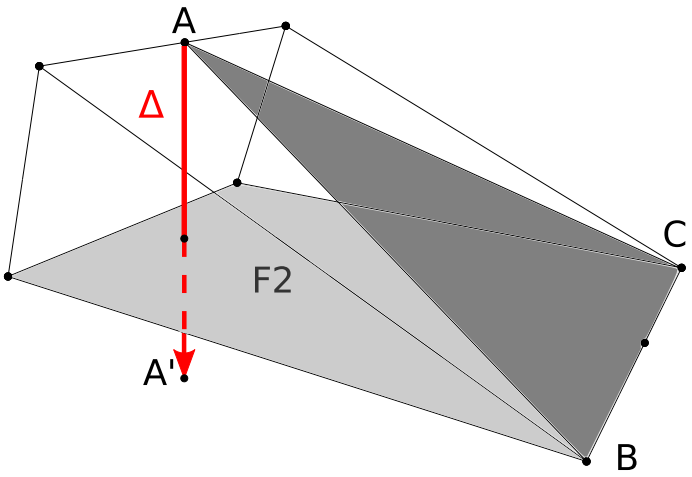


FIGURE 3.7 – Cas de figures de la collision face-face. Le sommet A a pour vecteur de déplacement $\vec{\Delta}$, nous testons une collision face-face avec la face F2.

3.2 Nouvelles opérations topologiques

Dans cette partie, nous présentons de nouvelles opérations topologiques dont nous avons avons besoin lors du traitement des événements. Les opérations d'éclatement d'un sommet en arête et de division d'un sommet appartenant à plusieurs volumes sont également utilisées dans les opérations de traitement des événements.

3.2.1 Éclatement d'un sommet en arête

L'opération d'éclatement d'un sommet S en arête permet de diviser un sommet en deux pour ensuite pouvoir les déplacer indépendamment l'un de l'autre. Ces deux sommets S_1 et S_2 à l'issue de l'opération sont liés par une arête. La figure 3.8 représente la vue plongée et la vue éclatée avant et après l'opération. La position de la caméra sur cette figure représente le point de vue utilisé pour les schémas de la figure 3.9. Notre algorithme 3.1 a pour argument deux brins dA et dB qui déterminent comment doit être scindé le sommet. Sur la figure 3.9b, la découpe est représentée par le trait noir en pointillé.

Algorithme 3.1 : ÉclatementSommetArête(Brin dA, Brin dB)

Entrée : Un Brin dA et un Brin dB appartenant à la même orbite sommet (Figures 3.8c et 3.9a).

Brin dA1 $\leftarrow \alpha_1(dA)$;
Brin dB1 $\leftarrow \alpha_1(dB)$;

// On découd par α_1 pour obtenir deux sommets distincts (Figure 3.9b).
Pour chaque *Brin b* $\in < \alpha_2, \alpha_3 > (dA)$ **Faire**
  Si *b n'est pas libre par* α_1 ***ET*** $\alpha_1(b) \in < \alpha_2, \alpha_3 > (dA1)$ **Alors**
    découdre$_1$(*b*);
  FinSi
FinPour
Pour chaque *Brin b* $\in < \alpha_2, \alpha_3 > (dB)$ **Faire**
  Si *b n'est pas libre par* α_1 ***ET*** $\alpha_1(b) \in < \alpha_2, \alpha_3 > (dB1)$ **Alors**
    découdre$_1$(*b*);
  FinSi
FinPour

// On clôt les faces non fermées de l'orbite sommet de dA en liant par une arête les brins libres par α_1 (Figure 3.9c).
Pour chaque *Brin b* $\in < \alpha_1, \alpha_2, \alpha_3 > (dA)$ **Faire**
  Si *l'orbite* $< \alpha_0, \alpha_1 > (b)$ *n'est pas fermée* **Alors**
    Soit c et d les deux brins de l'orbite $< \alpha_0, \alpha_1 > (b)$ tels que $\alpha_1(c) = c$ et $\alpha_1(d) = d$;
    insérerArête(*c, d*);
  FinSi
FinPour

// On clôt les faces non fermées de l'orbite sommet de dB en liant les brins libres par α_1.
Pour chaque *Brin b* $\in < \alpha_1, \alpha_2, \alpha_3 > (dB)$ **Faire**
  Si *l'orbite* $< \alpha_0, \alpha_1 > (b)$ *n'est pas fermée* **Alors**
    Soit c et d les deux brins de l'orbite $< \alpha_0, \alpha_1 > (b)$ tels que $\alpha_1(c) = c$ et $\alpha_1(d) = d$;
    coudre$_1$(*c, d*);
  FinSi
FinPour

// On coud par α_2 les brins des arêtes libres par α_2 (Figure 3.9d).
// On reconstruit d'abord les involutions $\alpha_{1212}(i) = i$ possibles.
Pour chaque *Brin b* $\in < \alpha_1, \alpha_2, \alpha_3 > (dA)$ **Faire**
  Si $\alpha_1(b)$ *et* $\alpha_{21}(b)$ *sont libres par* α_2 **Alors**
    coudre$_2$($\alpha_1(b)$, $\alpha_{21}(b)$);
  FinSi
FinPour

// On complète ensuite l'orbite $\in < \alpha_1, \alpha_2 > (i)$ (Figure 3.9e).
Pour chaque *Brin b* $\in < \alpha_1, \alpha_2, \alpha_3 > (dA)$ **Faire**
  Si *b est libre par* α_2 **Alors**
    Il existe un Brin d $\in < \alpha_1, \alpha_2 > (b)$ libre par α_2;
    coudre$_2$(*b,d*);
  FinSi
FinPour

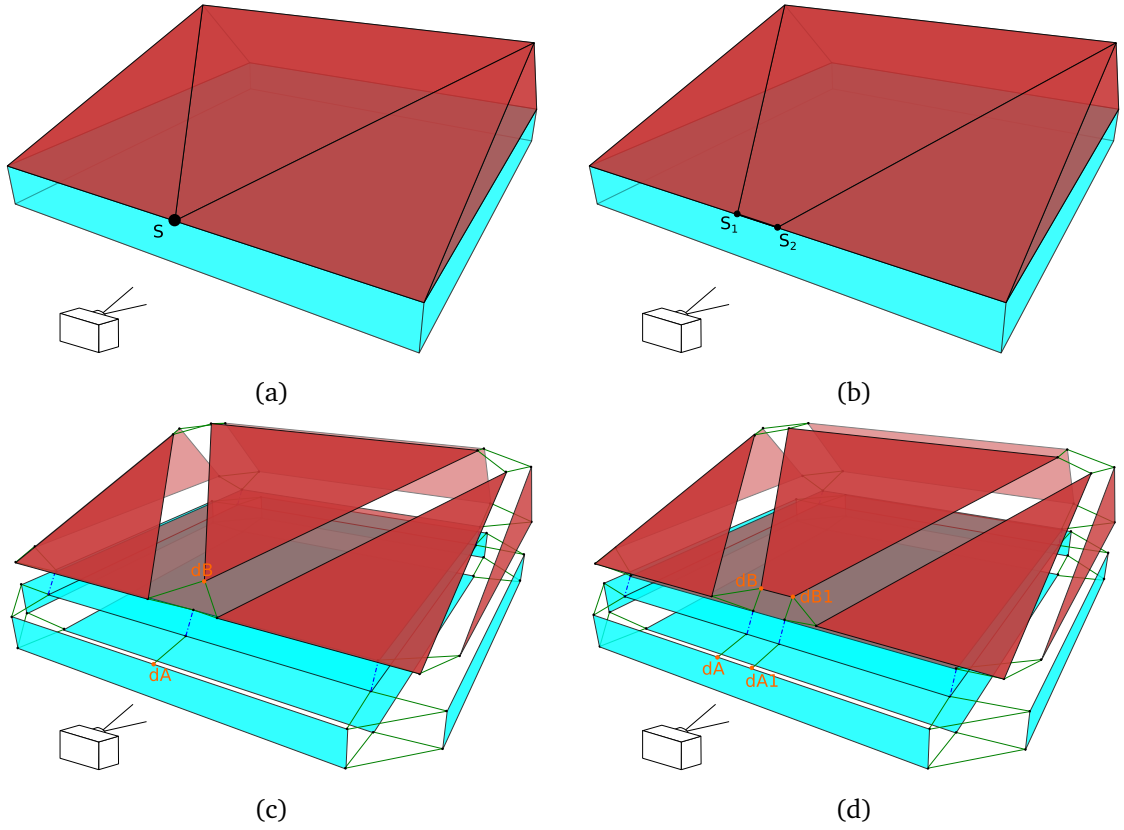


FIGURE 3.8 – Exemple d'éclatement d'un sommet S en arête en utilisant l'algorithme 3.1. (a et c) le sommet S auquel appartiennent dA et dB est éclaté en arête (b et d).

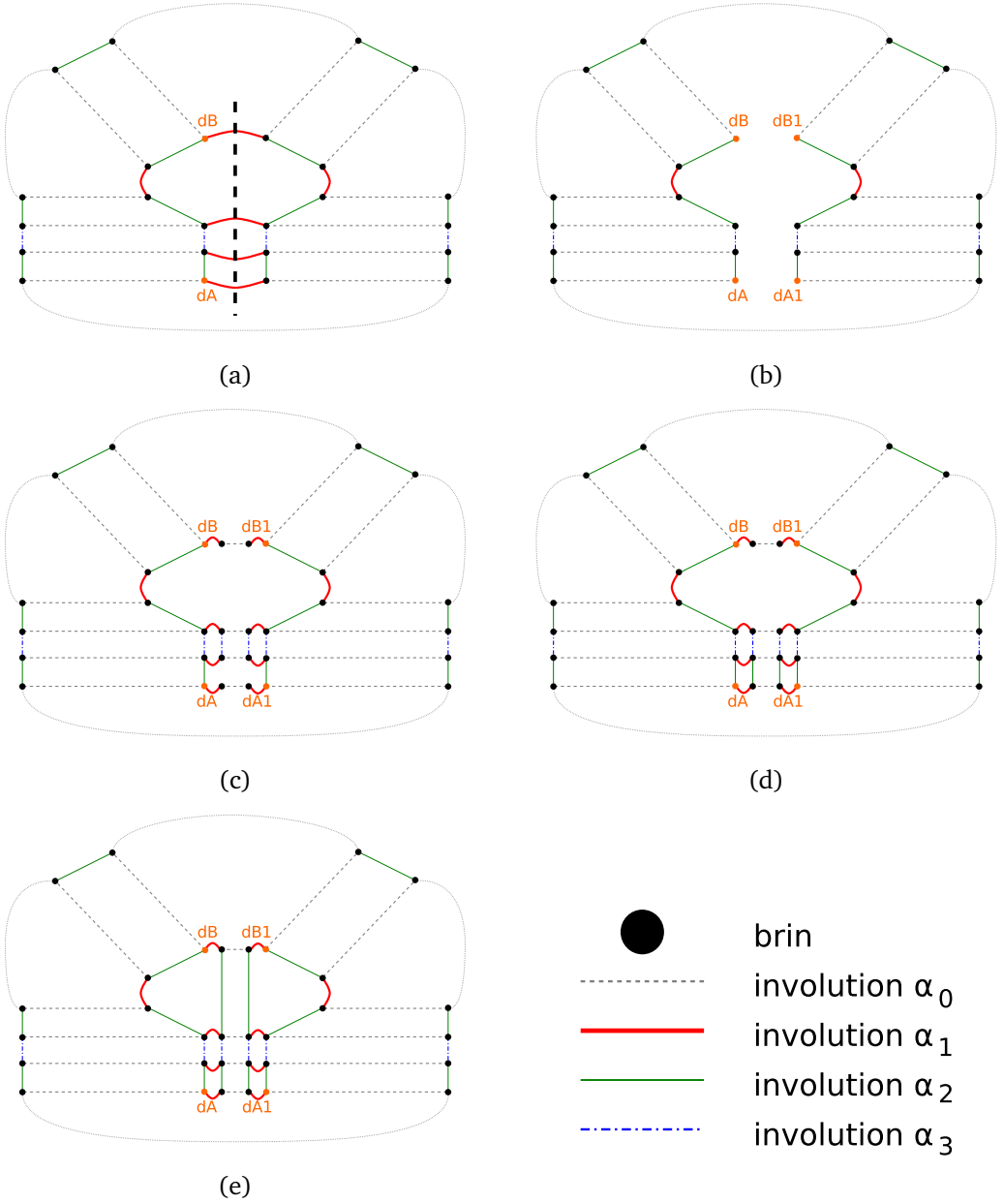


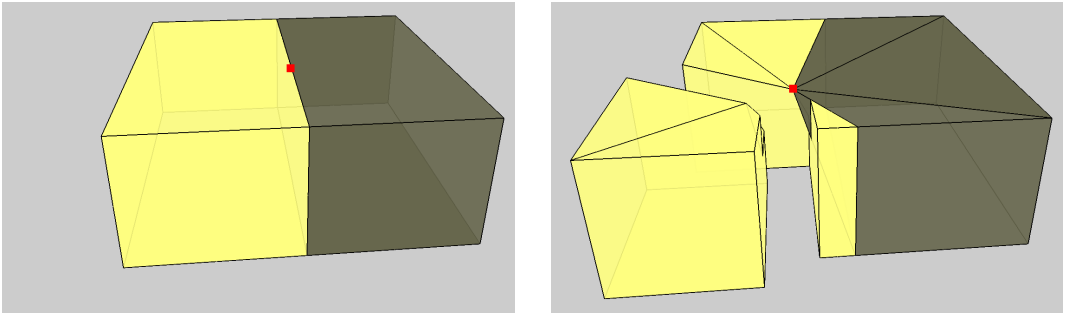
FIGURE 3.9 – Schémas descriptifs de l'algorithme 3.1.

3.2.2 Opération de Rosace : duplication d'un sommet appartenant à plusieurs volumes

Dans le contexte de structure en couches géologiques que nous avons développé dans l'introduction de cette partie, une contrainte vient s'ajouter à la géométrie et la topologie des couches. En effet, le déplacement d'un sommet ne doit impliquer qu'un gain ou qu'une perte de volume. Un pré-traitement est alors nécessaire lorsque le sommet en mouvement appartient à plusieurs volumes, afin de différencier son comportement selon chaque volume auquel il appartient. Nous rappelons que seuls les sommets appartenant à la surface sont en mouvement, cela implique que le sommet traité possède des brins libres par α_3. Pour chaque volume, le sommet est dupliqué sur une face libre par α_3 du volume, incidente au sommet et adjacente à un horizon (la face a une arête commune avec une face non libre par α_3). Chaque sommet dupliqué peut ainsi avoir son propre déplacement alors que le sommet d'origine permet de faire la jonction entre les volumes le long de l'arête commune. Les sommets générés lors de cette opération sont ajoutés à la file d'attente avec leur propre déplacement.

Plaçons-nous dans la configuration initiale décrite par la Figure 3.10a avec un sommet, représenté par un carré rouge, appartenant à deux volumes différents. Déplacer le sommet peut aller à l'encontre des contraintes contextuelles (Figure 3.10b) : l'objet marron à droite ne doit pas gagner de volume alors que l'objet jaune à gauche en perd. Pour résoudre ce problème, nous avons choisi de diviser le sommet en mouvement de la manière suivante :

- un sommet appartenant à l'horizon qui est déplacé sur celui-ci ;
- un nouveau sommet pour chaque objet adjacent. Le nouveau sommet est introduit dans la liste des événements.



(a) (b)

FIGURE 3.10 – Incohérence due au déplacement d'un sommet (carré rouge) appartenant à deux volumes.

La figure 3.12a représente quatre objets liés par une arête et dont le sommet supérieur est en mouvement vers le bas, impliquant une perte de volume. Les figures 3.12b, 3.12c, 3.12d, 3.12e montrent le résultat de la méthode appliquée respectivement sur le premier, le deuxième, le troisième et le quatrième objet. Le déplacement du sommet sur l'horizon entraîne une perte de volume égale pour tous les objets adjacents au sommet. Le sommet créé sur chaque objet adjacent permet soit de compenser la perte de volume de l'objet, soit de l'accentuer. Plus la position du sommet créé est proche de celle du sommet d'origine, moins la perte de volume est importante. Ainsi en déplaçant différemment les nouveaux

sommets, on obtient des pertes de volume différentes par objet, comme l'illustre la figure 3.12f. Les nouveaux sommets des troisième et quatrième objets (en gris et orange) sont proches de la position d'origine du sommet, ces deux objets perdent donc peu de volume. À l'inverse le sommet du deuxième objet (en jaune) est déplacé à l'opposé du sommet d'origine, la quantité de volume perdue est plus importante.

Afin de respecter la topologie, nous créons des faces reliant chaque nouveau sommet à son objet. Pour cela, nous parcourons chaque brin de l'orbite du sommet, pour chaque nouvel objet nous appliquons la méthode décrite par l'algorithme 3.2.Ces nouvelles faces sont triangulées en insérant une arête entre le sommet d'origine et son dupliqué. En effet, le dupliqué est voué à être déplacé et la face a de grande chance de ne plus être plane. Nous avons donc préféré effectuer une triangulation préventive. Les brins dA et dB passés en argument à l'algorithme appartiennent à l'orbite du sommet commun et au même volume, ils sont non libres par α_3. $\alpha_2(dA)$ et $\alpha_2(dB)$ sont eux libres par α_3. Cette opération fonctionne correctement pour n'importe quel sommet appartenant à la surface et appartenant également à un horizon.

Algorithme 3.2 : TraitementVolume(Brin d, Brin d1)

Entrée : Un Brin *dA* appartenant à l'orbite du sommet, non libre par α_3 et dont $\alpha_2(dA)$ est libre par α_3.

Entrée : Un Brin *dB* appartenant à l'orbite du sommet et au même volume que *dA*, non libre par α_3 et dont $\alpha_2(dB)$ est libre par α_3.

```
// On conserve un lien vers les α2 des brins dA et dB. Figure 3.11b
```
Brin dA2 ← α_2(dA);
Brin dB2 ← α_2(dB);

```
// On découd par α2 dA et dB. On suppose que l'opération découd
   également les α0 de dA et dB. Figure 3.11c
```
découdre$_2$(*dA*);
découdre$_2$(*dB*);

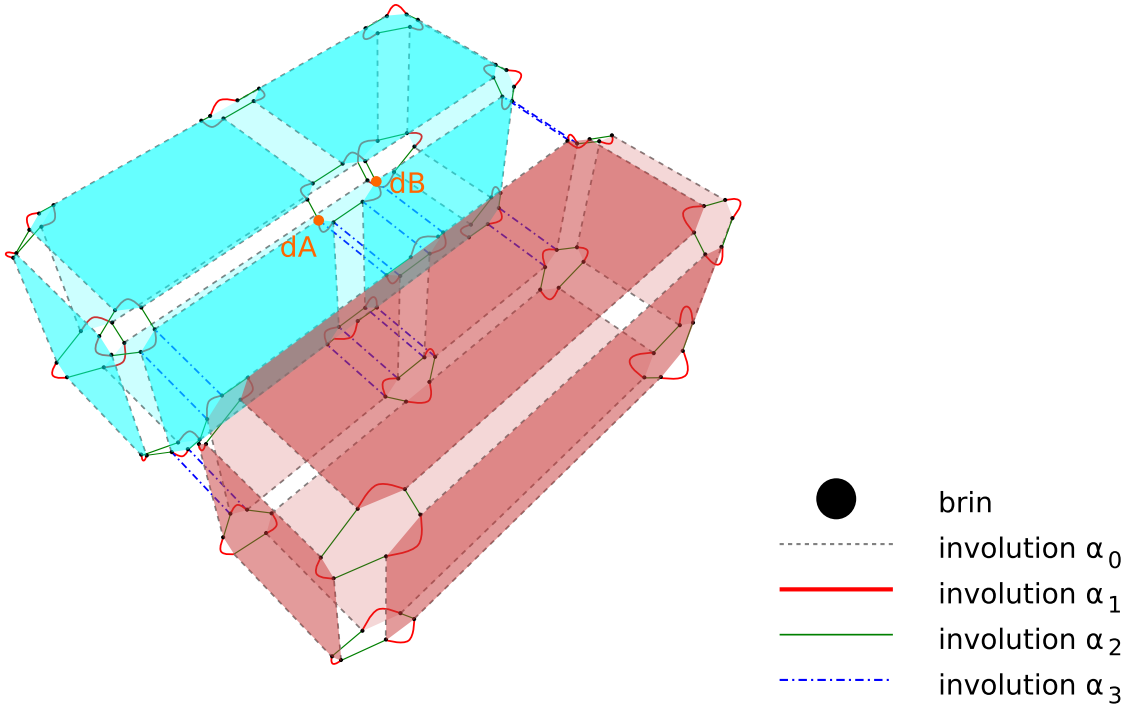
```
// On duplique les brins des arêtes auxquelles appartiennent dA, dB,
   dA2 et dB2. Les brins liés par α0 et la liaison sont également
   dupliqués. Figure 3.11d
```
Brin ddA ← dupliquer$_0$(*dA*);
Brin ddB ← dupliquer$_0$(*dB*);
Brin ddA2 ← dupliquer$_0$(*dA2*);
Brin ddB2 ← dupliquer$_0$(*dB2*);

```
// On coud par α1 les nouvelles arêtes entre elles, et par α2 les
   brins dupliqués avec leur originel. On suppose que l'opération de
   couture par α2 coud également les α0 des brins considérés. Figure
   3.11e
```
coudre$_1$(*ddA, ddB*);
coudre$_1$(*ddA2, ddB2*);
coudre$_1$($\alpha_0(ddA)$, $\alpha_0(ddA2)$);
coudre$_1$($\alpha_0(ddB)$,$\alpha_0(ddB2)$);

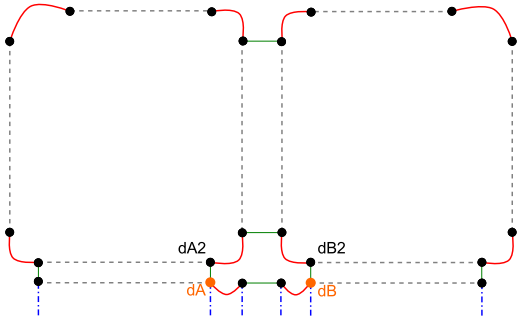
coudre$_2$(*dA, ddA*);
coudre$_2$(*dB, ddB*);
coudre$_2$(*dA2, ddA2*);
coudre$_2$(*dB2, ddB2*);

```
// On triangule la face en insérant une arête.
```
insérerArête(*ddA, ddB2*);

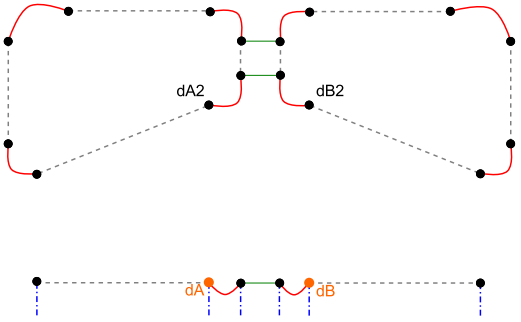
```
// On gère le déplacement du « nouveau » sommet auquel appartiennent
   dA2, dB2, ddA2 et ddB2 en l'ajoutant à la pile d'événements.
```



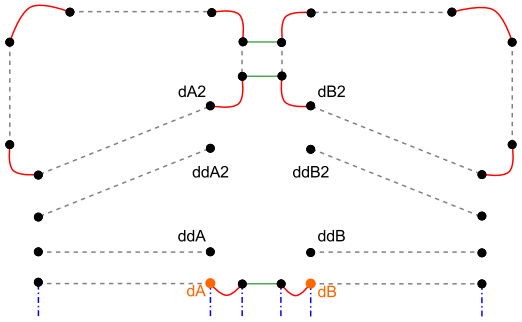
(a)



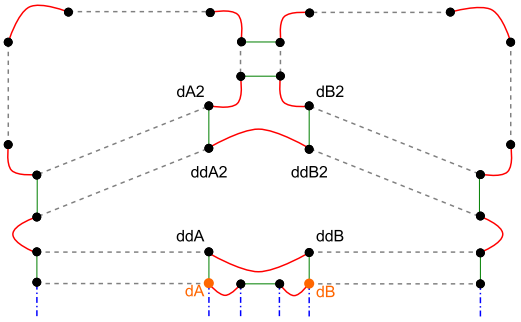
(b)



(c)

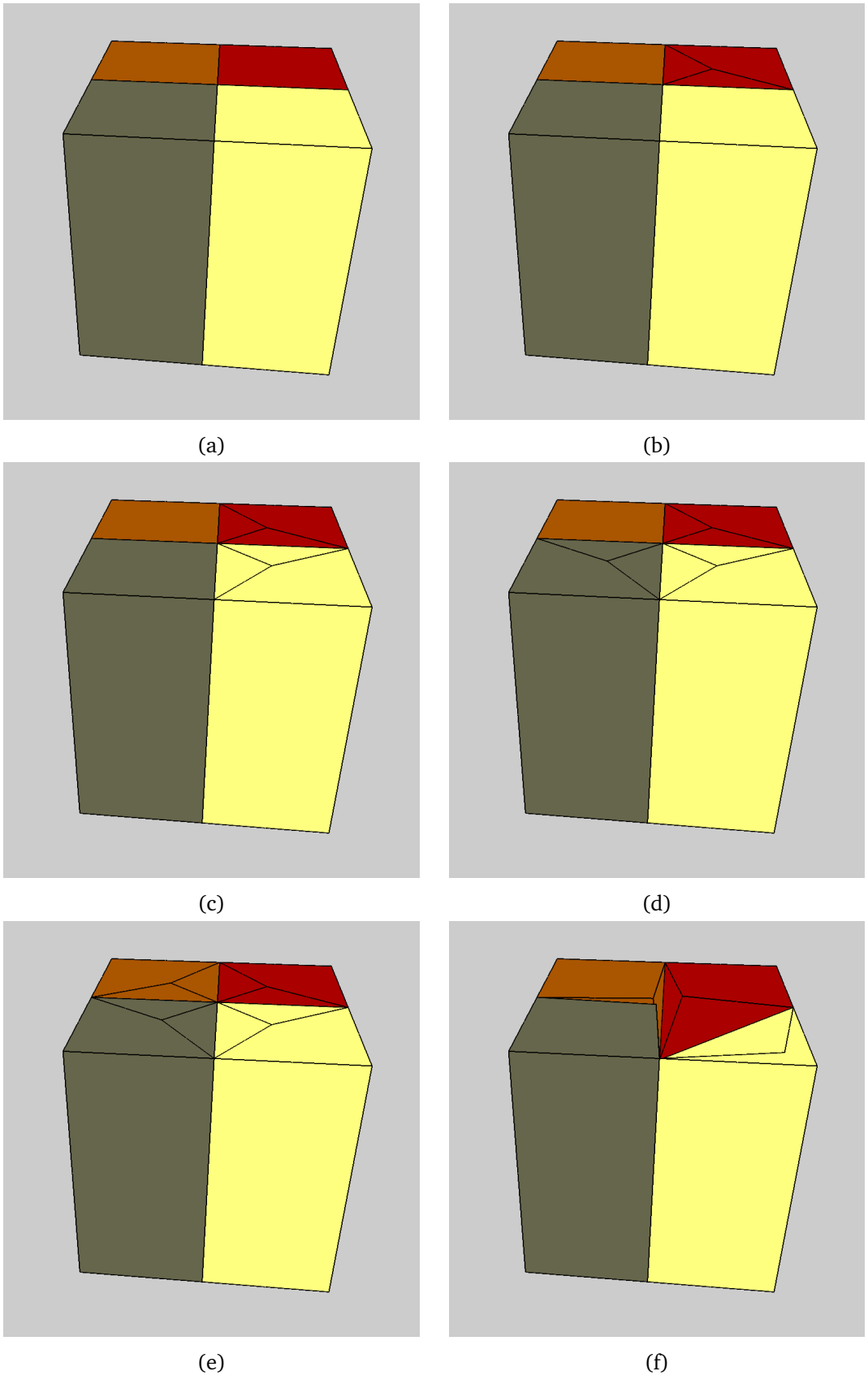


(d)



(e)

FIGURE 3.11 – Schémas descriptifs de l'algorithme 3.2



(a) (b) (c) (d) (e) (f)

FIGURE 3.12 – Exemple d'utilisation de l'algorithme 3.2 sur 4 volumes liés par une arête.

3.2.3 Création de volume

Le déplacement d'un sommet peut induire le gain de volume, mais également la création d'un volume. Nous avons alors construit trois opérations de création de volumes : à partir d'un sommet, à partir d'une face et à partir d'un ensemble connexe de faces. Pour créer un volume à partir d'un sommet, nous utilisons une opération de chanfreinage analogue au traitement effectué lors de la collision sommet-face (voir algorithme 3.5). La création d'un volume à partir d'une face peut se faire de deux manières : à l'aide de l'opération de cône vue dans la partie 1.3.3, ou bien par extrusion de la face. Nous avons également étendu la création de volume à un ensemble de faces connexes. Pour cela, nous appliquons un traitement identique à celui de la création de cône. Nous dupliquons l'ensemble des brins des faces connexes que nous lions par α_3 aux brins originaux. Nous dupliquons une seconde fois ces brins, mais nous ne lions cette fois que les bords par α_2. Nous obtenons ainsi un volume qui conserve même la topologie que l'ensemble d'origine.

3.3 Opérations topologiques événementielles

Afin de gérer les incohérences, nous avons élaboré un ensemble d'opérations topologiques. Ces opérations sont basées sur les opérations topologiques de base que nous avons détaillées dans la partie 1.3 et sur les opérations topologiques non-événementielles.

3.3.1 Intersection Sommet-Sommet

Parmi les 4 cas de collision sommet-sommet, nous avons choisi d'effectuer 3 traitements différents. Nous appelons A et B les deux sommets et `dA` et `dB` un brin respectivement de A et de B. Dans tous les cas, le plongement du sommet final est celui de B.

Le cas le plus simple est lorsque les deux sommets sont liés par une arête. Nous rappelons que dans ce cas, `dA` appartient à l'orbite $< \alpha_0, \alpha_2, \alpha_3 >$ (`dB`). Nous appliquons alors une opération de contraction sur l'arête en question.

Dans le second cas où les deux sommets appartiennent à la même orbite face, nous divisons la face en deux. Nous utilisons l'algorithme 3.3 afin de traiter ce pincement de face (voir Figure 3.13).

Ces opérations peuvent produire des faces dégénérées composées de seulement 4 brins. Nous marquons les brins de ces faces pour les supprimer par contraction de faces ensuite.

Dans le troisième et quatrième cas, les deux sommets ne sont pas liés par une arête et n'appartiennent pas à une même face. La différence réside dans leur appartenance ou non au même volume. Nous effectuons le même traitement dans les deux cas, et gérons cette distinction ensuite. Pour cela, nous chanfreinons A et B afin d'obtenir deux faces, respectivement F_A et F_B, que nous cousons par α_3 (Figure 3.14c). Si A et B appartiennent au même volume, nous supprimons la face à l'aide de l'opération de fusion de volume (Figure 3.14d). Nous ne pouvons coudre F_A et F_B que si les deux faces sont isomorphes.

Algorithme 3.3 : SommetSommetFace(Brin `dA`, Brin `dB`)

Entrée : Un Brin `dA` appartenant à l'orbite du sommet A et un brin `dB` appartenant à l'orbite du sommet B (voir Figure 3.13).

Brin dA1 $\leftarrow \alpha_1$(dA);
Brin dB1 $\leftarrow \alpha_1$(dB);
`// On découd par` α_1 `dA et dB. Figure 3.13b`
découdre$_1$(*dA*);
découdre$_1$(*dB*);
`// On coud par` α_1 `dA avec dA1 ou dB1 et dB avec dB1 ou dB1 en évitant de créer une face croisée. Figure 3.13c`
Si $dA \in < \alpha_0, \alpha_1 > (dB)$ **Alors**
  coudre$_1$(*dA, dB*);
  coudre$_1$(*dA1, dB1*);
Sinon
  coudre$_1$(*dA, dB1*);
  coudre$_1$(*dA1, dB*);
FinSi

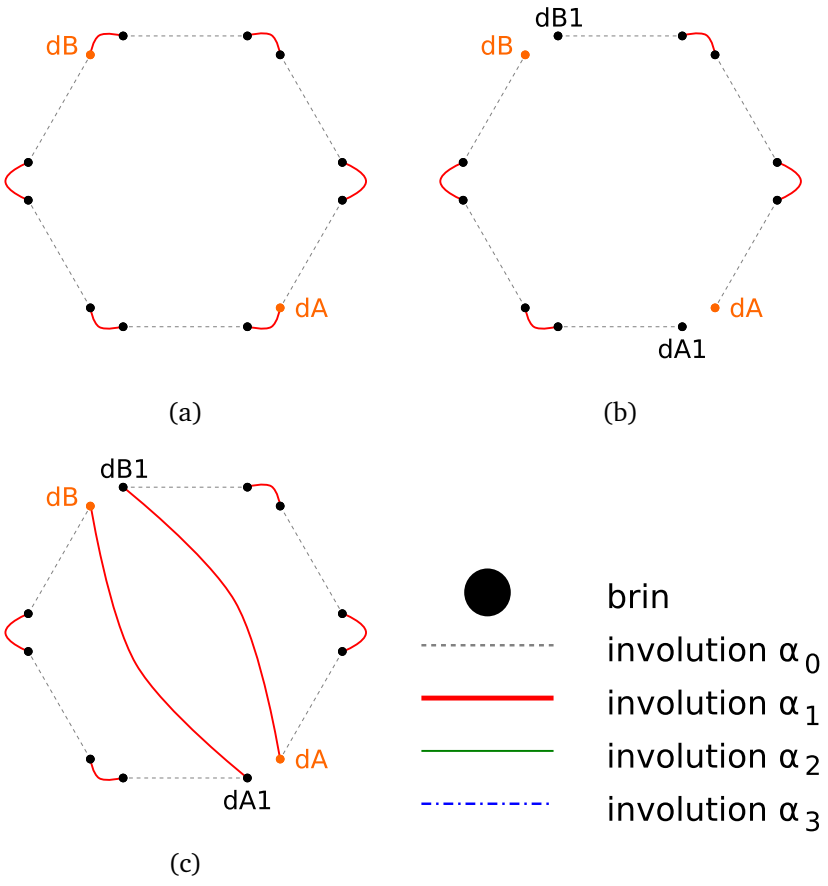


FIGURE 3.13 – Schémas descriptifs de l'algorithme 3.3.

Dans le cas contraire, nous insérons le nombre de sommets nécessaire sur la face qui en possède le moins (voir algorithme 3.4). Cette méthode ne fonctionne pas dans le cas de face à trou par exemple, mais elle est suffisante dans notre contexte. Nous déterminons les brins à coudre ensemble par α_3 en nous basant sur la géométrie des faces : F_A et F_B sont coplanaires, nous cherchons un couple de brins dFA et dFB appartenant respectivement à un sommet de chacune des faces dont la distance les séparant est minimale. dFA et dFB doivent être orientés de manière à ce que la face obtenue après couture ne soit pas croisée.

Algorithme 3.4 : SommetSommetVolume(Brin `dA`, Brin `dB`)

Entrée : Un Brin `dA` appartenant à l'orbite d'un sommet et un brin `dB` appartenant à l'orbite d'un autre sommet.

`// Chanfreinage respectivement du sommet auquel appartient dA et du sommet auquel appartient dB. (Figure 3.14b).`

Brin fA ← `chanfreinage`(*dA*);

Brin fB ← `chanfreinage`(*dB*);

Si *l'orbite face de fB possède plus de sommets que l'orbite face de fA* **Alors**

  Tant que *l'orbite face de fB possède plus de sommets que l'orbite face de fA* **Faire**

    `InsertionSommet`(fA)

  FinTq

Sinon si *l'orbite face de fA possède plus de sommets que l'orbite face de fB* **Alors**

  Tant que *l'orbite face de fA possède plus de sommets que l'orbite face de fB* **Faire**

    `InsertionSommet`(fB)

  FinTq

FinSi

`// On coud par` α_3 `les deux faces obtenues. (Figure 3.14c). Les brins` dFA `et` dFB `sont déterminés en se basant sur la géométrie.`

`coudre`$_3$(*dFA,dFB*)

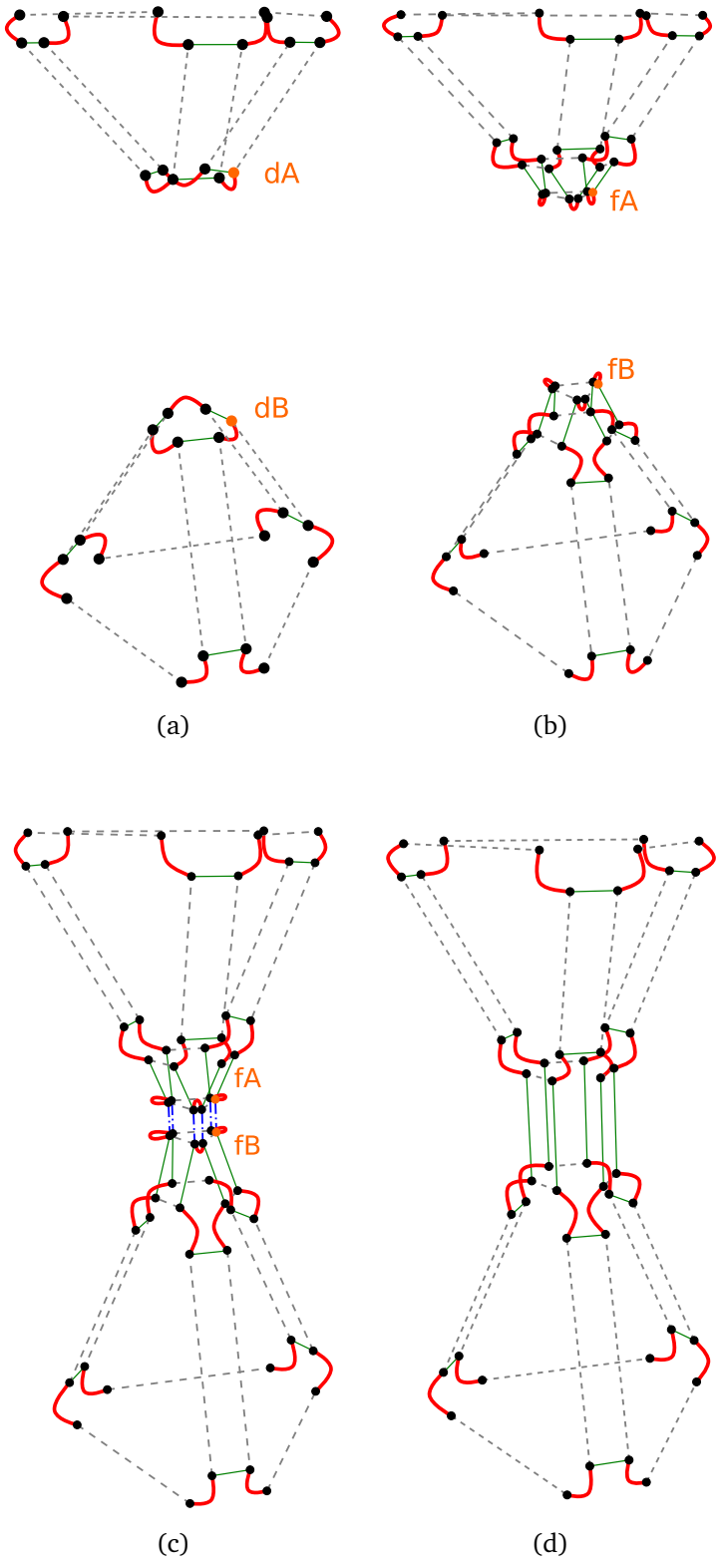


FIGURE 3.14 – Schémas descriptifs de l'algorithme 3.4.

3.3.2 Intersection Sommet-Arête

L'incohérence générée par la collision entre un sommet en mouvement et une arête peut être ramenée à une collision sommet-sommet en insérant un sommet sur l'arête avec pour plongement le point d'intersection. Nous réutilisons donc ici l'opération précédente.

3.3.3 Intersection Sommet-Face

L'opération intervient lorsqu'un sommet S d'un objet entre en collision avec une face F, qu'il n'appartient pas à cette face et qu'il n'est relié à elle par aucune arête. Le traitement s'effectue en deux parties : nous effectuons un chanfreinage du sommet en mouvement à l'aide de l'algorithme 3.5. Cette opération permet de créer un volume correspondant à la partie supérieure de la pyramide formée par le sommet S et ses faces incidentes. Ce volume est lié par α_3 à la face issue de l'objet de départ. Ensuite, pour que la face intersectée soit trouée, nous devons relier celle-ci à la nouvelle face créée par une arête tout en conservant l'orientation de la G-carte. Sur la figure 3.15e, nous avons inséré une arête A entre un brin, dA, de la face intersectée et un brin dB appartenant à la nouvelle face créée par l'algorithme. Cette arête n'a d'intérêt que topologiquement, nous la marquons pour pouvoir la considérer en tant que telle. La figure 3.16 illustre le traitement d'un sommet S entrant en collision avec une face F.

Algorithme 3.5 : TraitementDuSommet(Brin d)

Entrée : Un Brin *d* appartenant à l'orbite du sommet S en mouvement

```
// On insère un sommet sur chaque arête incidente au sommet S. Figure
   3.15b
```

Pour chaque *Brin b appartenant à une arête incidente au sommet S* **Faire**

```
    insérerSommet(b);
```

FinPour

```
// On insère une arête entre chaque nouveau sommet créé. Figure 3.15c
```

Pour *b1 et b2 deux brins appartenant à l'orbite de deux nouveaux sommets sur deux arêtes consécutives* **Faire**

```
    insérerArête(b1, b2);
```

FinPour

```
// On crée une face entre les nouvelles arêtes. Figure 3.15d
```

Si la face F est un horizon entre deux objets (ses brins ne sont pas libres par α_3), nous appelons *objet supérieur* l'objet auquel appartient S, et *objet inférieur* l'objet relié au précédent par la face F. Dans ce cas, nous conservons les faces incidentes à S : le déplacement de S provoque alors une perte de volume de l'*objet inférieur*, nous le réintroduisons dans la file d'événements. Si la face F n'est pas un horizon, nous supprimons les faces incidentes à S.

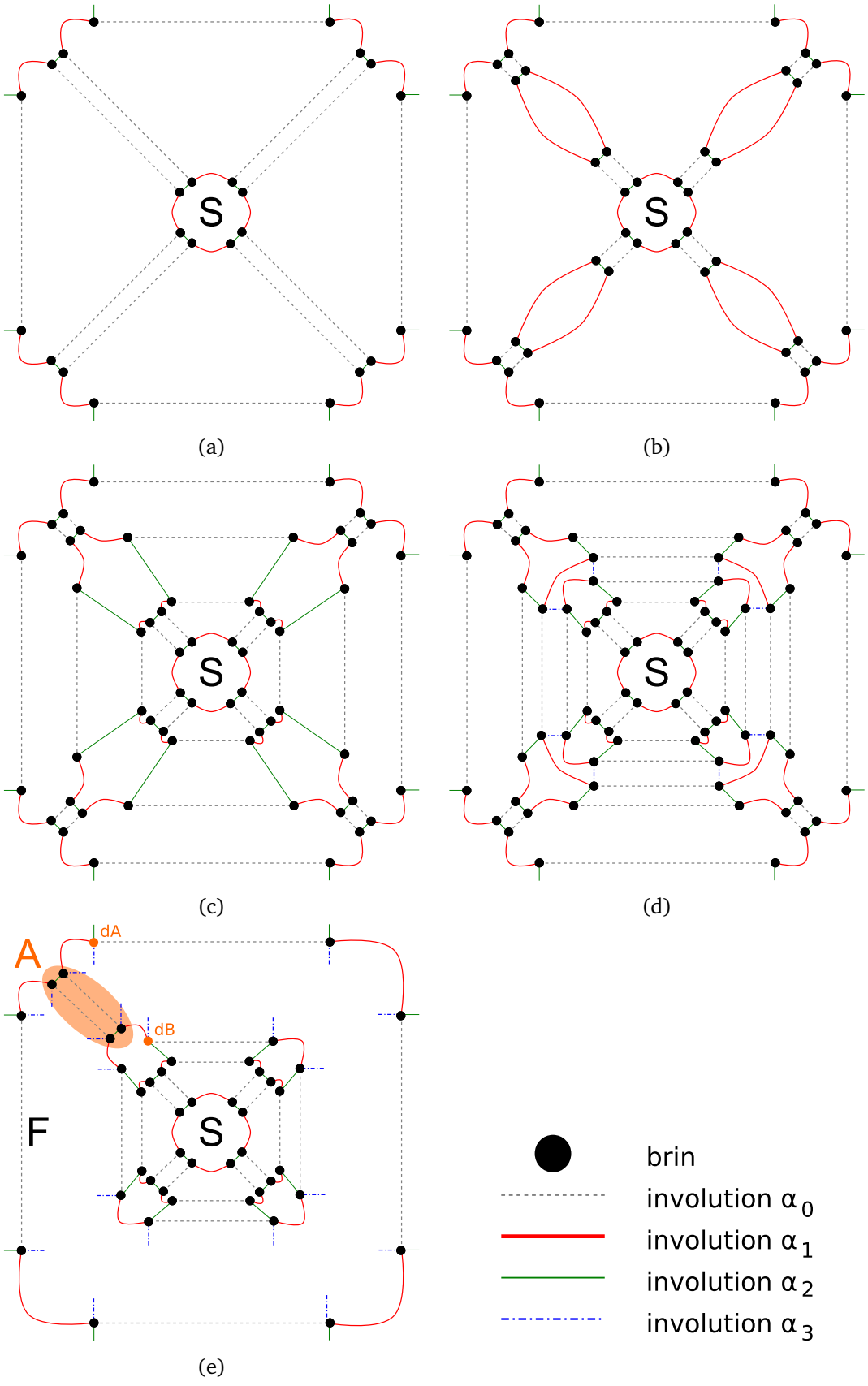
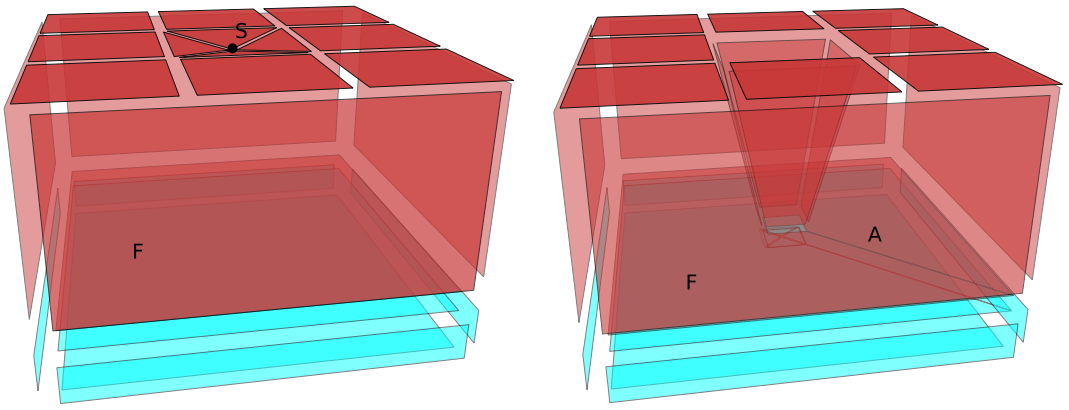


FIGURE 3.15 – Schémas descriptifs de l'algorithme 3.5



(a) (b)

FIGURE 3.16 – (a) Le sommet S entre en collision avec la face F, (b) Après traitement, la face F est trouée.

3.3.4 Intersection Arête-Sommet

L'opération Arête-Sommet est la même que l'opération Sommet-Arête, à cela prêt que le plongement final est celui du sommet intersecté.

3.3.5 Intersection Arête-Arête

L'intersection Arête-Arête utilise les opérations d'intersection sommet-sommet et arête-face. En effet, nous insérons un sommet sur les arêtes avec pour plongement les coordonnées de l'intersection et nous utilisons alors l'opération sommet-sommet. Ensuite, nous appliquons l'opération arête-face pour chaque demi-arête.

3.3.6 Intersection Arête-Face

Pour rappel, nous déplaçons les sommets de manière indépendante. Si les sommets d'une arête intersectent une face, nous traitons cet événement en deux fois : un événement de type sommet-face, puis un événement arête-face. Cela implique pour ce deuxième événement qu'une des extrémités de l'arête appartient à la face. L'opération topologique traite le cas d'un sommet S qui intersecte une face F et qui est lié à la face par une arête A. Nous qualifions cette opération de pincement arête-face.

L'opération permettant de traiter un pincement d'une face par une arête est décrite par l'algorithme 3.6. Le traitement prend en compte le cas de figure où l'extrémité de l'arête A liée à la face F appartient ou non à plusieurs volumes. Nous utilisons ici l'opération d'éclatement d'un sommet en arête (voir partie 3.1). La position de la caméra sur les figures 3.17 et 3.19 représente le point de vue utilisé pour les schémas 3.18. Le résultat de l'opération est illustré par la figure 3.17.

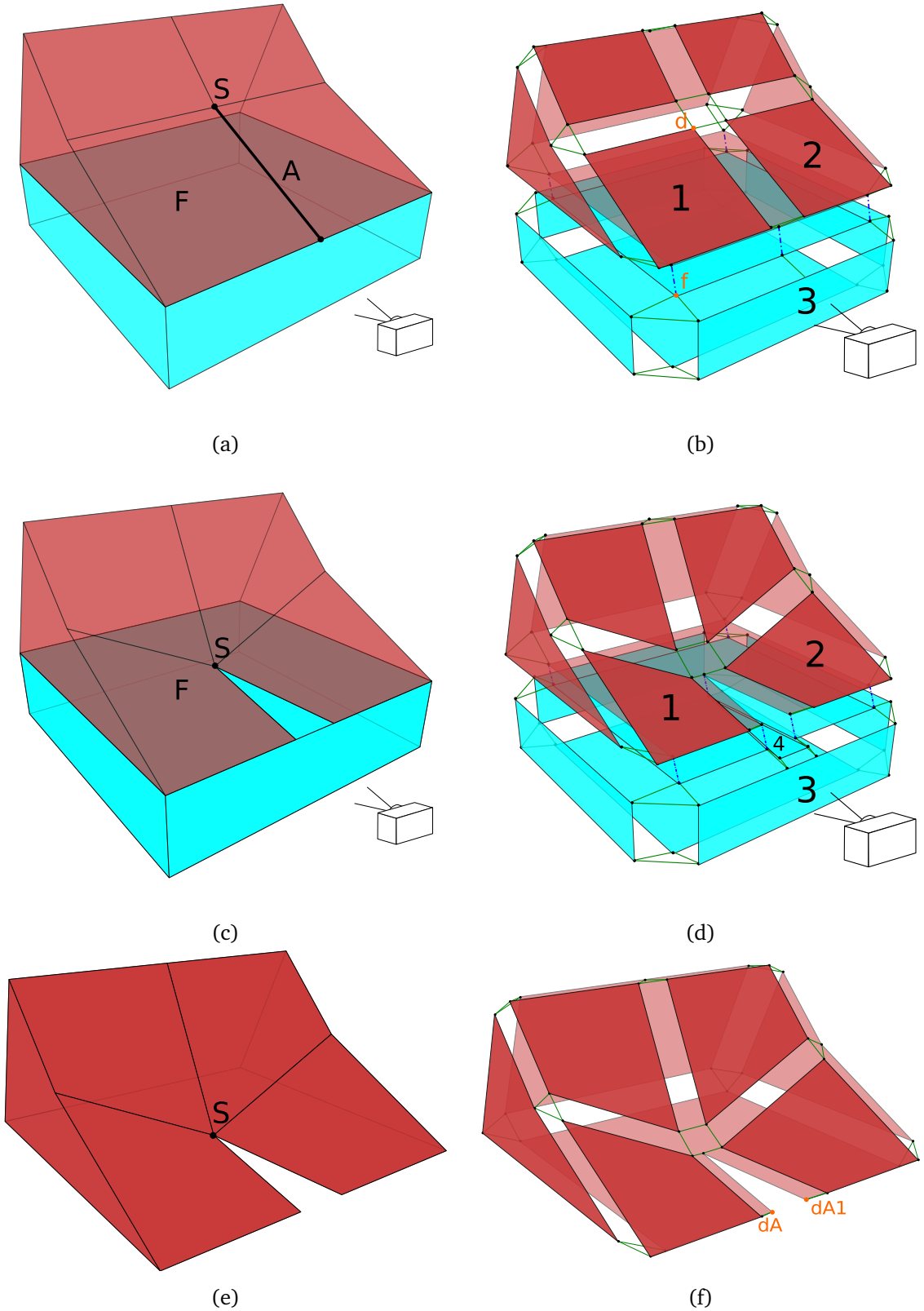


FIGURE 3.17 – Résultat de l'opération arête-face décrite par l'algorithme 3.6. (a) L'arête A intersecte la face d'horizon F entre les deux volumes, (c) Une face est créée et appartient au volume inférieur. (b) et (d) sont les vues éclatées de (a) et (c), les brins d et f sont ceux passés en argument de l'algorithme. Les figures (e) et (f) représentent le cas où la face intersectée n'est pas un horizon. Les faces numérotées correspondent à celles de la figure 3.18.

Algorithme 3.6 : PincementAreteFace(Brin d, Brin f)

Entrée : Un Brin *d* appartenant à l'orbite du sommet S en mouvement et à l'arête A qui le lie à la face F et un Brin *f* appartenant à l'orbite de la face F (Figures 3.18a et 3.19a).

```
// On éclate l'arête A en face (Figure 3.18b).
insérerFace(d);
// On détermine deux brins de la manière suivante (Figures 3.18c et
   3.19b):
```
Brin dF $\leftarrow < \alpha_1, \alpha_2 > (\alpha_0(d)) \cap < \alpha_0, \alpha_1, \alpha_3 > (f)$;
Brin dA $\leftarrow \alpha_{20}(d)$;

Si *dF est non libre par* α_3 **Alors**

Brin dB $\leftarrow \alpha_{32}(dF)$;

```
// On éclate le sommet auquel appartiennent les brins dA et dB en
   arête (Figures 3.18d et 3.19c).
ÉclatementSommetArête(dA,dB);
// On est maintenant dans le cas d'un pincement d'une face par une
   autre (voir section 3.3.8).
PincementFaceFace(dA, f);
// La figure 3.19d représente le résultat final de l'opération.
```

Sinon

Brin dA1$\leftarrow \alpha_1(dA)$;
Brin dF1$\leftarrow \alpha_1(dA)$;

```
// On découd par α1
découdre1(dF);
découdre1(dA);
// On coud par α1 les arêtes.
```

Si *dF* $\in < \alpha_1, \alpha_2, \alpha_3 >$*(dA)* **Alors**

```
coudre1(dF,dA);
coudre1(dF1,dA1);
```

Sinon

```
coudre1(dF,dA1);
coudre1(dF1,dA);
```

FinSi

```
// Les figures 3.17e et 3.17f représentent le résultat final de
   l'opération.
```

FinSi

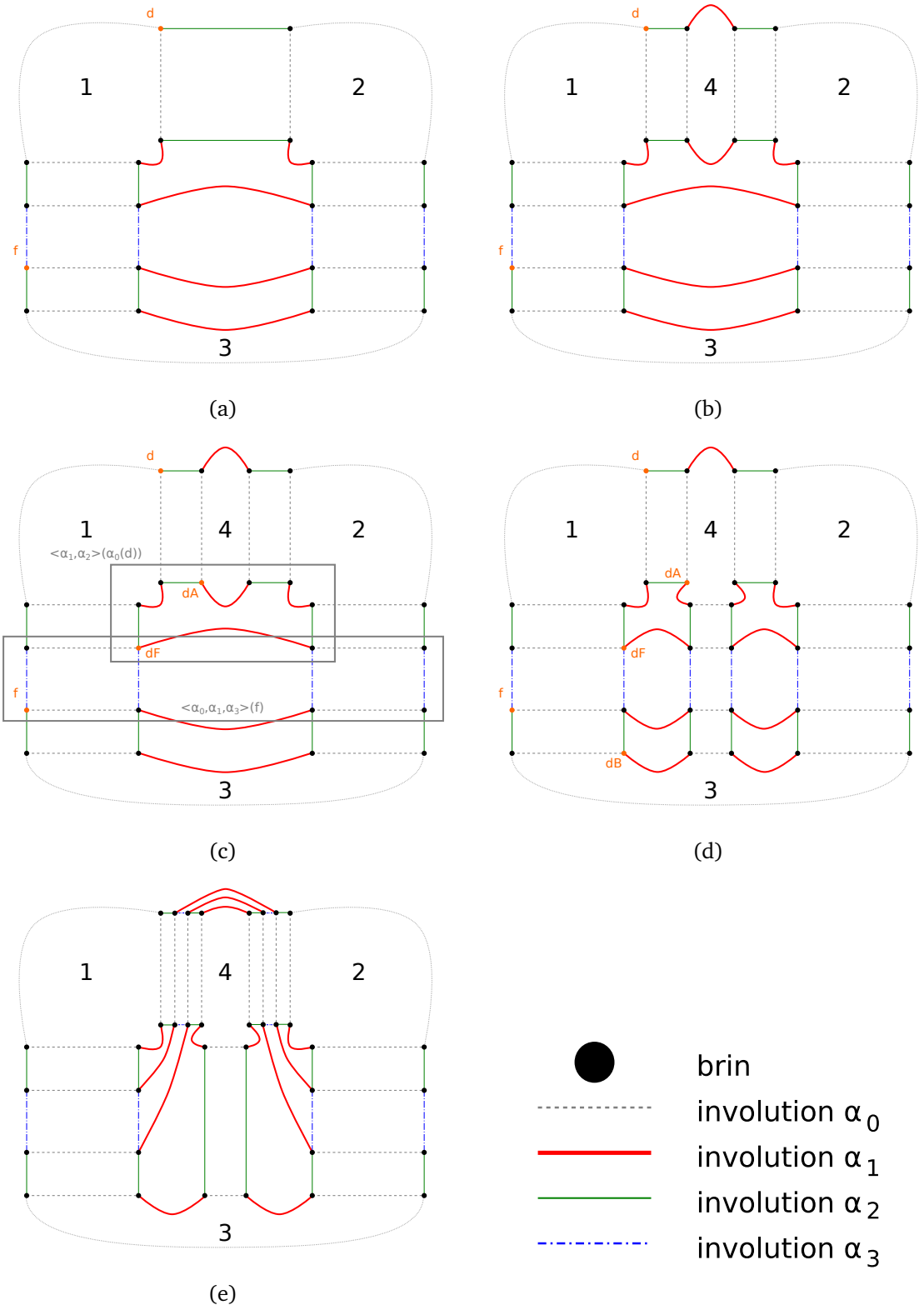
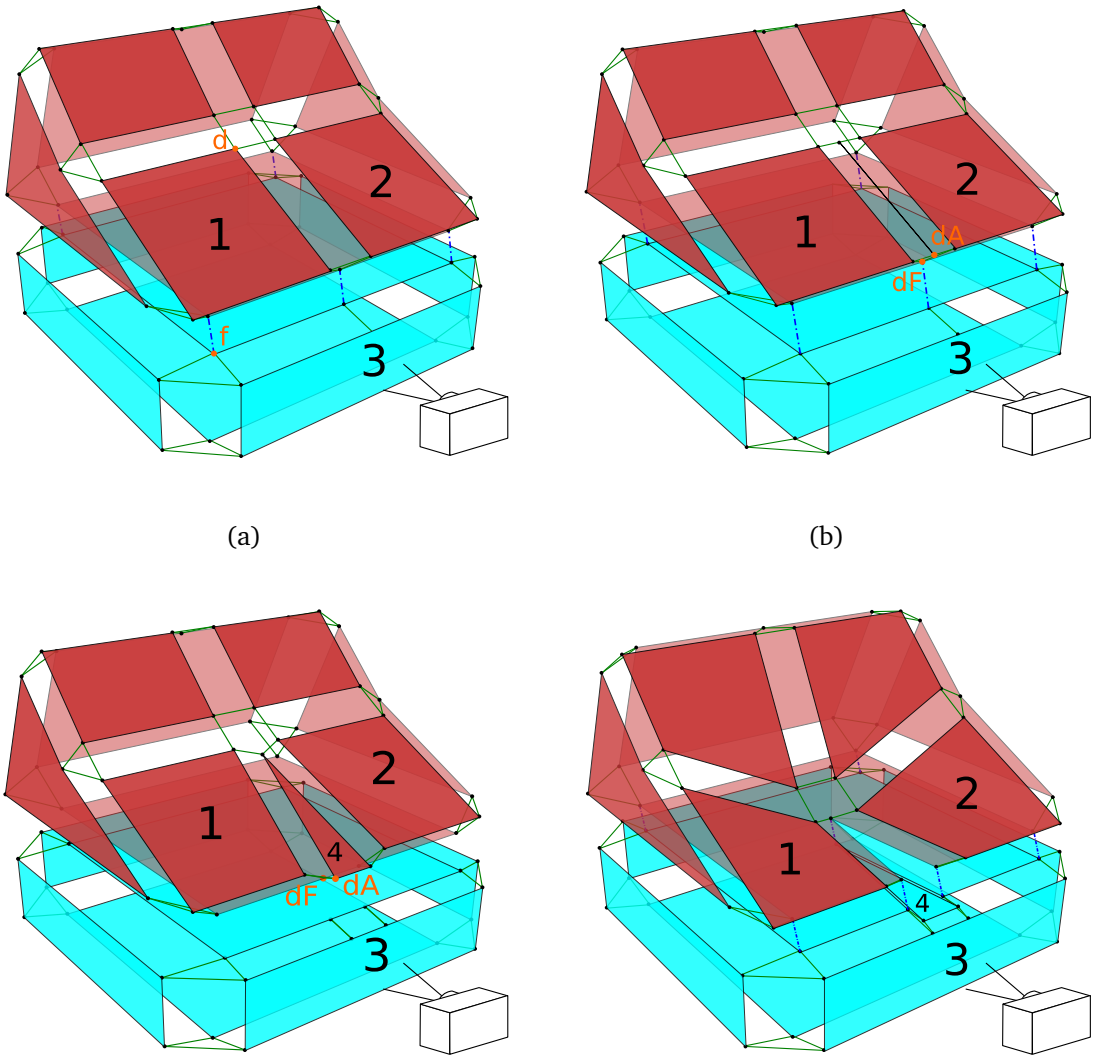


FIGURE 3.18 – Schémas descriptifs de l'algorithme 3.6. Les faces numérotées correspondent à celles de la figure 3.17.



(a) (b)

(c) (d)

FIGURE 3.19 – Schémas descriptifs de l'algorithme 3.6 en vue éclatée.

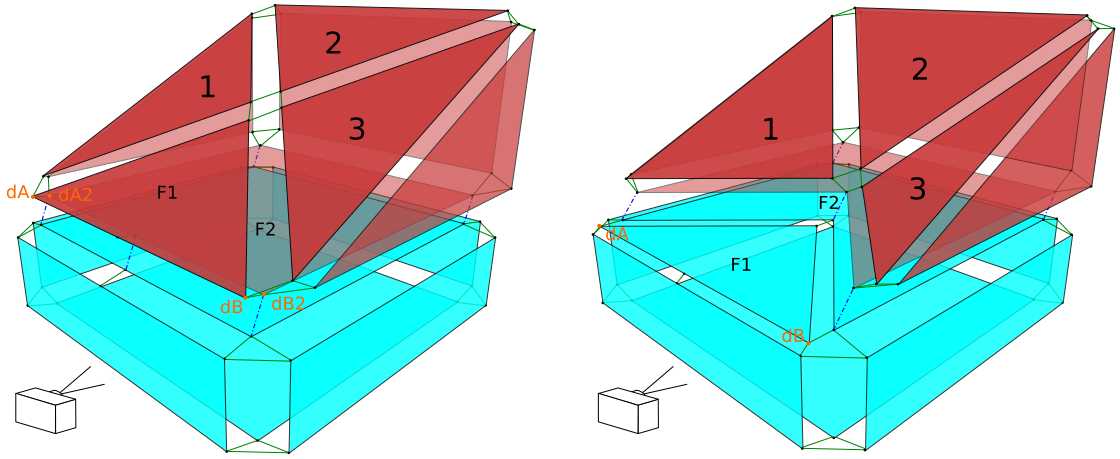
3.3.7 Intersection Face-Sommet et Intersection Face-Arête

Les opérations d'intersection de type face-sommet et face-arête sont respectivement les mêmes que les opérations sommet-face et arête-face.

3.3.8 Intersection Face-Face

Comme pour l'intersection Arête-Face, il s'agit d'un pincement : les deux faces sont liées par α_2 par au moins une arête. Le déplacement d'un sommet de la première face intersecte la seconde face. Nous appelons $F1$ la face à laquelle appartient le sommet en mouvement, $F2$ la face intersectée (voir Figure 3.20). Nous utilisons l'algorithme 3.7 pour gérer cette collision, illustré par les figures 3.21 et 3.22. La première étape de notre algorithme est de déterminer quels sont les brins de début (`dA`) et de fin (`dB`) de l'ensemble des arêtes qui lient $F1$ et $F2$. Nous partons de l'hypothèse qu'il existe obligatoirement un chemin allant de `dA` à `dB` en utilisant les involutions α_0 et α_1 dont tous les brins parcourus appartiennent à une arête liant $F1$ et $F2$. En d'autres termes, il s'agit des deux brins qui remplissent les conditions suivantes :

- le brin `b` appartient à l'orbite face de $F1$,
- $\alpha_2(b)$ appartient à l'orbite face de $F2$,
- $\alpha_{12}(b)$ n'appartient pas à l'orbite face de $F2$.



(a) (b)

FIGURE 3.20 – Cas du pincement face-face. (a) La face F1 intersecte la face d'horizon F2 entre les deux volumes. dA et dB appartiennent à F1, $dA2$ et $dB2$ appartiennent à F2. Ces 4 brins sont situés aux extrémités de l'ensemble des arêtes liant F1 et F2. (b) Résultat après le traitement face-face (Algorithme 3.7). Les faces numérotées correspondent à celles de la figure 3.21.

Algorithme 3.7 : PincementFaceFace(Brin dF1, Brin dF2)

Entrée : Un Brin `dF1` appartenant à l'orbite du la face $F1$ et un brin `dF2` appartenant à l'orbite du la face $F2$. (Figures 3.21a et 3.22a).

Brin dA ← extrémité de l'ensemble des arêtes liant $F1$ et $F2$;
Brin dB ← autre extrémité de l'ensemble des arêtes liant $F1$ et $F2$;
Brin dA2 ← α_2(dA);
Brin dB2 ← α_2(dB);
Brin dA21 ← α_1(dA2);
Brin dB21 ← α_1(dB2);
// On découd par α_1 l'ensemble des arêtes communes appartenant à la face $F2$. (Figure 3.21b).
découdre$_1$(*dA2*);
découdre$_1$(*dB2*);
// On « transfère » la face $F1$ au volume sous-jacent. (Figures 3.21c et 3.22b).
Brin temp ← α_1(dA);
Brin temp3 ← dA21;
Tant que *temp* $\neq \alpha_1$*(dB)* **Faire**
  Brin dupli ← dupliquer$_0$(*temp*);
  Brin dupli2 ← dupliquer$_0$(*temp*);
  Brin temp2 ← $\alpha_2(temp)$;
  découdre$_2$(*temp*);
  coudre$_2$(*temp, dupli*);
  coudre$_2$(*temp2, dupli2*);
  coudre$_3$(*temp, temp2*);
  coudre$_1$(*temp3, dupli2*);
  temp ← $\alpha_{01}(temp)$;
  temp3 ← $\alpha_0(dupli2)$;
FinTq
coudre$_1$(*temp3, dB21*);
// On lie $F1$ par α_2 aux faces voisines.(Figures 3.21d et 3.22c).
temp ← dA;
Tant que *temp* $\neq \alpha_1$*(dB)* **Faire**
  Brin temp2 ← $\alpha_{232}(temp)$;
  découdre$_2$(*temp*);
  découdre$_2$(*temp2*);
  coudre$_2$(*temp, temp2*);
  temp ← $\alpha_{01}(temp)$;
FinTq
// On supprime l'ensemble des arêtes liant $F1$ et $F2$ au départ. Il nous suffit de parcourir les brins de l'orbite $< \alpha_0, \alpha_1, \alpha_3 >$(dB2). (Figures 3.21e et 3.22d).

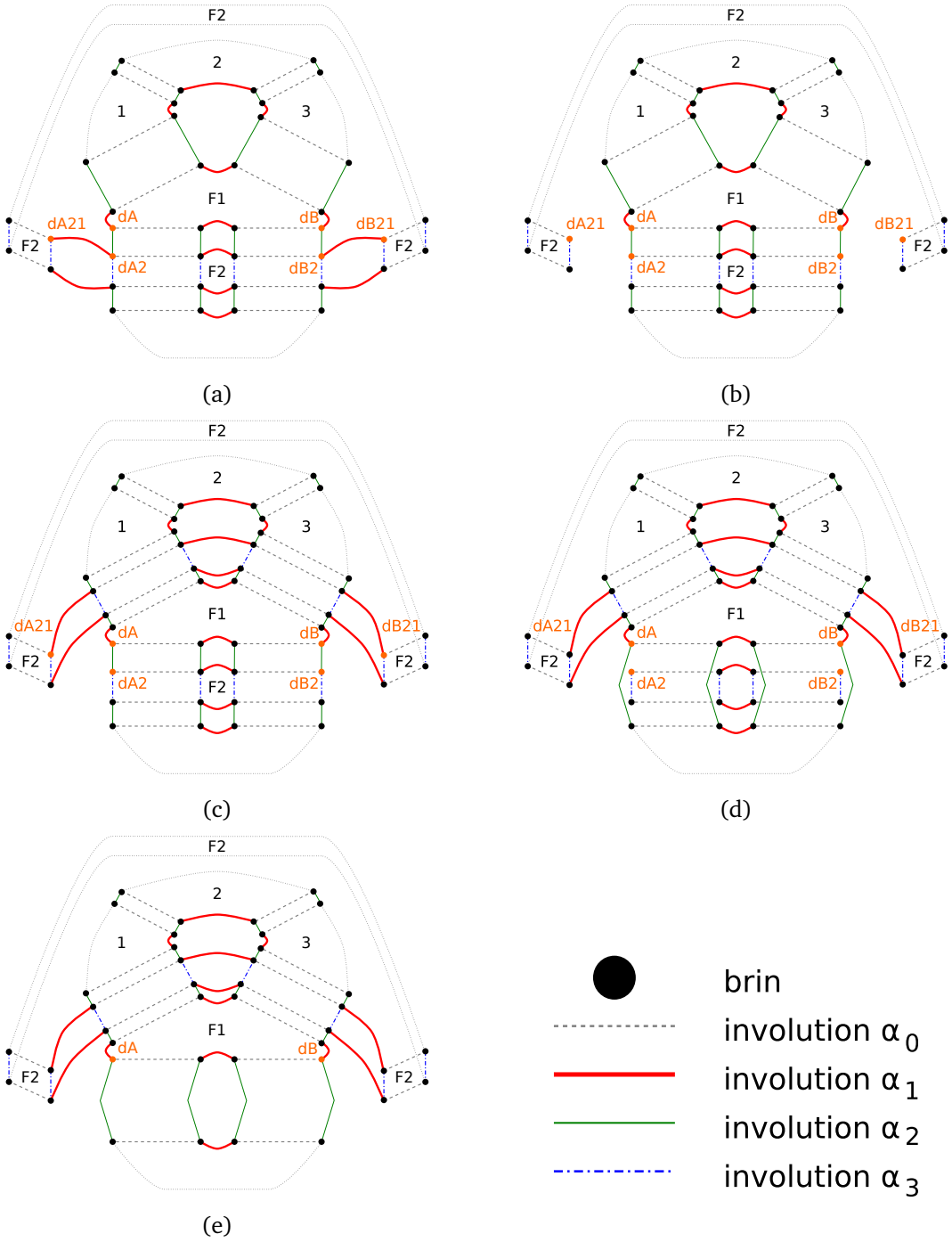


FIGURE 3.21 – Schémas descriptifs de l'algorithme 3.7. Les faces numérotées correspondent à celles de la figure 3.20.

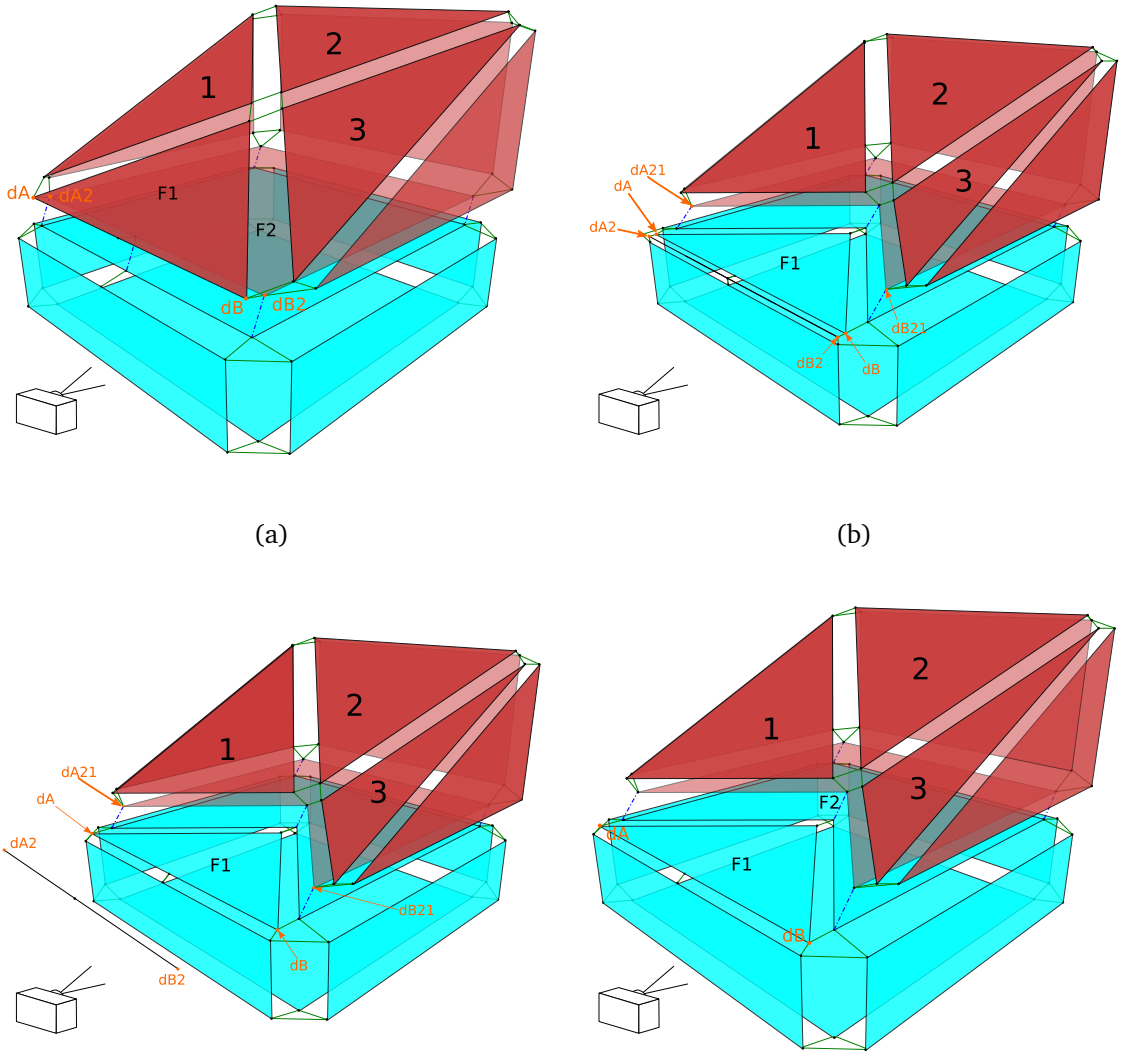
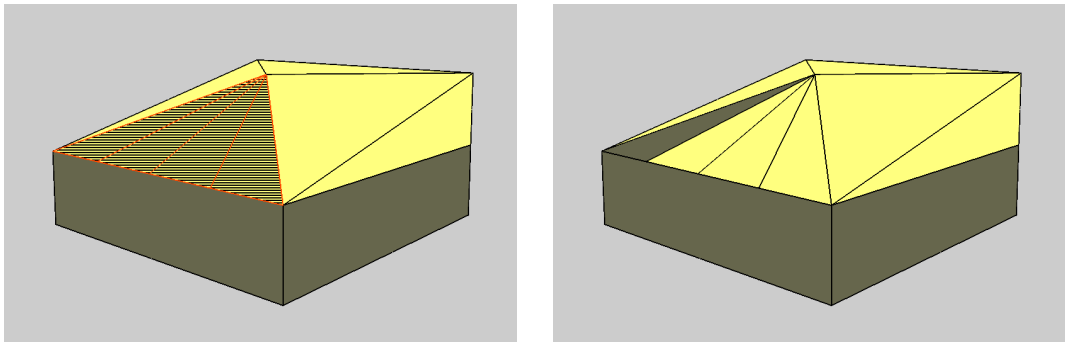


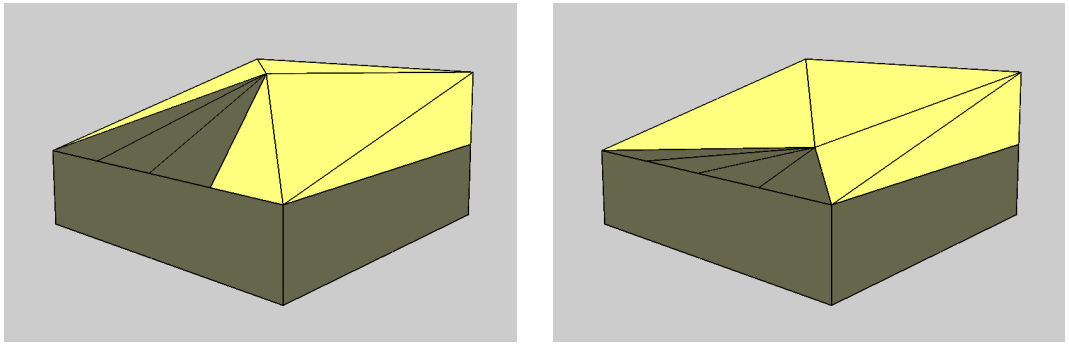
FIGURE 3.22 – Schémas descriptifs de l'algorithme 3.7 en vue éclatée.

Un sommet en se déplaçant peut provoquer plusieurs collisions Face-Face avec la même face (voir Figure 3.23). Nous appliquons successivement le traitement à chaque face et obtenons le résultat illustré par la figure 3.23d qui correspond à nos attentes : l'ensemble des faces est transféré au volume sous-jacent. En effet, les faces d'origine étant coplanaires avec l'horizon, le volume qu'elles représentaient est devenu nul. De plus, on remarque que l'opération conserve la topologie : il y avait 4 faces appartenant au volume jaune, et il y a après traitement le même nombre de faces appartenant au volume gris intersecté.

Étudions le cas, illustré par la Figure 3.24, d'une pyramide ayant pour base carrée F et un apex (pointe de la pyramide) S. La pyramide est liée à un autre volume par α_3 au niveau de F. Si S intersecte F, nous avons quatre collisions Face-Face. Une fois le traitement successif pour les trois premières faces effectué (Figures 3.24b et 3.24c), le volume de la pyramide est nul (Figure 3.24c), mais elle existe toujours topologiquement. Nous avons donc inclus dans notre algorithme un traitement dans le cas où les brins `dF1` et `dF2` appartiennent à la même orbite face $< \alpha_0, \alpha_1, \alpha_3 >$. Dans ce cas nous supprimons simplement les brins appartenant au volume de la pyramide (Figure 3.24d). Nous appelons ce cas un « pincement complet de volume ».



(a) (b)



(c) (d)

FIGURE 3.23 – Cas de la collision entre plusieurs faces et un horizon. (a) Les faces à traiter sont hachurées. (b) La première face a été traitée, elle appartient maintenant au volume gris. (c) La deuxième et la troisième ont été traitées. (d) Toutes les faces ont été traitées et le sommet a été déplacé.

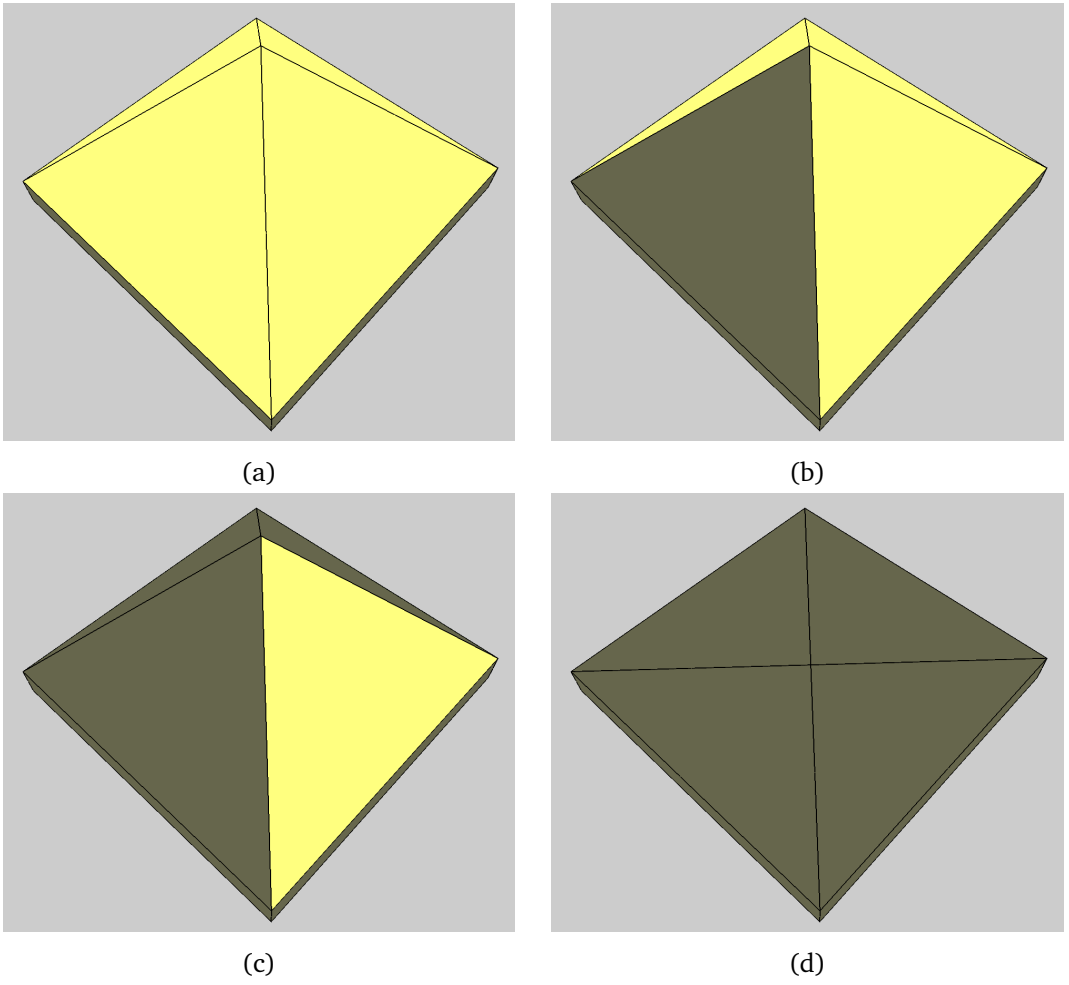


FIGURE 3.24 – Cas du pincement complet d'un volume. (a) La pyramide, en jaune, est liée au volume sous-jacent par une face, le sommet de la pyramide entre en collision avec cette face. (b) La première face est traitée. (c) La deuxième et la troisième faces sont traitées à leur tour. (d) La quatrième face est traitée, la pyramide a été supprimée.

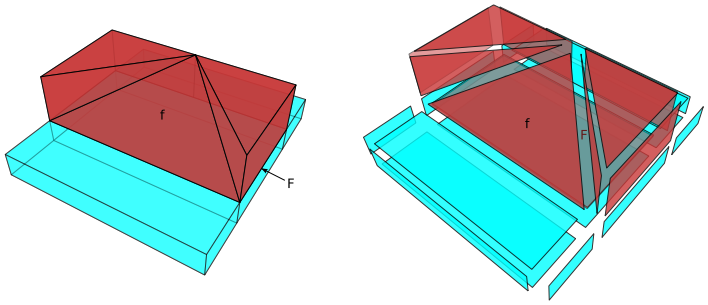
Pincement complet d'une face

Le pincement complet d'une face F peut survenir lors de l'opération d'intersection Face-Face entre une face f et F (voir Figure 3.25). Dans notre contexte où les sommets se déplacent indépendamment les uns des autres, ce cas survient quand le sommet en mouvement appartenant à f intersecte un bord de F. Ce cas peut également survenir lors d'une collision arête-face.

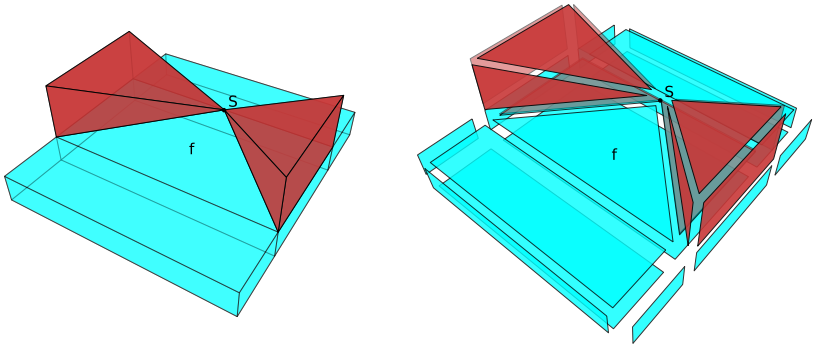
Une face est pincée si deux brins d1 et d2 appartenant à cette face sont tels que (Figure 3.26a) :

- d1 et d2 appartiennent au volume sur lequel nous avons effectué le traitement d'un pincement de face,
- d1 $\neq$ d2 **ET** d1 $\neq \alpha_1(d2)$,
- $< \alpha_1, \alpha_2, \alpha_3 > (d1) \ = < \alpha_1, \alpha_2, \alpha_3 > (d2)$

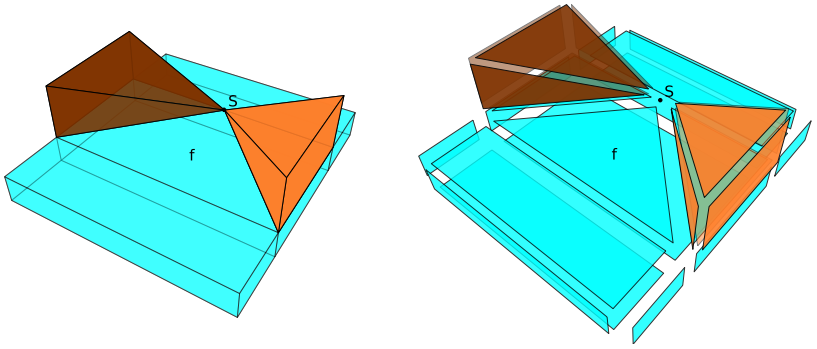
Sur la figure 3.25b, les brins d1 et d2 appartiennent à l'orbite du sommet S : la face F est pincée. Nous utilisons dans un premier temps l'algorithme 3.3 afin de couper la face F. Dans le cas où d1 et d2 sont libres par α_3, nous avons deux sommets distincts. Dans le cas contraire, ils sont toujours topologiquement liés par le sommet S (voir Figure 3.25c). Nous devons alors diviser le sommet pour séparer les deux volumes. Pour cela nous utilisons l'algorithme d'éclatement d'un sommet en arête (Algorithme 3.1) avec pour arguments $\alpha_{32}(d1)$ et $\alpha_{32}(d2)$, marqués respectivement A et B sur la Figure 3.26b. La figure 3.26c illustre le résultat final du traitement : nous avons deux sommets distincts liés par une arête, nous pouvons les déplacer indépendamment l'un de l'autre.



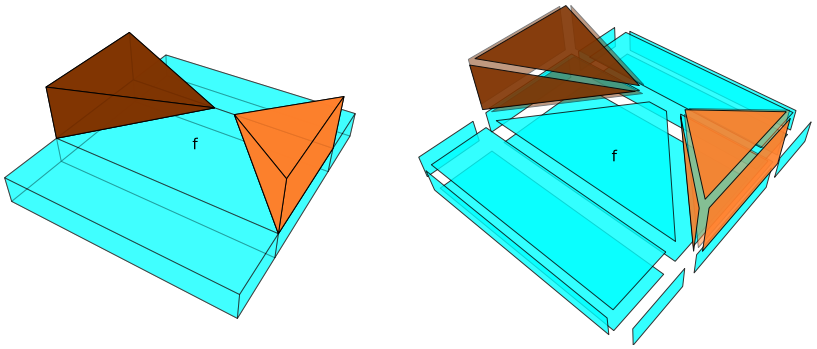
(a) La face f intersecte la face d'horizon F entre les deux volumes.



(b) Après le traitement effectué par le algorithme face-face 3.7, le volume supérieur est pincé au point S.

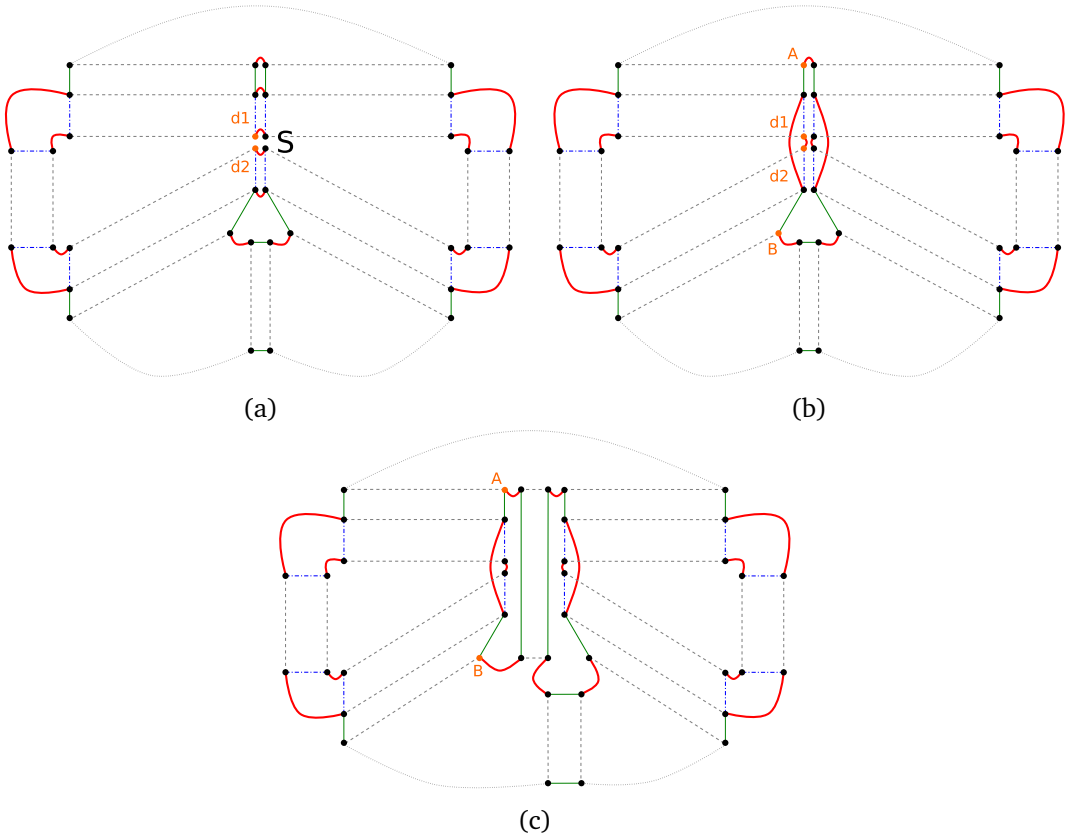


(c) Après le traitement effectué par l'algorithme sommet-sommet 3.3, le volume supérieur est scindé en deux volumes. Ils possèdent toujours le sommet S en commun.



(d) Après le traitement effectué par l'algorithme d'éclatement d'un sommet en arête 3.1, les deux volumes ne possèdent plus de sommets commun.

FIGURE 3.25 – Cas du pincement complet d'une face.



(a) (b) (c)

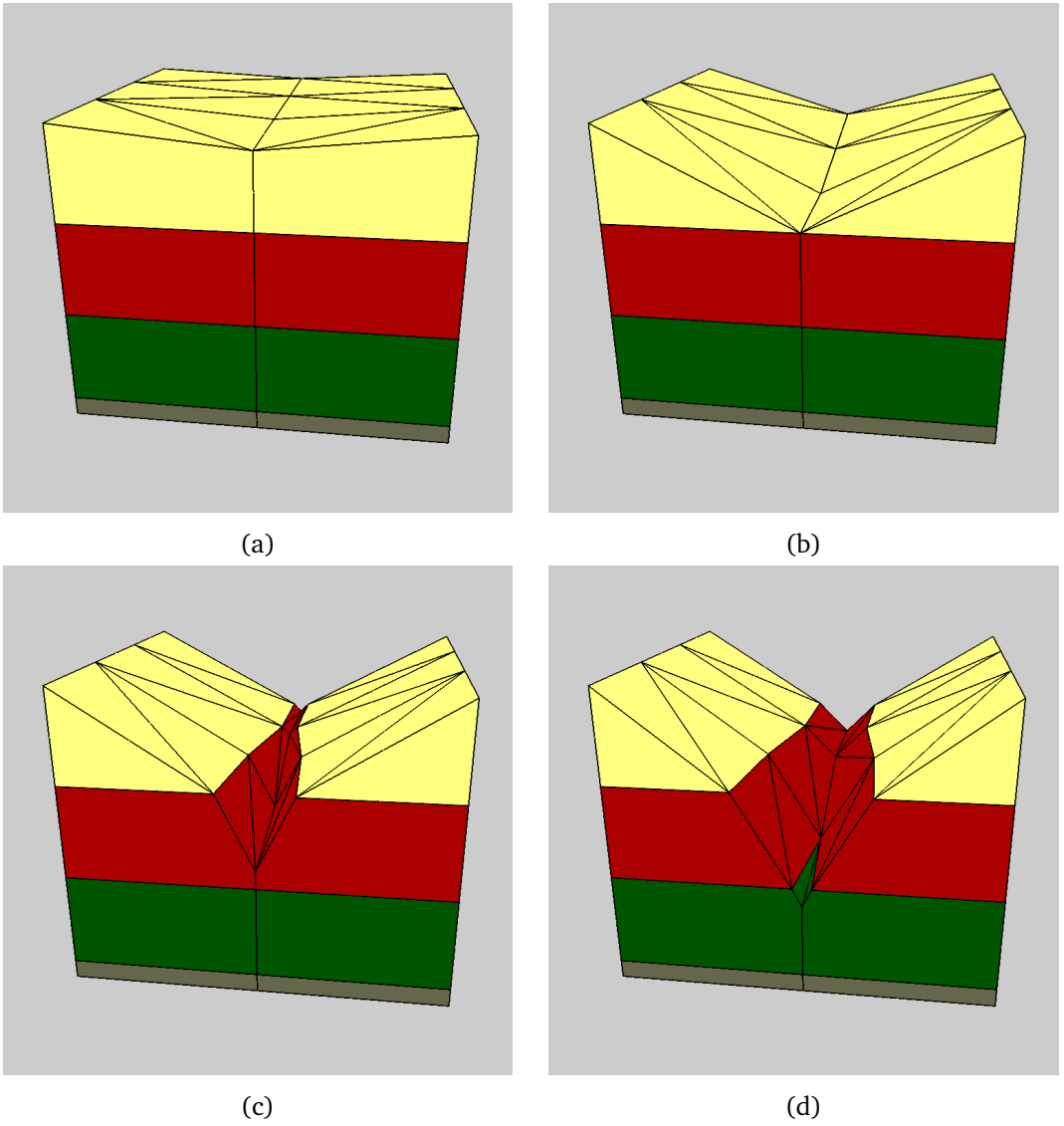
FIGURE 3.26 – Schémas descriptifs du traitement d'une face pincée complètement.

3.4 Applications

Afin de tester nos opérations, nous les avons utilisées dans le contexte de la géomorphologie : un volume représente une couche de matériaux. Nous avons établi différentes simulations de phénomènes naturels. Pour créer une simulation, nous avons besoin d'un maillage représentant un terrain et d'un ensemble de sommets marqués servant d'entrée à un scénario. Le scénario permet d'évaluer le comportement des sommets au cours de la simulation. Il détermine le déplacement des sommets initiaux et des sommets issus du traitement des événements en fonction du phénomène que l'on cherche à reproduire. Par exemple, le comportement des sommets résultant de l'opération de Rosace n'est pas le même dans le cas de création d'une arche et dans le cas de la formation d'une cheminée de fée.

3.4.1 Élargissement et creusement du lit d'une rivière

La scène est composée de quatre volumes superposés, représentant quatre couches. Les sommets de la surface du terrain sont les seuls déplacés, ils sont marqués en rouge sur la figure 3.27a. Nous simulons l'érosion du lit de la rivière en déplaçant verticalement vers le bas les sommets constituant le lit. Dans un premier temps, un sommet entre en collision avec un sommet, une arête ou bien une face appartenant à l'horizon entre la première et la deuxième couche. Après traitement, ce sommet appartient à l'horizon (Figure 3.27b).



(a) (b)

(c) (d)

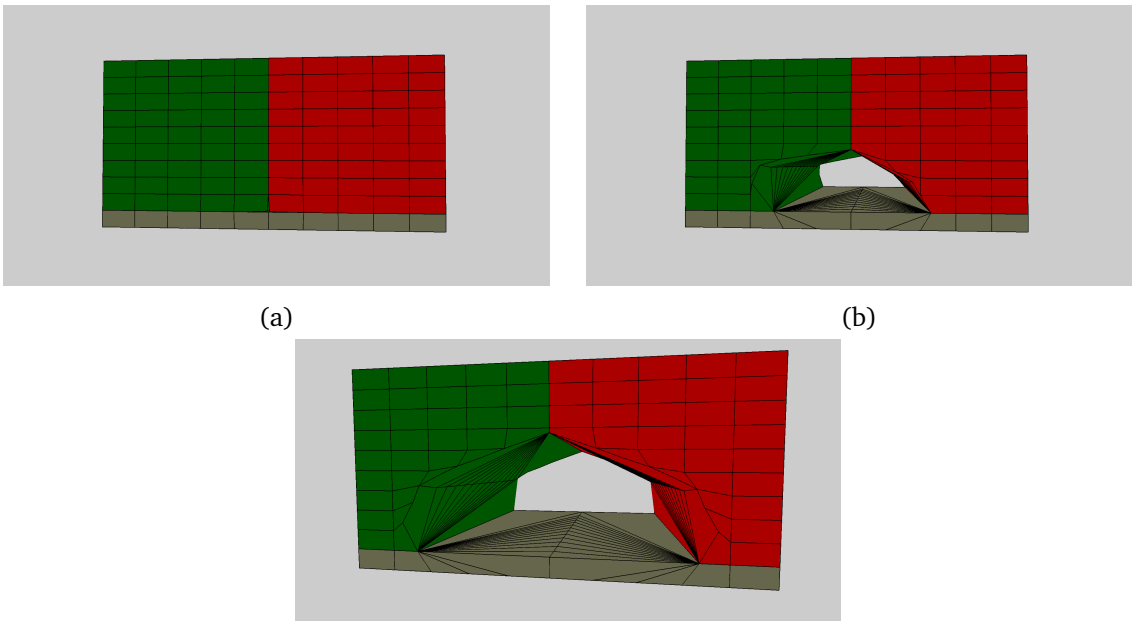
FIGURE 3.27 – Élargissement du lit d'une rivière.

Dans un second temps, un autre sommet entre en collision avec l'horizon, les traitements successifs des collisions font apparaître la couche sous-jacente et vont petit à petit couper la première couche en deux (Figure 3.27c). Le déplacement des sommets se poursuit, et au fur et à mesure la troisième couche apparaît et est érodée à son tour (Figure 3.27d).

La simulation comporte à l'initialisation 36 sommets. Au cours de son évolution, elle atteint au maximum 43 sommets.

3.4.2 Création d'une arche

La scène est composée de trois volumes (voir la figure 3.28a) : un socle non-érodable en gris, une couche verte et une couche rouge. Les 3 volumes partagent une arête A. À l'initialisation, nous ajoutons une extrémité de l'arête que nous appelons S à la liste des événements. Nous affectons un mouvement coplanaire à l'arête permettant de creuser les



(a) (b)

(c)

FIGURE 3.28 – Création d'une arche.

volumes. Afin d'élargir l'arche au fur et à mesure de la simulation, nous ajoutons également à la liste des événements les sommets extrémités des arêtes incidentes à S.

À chaque itération, le premier traitement est celui du sommet S. En effet, il appartient à plusieurs volumes. Nous utilisons alors l'opération de Rosace adéquate pour éclater S en 4 sommets : un pour chaque volume et un (le sommet S) appartenant toujours à l'arête conjointe. Les trois premiers sommets sont alors ajoutés à la liste avec leur propre mouvement pour qu'ils soient traités dans le pas de temps actuel. S est déplacé le long de l'arête afin de creuser un peu plus la cavité.
Le déplacement des autres sommets a pour but d'élargir la cavité. Leur mouvement peut engendrer des collisions de type sommet-sommet, sommet-arête et arête-sommet. À chaque traitement, les nouveaux sommets générés sont également ajoutés à la liste et possèdent leur propre mouvement.
Après un certain nombre d'itérations, le mouvement de S conduit à une collision avec l'autre extrémité de l'arête A. Nous utilisons alors l'opération sommet-sommet avec une arête liant les sommets. Nous utilisons ensuite un post-traitement permettant de créer un trou : nous décousons par α_3 les faces incidentes à S appartenant à plusieurs volumes. Le sommet est ici éclaté en 3 sommets (un pour chaque volume) qui ont leur mouvement propre pour agrandir le trou créé.

Notre simulation débute avec un maillage possédant 240 sommets. En tout, 42 sommets sont créés à l'aide du déplacement de S. Puis le nombre de sommets diminue jusqu'à la fin de la simulation à cause des opérations gérant les collisions. Sur la figure 3.28b, l'érodabilité de la couche verte est supérieure à celle de la couche rouge, elle est donc plus creusée. Sur la figure 3.28c, l'érodabilité des couches verte et rouge sont égales, on

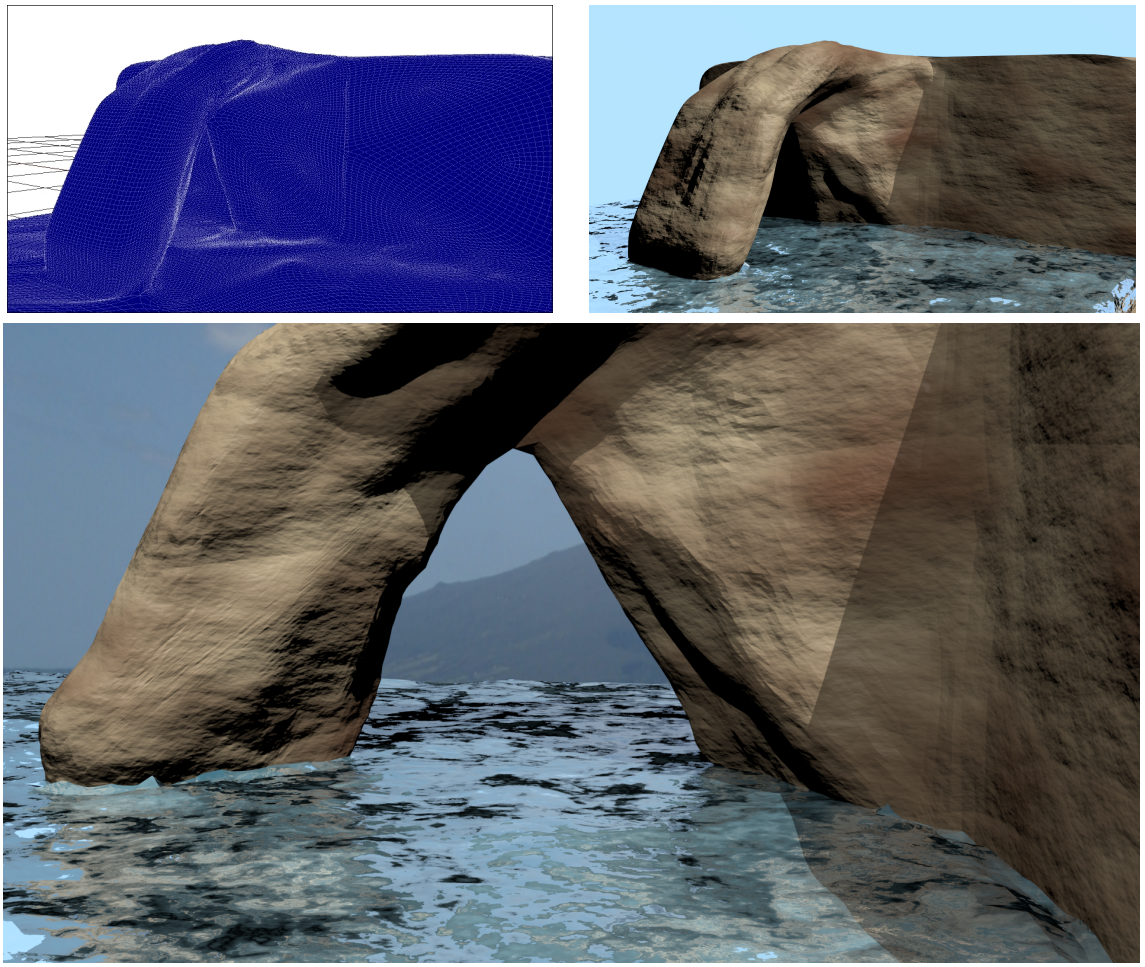


FIGURE 3.29 – Une image visuellement réaliste d'une arche rendue avec Blender avec un raffinement géométrique (Figure en haut à gauche) et des textures de roches et d'océan.

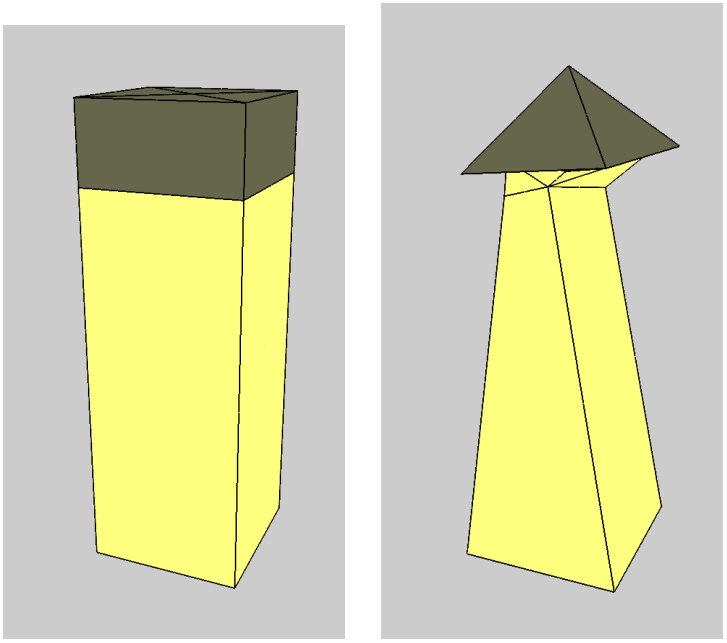
observe la symétrie de l'érosion et de l'arche.

À partir de notre simulation, et en utilisant Blender, un logiciel libre de modélisation et de rendu 3D, nous avons obtenu des images visuellement réalistes de l'arche 3.29.

3.4.3 Formation d'une cheminée de fée

Une cheminée de fée est une colonne naturelle constituée de roches friables dont le sommet est constitué d'une roche plus résistante à l'érosion. La partie plus friable est déblayée plus rapidement que la partie résistante : il s'agit d'un cas d'érosion différentielle. Nous avons élaboré un scénario constitué de deux couches superposées (voir Figure 3.30) : une couche résistante à l'érosion et une couche friable.

La première couche est érodée petit à petit en déplaçant ses sommets verticalement vers le bas, nous traitons ici des collisions de type sommet-sommet et sommet-arête. L'érosion différentielle est réalisée à l'aide de l'opération de « Rosace » permettant d'éclater un sommet appartenant à plusieurs volumes : le sommet appartenant à la couche dure est peu déplacé, celui appartenant à la couche friable permet de l'éroder plus rapidement, et celui appartenant aux deux couches peut être déplacé sur le plan de l'horizon.



(a) (b)

FIGURE 3.30 – Création d'une cheminée de fée à partir de deux volumes ayant une érodabilité différente.

La figure 3.31a représente une cheminée de fée présente à Cotteuge dans le Puy-de-Dôme. Nous avons obtenu la figure 3.31b à partir de notre simulation et en utilisant Blender.

3.4.4 Sédimentation

La scène de ce scénario est un maillage surfacique quadrangulé représentant la surface d'un terrain. À l'initialisation, nous cherchons le sommet ayant la coordonnée z la plus faible et nous marquons les faces incidentes à ce sommet pour leur appliquer l'opération de création de volumes. Nous remplaçons ici la liste des événements par une marque sur le brin composant une orbite sommet ou face parce que nous n'avons pas d'événement à gérer : seulement la création de volume.

Ainsi à chaque itération, nous marquons les faces voisines de celles déjà marquées pour les utiliser l'itération suivante et « élargir » le volume représentant les sédiments.

3.4.5 Création de stalactite et stalagmite

Notre scénario est composé de deux faces (aux normales opposées) F_A et F_B.

La première étape de la simulation est la triangulation des faces avec insertion de sommet que nous appelons S_A pour la face F_A et S_B pour la face F_B (Fig. 3.32a). Ces deux sommets se déplacent selon la normale d'origine de leur face respective. Lors des déplacements de ces sommets, les deux objets A et B gagnent du volume. Nous utilisons successivement l'opération de création de volume à partir d'un sommet. Les nouveaux sommets obtenus sont ajoutés à la file d'attente des événements avec un mouvement permettant de

(a) (b)

FIGURE 3.31

faire élargir l'objet. Enfin nous détectons la collision entre S_A et S_B (Fig. 3.32b) grâce à un calcul de distance, et nous appliquons l'opération d'intersection sommet-sommet. Les nouveaux sommets sont également ajoutés à la file d'attente. La stalactite et la stalagmite se sont rejointes (Fig. 3.32c) et forment une colonne qui s'élargit progressivement (Fig. 3.32d).

La figure 3.33 représente l'évolution d'une stalactite et d'une stalagmite au cours du temps (chronologiquement de gauche à droite), le résultat a été obtenu à l'aide d'un logiciel de modélisation et de rendu 3D.

3.4.6 Création et évolution de grottes

Une grotte se crée généralement grâce à l'infiltration de l'eau dans la roche. Elle forme alors ce que l'on appelle des pertes karstiques qui s'enfoncent dans le sol. Le trou peut s'étendre latéralement lorsqu'il rencontre une couche plus dure ou pour suivre une faille. Ce système peut se reproduire dans la grotte elle-même, ce qui crée plusieurs niveaux. Nous avons créé un scénario montrant que notre modèle permet de reproduire l'évolution d'une grotte. Dans un premier temps, un sommet s'enfonce dans le sol, cela représente une perte karstique due par exemple à l'infiltration de l'eau. Pour simuler la faille, nous avons coupé le volume représentant la roche en deux. Ainsi, lorsqu'en s'enfonçant le sommet atteint l'horizon entre les deux volumes, nous élargissons latéralement la grotte jusqu'à ce qu'elle atteigne le bord de la falaise. Tous ces événements sont gérés par les opérations sommet-face et de l'opération de « rosace ». L'opération sommet-arête, combinée à l'opération arête-face, permet de percer la falaise. Nous avons créé un second niveau en réitérant le processus (Figure 3.34).

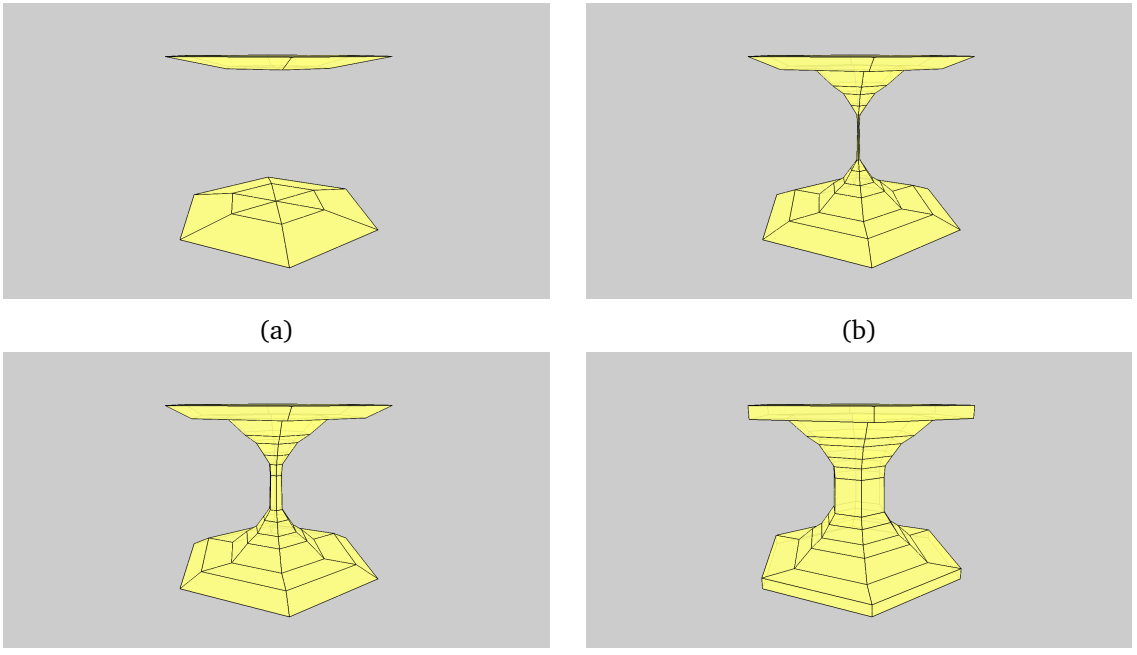


FIGURE 3.32 – Formation d'une colonne à partir de l'évolution d'une stalactite et d'une stalagmite.

3.5 Conclusion

La déformation de maillages volumiques est instable si l'on ne gère pas correctement les événements qui génèrent des incohérences topologiques et géométriques. Dans le cadre de la géomorphologie, ces événements sont nombreux et permettent la création d'une multitude de formes caractéristiques comme les canyons, les arches, stalactites et stalagmites, etc. Nous avons montré à travers ce chapitre une méthode basée sur un modèle d'animation et sur les 3-G-Cartes permettant d'analyser les événements, les trier et ensuite de les traiter à l'aide d'opérations topologiques. Nous avons, en effet, élaboré des opérations événementielles gérant les incohérences liées à des collisions entre sommets, arêtes et faces des volumes. Nous avons également construit des opérations permettant de conserver la cohérence de notre contexte : l'altération d'un terrain.

Nous avons proposé des scénarios permettant de mettre en avant notre modèle. L'élargissement et le creusement d'une rivière est une démonstration prouvant à la fois que notre modèle permet des déplacements latéraux et verticaux, mais également qu'il prend en compte l'érosion différentielle permettant de creuser plus ou moins certaines roches. Le scénario de création d'une arche met en exergue la capacité de notre modèle à générer des cavités et des tunnels à travers des couches de matières. Les deux derniers scénarios illustrent le fonctionnement des opérations de sédimentation, d'abord à grande échelle, et ensuite dans le cadre de la concrétion.

Cependant, les mouvements imprimés aux sommets sont empiriques. L'insertion de nouveaux sommets est problématique car il faut déterminer s'ils sont fixes ou bien en mouvement, et dans ce cas quelle en est la nature ? Ainsi chacun de nos scénarios possède un fonctionnement propre sur ce point afin d'obtenir les résultats que nous avons. Dans le

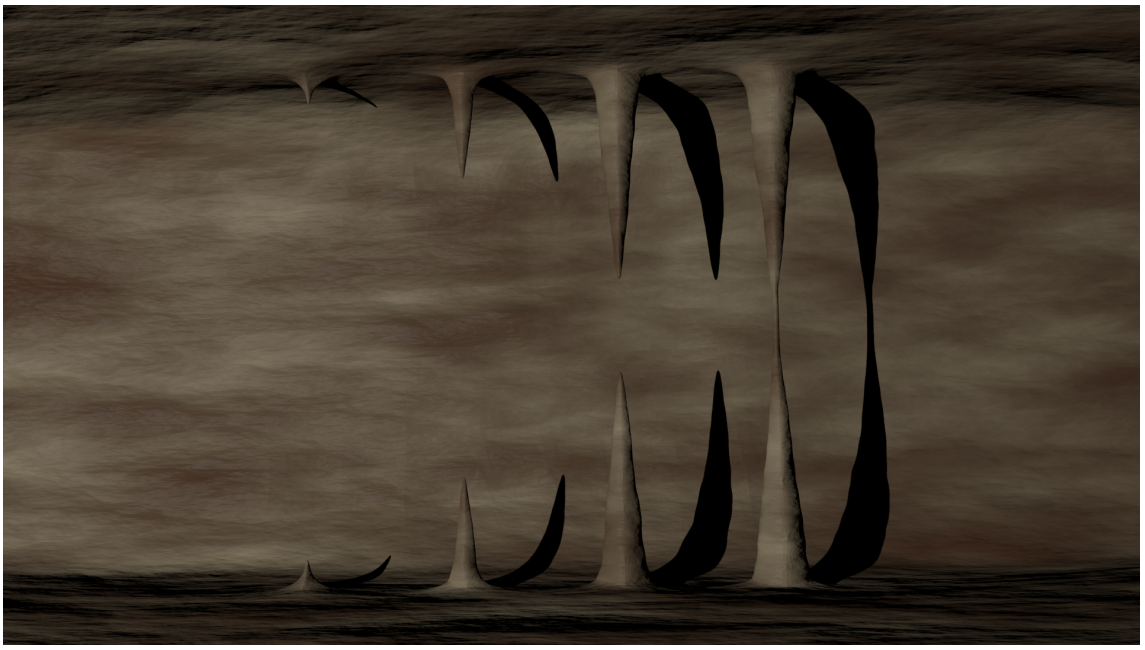
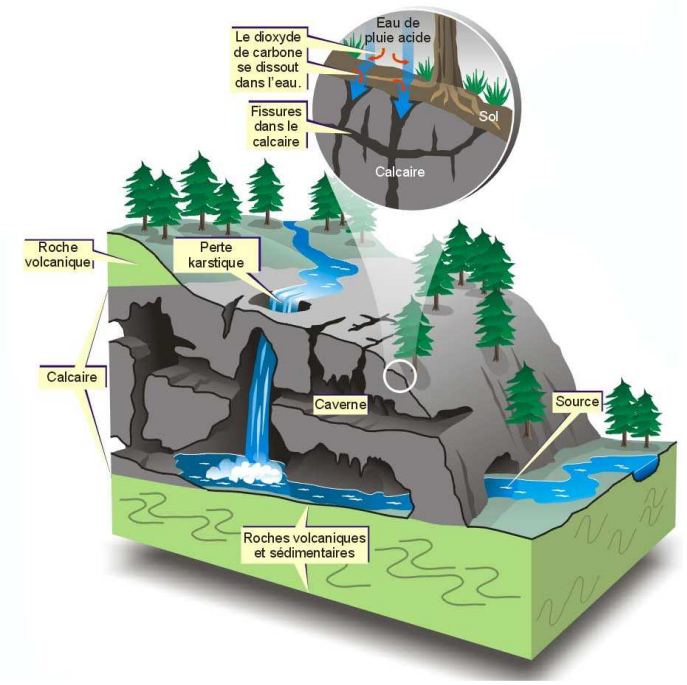
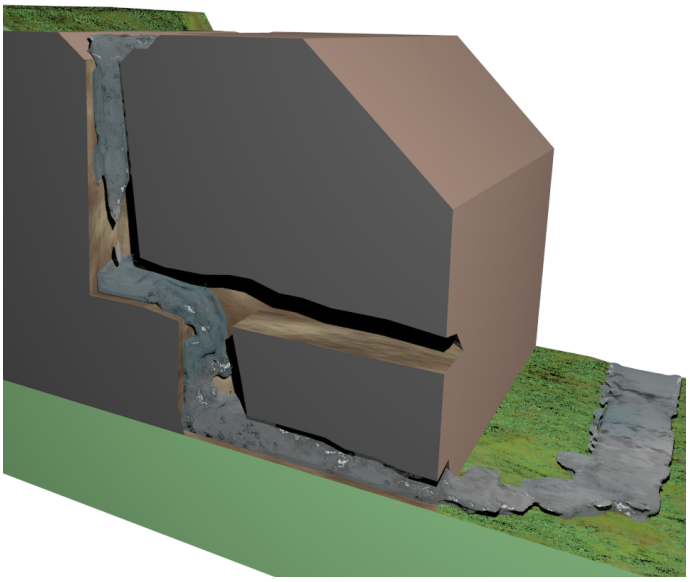


FIGURE 3.33 – Rendu réaliste montrant l'évolution chronologique et topologique (de gauche à droite) d'une stalactite et d'une stalagmite.

cas d'écoulement, la solution est d'utiliser la méthode d'interaction fluide-solide que nous avons développée dans le chapitre 2 pour calculer le déplacement des sommets.



(a)



(b)

FIGURE 3.34 – Formation d'une grotte dans un sol karstique. (a) Schéma issu de .., (b) Image réalisée à partir de notre simulation et de Blender.

Conclusion et perspectives

Au cours de cette thèse, nous avons proposé des méthodes permettant de simuler l'évolution d'un relief de manière réaliste en combinant la simulation de fluide et la modélisation à base topologique. Nous avons montré que la construction d'un paysage réaliste est un problème complexe, notamment du fait que les formes observées dans la nature sont issues d'un ensemble de phénomènes qui ont une influence au cours du temps. Nous pouvons constater que ces formes sont également présentes à différents stades de détérioration. Nous avons décidé de simuler l'évolution d'un terrain subissant l'altération du temps. Des méthodes présentées dans notre état de l'art représentent la structure du terrain et proposent de la faire évoluer au cours du temps, ce qui permet d'effectuer un traitement prenant en compte les caractéristiques des matières composant le terrain. Cependant, durant les modifications, il faut veiller à la cohérence de la structure du terrain.

Nous avons dans un premier temps développé une méthode d'interaction entre un fluide et un terrain permettant d'éroder, de transporter et de déposer de la matière en temps interactif. Nous avons tout d'abord utilisé une approche de type DEM afin de simuler le fluide, l'interaction avec le terrain est basée sur la collision entre les particules en mouvement et les triangles représentant le terrain. Cette approche simule la saltation des sédiments (ils se déplacent par sauts), ceux-ci suivent la pente tant que la vélocité du fluide est supérieure à un seuil déterminé. Elle est implantée sur une architecture parallèle GPU, ce qui permet d'obtenir une simulation en temps interactif. Nous avons ensuite proposé une approche basée sur les SPH, qui a permis de remédier aux problèmes de paramétrages de la simulation. Nous avons également mis en corrélation l'état d'un sédiment (érodé, transporté ou déposé) avec la vélocité du fluide et le type de matière à l'aide de l'abaque de Hjülstrom. Avec ces apports, nous pouvons gérer une érosion différentielle cohérente selon le type de roche. Nous avons montré que notre approche permet de reproduire l'écoulement d'une rivière créant un canyon, mais également la possibilité d'effectuer une érosion latérale d'une surface.

Dans la seconde partie de nos travaux, nous avons proposé un modèle d'évolution topologique appliqué à la géomorphologie. Nous avons vu que l'évolution de la structure d'un terrain représenté en couches provoque des incohérences qu'il faut traiter. Alors que d'autres approches restreignent les déplacements pour éviter ce type d'incohérences, nous n'imposons aucune contrainte en dehors de celles du contexte de la géologie. Pour cela, nous avons mis en place une approche événementielle qui détecte les incohérences topologiques du modèle de terrain ainsi qu'un ensemble d'opérations topologiques permettant de traiter les événements qui surviennent. Elles maintiennent la cohérence topologique et contextuelle du terrain. Cela nous a permis de créer plusieurs scénarios reproduisant

l'évolution de formes remarquables telles que les arches, les stalactites et stalagmites, les grottes, etc.

Perspectives

Nous l'avons vu en conclusion de la partie 3, l'élaboration de scénarios reproduisant l'évolution d'une forme géologique est un problème difficile. Dans [LSM08], Léon *et al.* utilisent des scénarios de haut-niveau qui décrivent l'enchaînement de différentes phases d'érosion ou de sédimentation et permettent de faire évoluer leur terrain représenté en vue 2D en coupe. Pour la sédimentation, leur méthode prend en paramètre une date de début et une date de fin, la vitesse constante de sédimentation ainsi que deux sommets qui définissent les limites de la zone sédimentée. La définition d'une phase d'érosion est similaire à celle d'une phase de sédimentation, la vitesse de sédimentation étant remplacée par la vitesse d'érosion. Nous pourrions adapter ce type de mécanisme à notre modèle. Pour définir les zones, nous pouvons soit utiliser l'ensemble des arêtes appartenant au bord de la zone, soit l'ensemble des faces composant la zone. Néanmoins, Léon *et al.* appliquent un déplacement seulement vertical sur les sommets, sans tenir compte de la géométrie locale autour de ces derniers. Or, nous ne souhaitons pas limiter le déplacement des sommets. Nous souhaitons définir une méthode de calcul du déplacement, spécifique à chaque sommet, en utilisant la somme des normales des faces incidentes au sommet par exemple. Le *sketching* est une autre méthode pour définir les zones d'application. L'utilisateur à l'aide d'une interface dessine directement sur le terrain les zones d'érosion ou de sédimentation. Il peut également fournir la direction et l'intensité de l'opération.

Par ailleurs, les opérations que nous avons créées correspondent à l'évolution d'un terrain telle que décrite par la géomorphologie dynamique. Nous avons vu dans l'introduction qu'il existe également des processus internes à la structure du terrain qui influent sur celui-ci. Ces phénomènes sont étudiés par la géomorphologie structurelle. La tectonique des plaques, par exemple, cause des tensions sur les différentes couches et provoque la création de failles. Celles-ci peuvent avoir une taille de quelques millimètres jusqu'à plusieurs centaines de kilomètres. Il en existe trois sortes (voir Figure 3.35) : normale, inverse et de décrochement, qui sont caractérisées par leur type de glissement. [LSM08] décrit deux types de création de faille : à partir de la surface ou interne à une couche. Léon *et al.* utilisent une forme prédéfinie permettant d'agrandir la faille. Le modèle de glissement de faille est défini par l'ensemble des arêtes constituant la faille, la direction et la vitesse du glissement.

Actuellement, notre modèle ne prend pas en compte la création et le glissement de faille. Néanmoins, nous avons montré dans le scénario d'élargissement d'un lit de rivière qu'il était possible à l'aide de nos opérations topologiques actuelles de scinder un volume en deux, partiellement ou complètement à partir de sa surface. En ce qui concerne la création d'une faille interne à une couche, il nous faut définir une nouvelle opération permettant de créer une cavité à l'intérieur d'un volume. Nous pourrions utiliser le système d'arête fictive que nous avons utilisé pour créer un trou dans une face (voir partie Intersection Sommet-Face 3.3.3). À partir de cette cavité, nous pouvons utiliser les mêmes

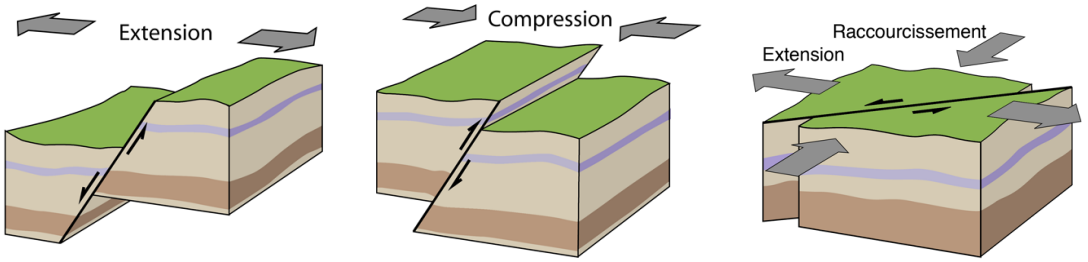


FIGURE 3.35 – Schémas de types de faille : (a) normale, (b) inverse et (c) décrochante. (Images issues de Wikipedia)

opérations qu'à la surface d'un volume. Pour simuler le glissement, nous pouvons combiner les opérations que nous avons créées afin de créer ou de supprimer des éléments topologiques. Par exemple, pour simuler le glissement d'un sommet d'une face à une autre, nous pouvons utiliser successivement les opérations de collision sommet-sommet puis d'éclatement d'un sommet en arête.

Ainsi, les opérations que nous avons déjà définies pourraient être utilisées, voire complétées, afin de reproduire d'autres phénomènes géologiques.

Une autre perspective de notre travail, peut-être la principale, est de faire le lien entre notre modèle d'interaction entre un fluide et un solide et notre modèle d'évolution topologique. En effet, l'objectif est de pouvoir simuler l'évolution d'un terrain sur lequel l'écoule un fluide, comme une rivière, la marée ou le vent. Notre modèle de fluide permet actuellement de prélever de la matière, de la transporter et de la déposer en prenant en compte les propriétés du fluide (vélocité et capacité de transport), et celles de la matière composant le terrain (type et taille des sédiments, contraintes d'érosion, de transport et de sédimentation). Nous pouvons donc définir des zones d'érosion et de sédimentation à l'aide d'une approche physique. Les particules de terrain qui servent d'interface entre le fluide et le maillage représentant le terrain. Ces particules permettent de connaître la quantité de matière érodée et sédimentée à leur position. Notre modèle d'évolution topologique traite les incohérences potentielles engendrées par le déplacement des sommets. Il prend en compte le contexte géologique et permet de reproduire des formes caractéristiques d'un écoulement de rivière. Pour cela, il nous faut définir le comportement des sommets composant le maillage : direction et intensité du déplacement. Notre modèle de fluide ne nous fournit pas encore ces informations. En effet, nous avons donc un problème d'échelle entre le modèle de fluide et le terrain. Pour le moment, nous pouvons calculer la quantité de matière transférée à un point donné du terrain, mais cette quantité est exprimée en millimètres. La taille des sédiments fournie par le diagramme de Hjulström est en millimètres, alors que les dimensions du terrain est de l'ordre de plusieurs centaines de mètres dans le cas d'une rivière. Il nous faut donc imaginer un moyen de faire correspondre le gain ou la perte de matière du terrain et le déplacement effectif des sommets du maillage.

Une dernière perspective, à plus long terme, consiste en la création d'applications utilisables par des géologues, dans le cadre de leur pratique pédagogique ou de la recherche.

En effet, le modèle d'évolution topologique présenté dans ce mémoire permet d'une part d'obtenir un terrain réaliste résultant de l'action de divers phénomènes, et d'autre part, d'obtenir un état futur plausible visuellement et structurellement à partir d'un terrain quelconque. Par conséquent, nous pourrions envisager tout d'abord une application qui permettrait aux enseignants en géologie de créer des animations utiles pour illustrer les effets de la géomorphologie en général. L'intérêt pédagogique serait bien entendu pour les étudiants la possibilité de comprendre et de s'approprier plus facilement les phénomènes sous-jacents. De plus, la version purement scénarisée évoquée en début de ces perspectives pourrait offrir un outils pour la géologie, plus orientés vers la recherche, dont le but serait de tester des hypothèses d'enchaînements des phénomènes qui ont pu conduire à la formation d'un terrain. La partie simulation de l'interaction fluide/solide pourrait également être utile, à condition de proposer des solutions d'ingénierie inverse : il faut alors trouver les conditions initiales permettant d'obtenir le résultat escompté. C'est un objectif à long terme qui demandera une collaboration étroite avec des géologues. Pour toutes ces applications, le challenge réside également dans le fait de mettre à disposition d'utilisateurs non-spécialistes des outils intuitifs ne nécessitant pas de connaissances approfondies dans le domaine informatique, et plus particulièrement celui des modèles topologiques.

L'application de la topologie à la géomorphologie nous a permis d'obtenir des résultats probants, il pourrait être intéressant d'utiliser la même méthodologie dans d'autres domaines scientifiques.

Bibliographie

[Ama06] Takashi Amada. Real-time particle-based fluid simulation with rigid body interaction. In Michael Dickheiser, editor, *Game Programming Gems 6*, pages 189–205. Charles River Media, 2006.

[Bad90] Didier Badouel. An efficient ray-polygon intersection. In *Graphics Gems I*, pages 390–393, 1990.

[BCS+11a] Richard Bézin, Benoît Crespin, Xavier Skapin, Olivier Terraz, and Philippe Meseure. Opérations topologiques pour la géomorphologie. In *Actes de l'AFIG 2011*, pages 95–104, Biarritz, France, October 2011.

[BCS+11b] Richard Bézin, Benoît Crespin, Xavier Skapin, Olivier Terraz, and Philippe Meseure. Topological Operations for Geomorphological Evolution. In *Proceedings of VRIPHYS 2011*, Lyon, France, December 2011.

[Bel07] Farès Belhadj. Terrain modeling : a constrained fractal model. In *Proceedings of the 5th international conference on Computer graphics, virtual reality, visualisation and interaction in Africa*, AFRIGRAPH '07, pages 197–204, New York, NY, USA, 2007. ACM.

[BF01] Bedřich Beneš and Rafael Forsbach. Layered data representation for visual simulation of terrain erosion. In *Proceedings of the 17th Spring conference on Computer graphics*, SCCG '01, pages 80–, Washington, DC, USA, 2001. IEEE Computer Society.

[BF02] Bedřich Beneš and Rafael Forsbach. Visual simulation of hydraulic erosion. In *WSCG Proceedings of the 10-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*, pages 120–132, 2002.

[BFO+07] Mathew Beardall, Mckay Farley, Darius Ouderkirk, Jeremy Smith, Michael Jones, and Parris Egbert. Goblins by Spheroidal Weathering. In D. Ebert and S. Merillou, editors, *Natural Phenomena*, pages 7–14, Prague, Czech Republic, 2007. Eurographics Association.

[BPC+10] Richard Bézin, Alexandre Peyrat, Benoît Crespin, Olivier Terraz, Xavier Skapin, and Philippe Meseure. Interactive hydraulic erosion using CUDA. In *Proceedings of ICCVG 2010*, volume 6374 of *Lecture Notes in Computer Science*, pages 225–232, Varsovie, Pologne, Sep 2010. Springer Verlag.

[BPC+11] Richard Bézin, Alexandre Peyrat, Benoît Crespin, Olivier Terraz, Xavier Skapin, and Philippe Meseure. Interactive hydraulic erosion using CUDA. *Machine Graphics & Vision*, 20 :157–171, 2011. version étendue de [BPC+10].

[BSP+04] Sylvain Brandel, Sébastien Schneider, Michel Perrin, Nicolas Guiard, Jean-François Rainaud, Pascal Lienhardt, and Yves Bertrand. Automatic Building of Structured Geological Models. In *2004 ACM Symposium on Solid Modeling and Applications*, June 2004.

[BTHB06] Bedřich Beneš, Václav Těšínský, Jan Hornyš, and Sanjiv K. Bhatia. Hydraulic erosion : Research articles. *Comput. Animat. Virtual Worlds*, 17(2) :99–108, 2006.

[Caz97] David Cazier. *Construction de systèmes de réécriture pour les opérations booléennes en modélisation géométrique*. PhD thesis, Université Louis Pasteur de Strasbourg, 1997.

[CD96] David Cazier and Jean-François Dufourd. Rewriting-based derivation of efficient algorithms to build planar subdivisions. In *Spring Conference on Computer Graphics*, Jun 1996.

[CD97] David Cazier and Jean-François Dufourd. Reliable boolean operations on polyhedral solids thanks to a 3d refinement. In *Winter School on Computer Graphics*, pages 40–49, Feb 1997.

[CI84] B. Chazelle and J. Incerpi. Triangulation and shape-complexity. *ACM Trans. Graph.*, 3(2) :135–152, April 1984.

[CMF98] Norishige Chiba, Kazunobu Muraoka, and Kunihiko Fujita. An erosion model based on velocity fields for the visual simulation of mountain scenery. *Journal of Visualization and Computer Animation*, 9(4) :185–194, 1998.

[CTVW88] K. L. Clarkson, R. E. Tarjan, and C. J. Van Wyk. A fast las vegas algorithm for triangulating a simple polygon. In *Proceedings of the fourth annual symposium on Computational geometry*, SCG '88, pages 18–22, New York, NY, USA, 1988. ACM.

[DEJ+99] Julie Dorsey, Alan Edelman, Henrik Wann Jensen, Justin Legakis, and Hans Køhling Pedersen. Modeling and rendering of weathered stone. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '99, pages 225–234, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[DGC96] Mathieu Desbrun and Marie-Paule Gascuel-Cani. Smoothed particles : a new paradigm for animating highly deformable bodies. In *Proceedings of the Eurographics workshop on Computer animation and simulation '96*, pages 61–76, New York, NY, USA, 1996. Springer-Verlag New York, Inc.

[FFC82] Alain Fournier, Don Fussell, and Loren Carpenter. Computer rendering of stochastic models. *Commun. ACM*, 25(6) :371–384, June 1982.

[FM84] A. Fournier and D. Y. Montuno. Triangulating simple polygons and equivalent problems. *ACM Trans. Graph.*, 3(2) :153–174, April 1984.

[FM96] Nick Foster and Dimitri Metaxas. Realistic animation of liquids. In *Proceedings of the conference on Graphics interface '96*, GI '96, pages 204–212, Toronto, Ont., Canada, Canada, 1996. Canadian Information Processing Society.

[GGG+12] Jean-David Génevaux, Éric Galin, Éric Guérin, Adrien Peytavie, and Bedřich Beneš. Génération procédurale de rivières et de terrains. In *Actes de l'AFIG 2012*, Calais, France, 2012.

[Gre08] Simon Green. Particle-based fluid simulation for games. In *Game Developers Conference*, 2008.

[Gui06] Nicolas Guiard. *Construction de modèles géologiques 3D par co-raffinement de surfaces*. PhD thesis, École des Mines de Paris, 2006.

[GZWW11] Bo Geng, HuiJuan Zhang, Heng Wang, and GuoPing Wang. Approximate poisson disk sampling on mesh. *Science China Information Sciences*, pages 1–12, 2011.

[HBD10] A. Herault, G. Bilotta, and R. A. Dalrymple. SPH on GPU with CUDA. *Journal of Hydraulic Research*, 48(Extra Issue) :74–79, 2010.

[Hju35] F. Hjulström. Studies of the morphological activity of rivers as illustrated by the river Fyris. *Bulletin of the Geological Institute of the University of Uppsala*, 25 :221–346, 1935.

[HKK07] T. Harada, S. Koshizuka, and Y. Kawaguchi. Smoothed particle hydrodynamics on GPUs. In *Computer Graphics International*, pages 63–70, 2007.

[HNW93] E. Hairer, S. P. Nørsett, and G. Wanner. *Solving ordinary differential equations I (2nd revised. ed.) : nonstiff problems*. Springer-Verlag New York, Inc., New York, NY, USA, 1993.

[IFMC03] Tomoya Ito, Tadahiro Fujimoto, Kazunobu Muraoka, and Norishige Chiba. Modeling rocky scenery taking into account joints. In *Computer Graphics International*, pages 244–247. IEEE Computer Society, 2003.

[JCD09] Thomas Jund, David Cazier, and Jean-François Dufourd. Particle-based forecast mechanism for continuous collision detection in deformable environments. In *SPM '09 : 2009 SIAM/ACM Joint Conference on Geometric and Physical Modeling*, pages 147–158, New York, NY, USA, 2009. ACM.

[JCD10] Thomas Jund, David Cazier, and Jean-François Dufourd. Edge collision detection in complex environment. In *CGI 2010 : Computer Graphics International*, 2010.

[Jen08] W. Jenkins. Modeling, data analysis and numerical techniques for geochemistry. http://w3eos.whoi.edu/12.747/chaps/chap11/chapter_ade.pdf, 2008.

[KBKv09] Peter Krištof, Bedřich Beneš, Jaroslav Křivánek, and Ondřej Šťava. Hydraulic erosion using smoothed particle hydrodynamics. *Computer Graphics Forum (Proceedings of Eurographics 2009)*, 28(2), mar 2009.

[Kel06] Micky Kelager. Lagrangian Fluid Dynamics Using Smoothed Particle Hydrodynamics. Master's thesis, University of Copenhagen, 2006.

[KMN88] Alex D. Kelley, Michael C. Malin, and Gregory M. Nielson. Terrain simulation using a model of stream erosion. *SIGGRAPH Comput. Graph.*, 22(4) :263–268, June 1988.

[KS09] Javor Kalojanov and Philipp Slusallek. A parallel algorithm for construction of uniform grids. In *High Performance Graphics*, pages 23–28, 2009.

[LC87] William E. Lorensen and Harvey E. Cline. Marching cubes : A high resolution 3d surface construction algorithm. *SIGGRAPH Comput. Graph.*, 21(4) :163–169, August 1987.

[Lew87] J. P. Lewis. Generalized stochastic subdivision. *ACM Trans. Graph.*, 6(3) :167–190, July 1987.

[Lie94] P. Lienhardt. N-dimensional generalized combinatorial maps and cellular quasi-manifolds. *International Journal on Computational Geometry and Applications*, 4(3) :275–324, 1994.

[LSM08] Pierre-François Léon, Xavier Skapin, and Philippe Meseure. A topology-based animation model for the description of 2d models with a dynamic structure. In François Faure and Matthias Teschner, editors, *VRIPHYS*, pages 67–76. Eurographics Association, 2008.

[Man83] Benoit B. Mandelbrot. *The Fractal Geometry of Nature*. W. H. Freedman and Co., New York, 1983.

[MCG03] Matthias Müller, David Charypar, and Markus Gross. Particle-based fluid simulation for interactive applications. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '03, pages 154–159, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[MDH07] Xing Mei, Philippe Decaudin, and Baogang Hu. Fast hydraulic erosion simulation and visualization on gpu. In *Pacific Graphics*, 2007.

[MKM89] F. K. Musgrave, C. E. Kolb, and R. S. Mace. The synthesis and rendering of eroded fractal terrains. *SIGGRAPH Comput. Graph.*, 23(3) :41–50, July 1989.

[Mon92] J. J. Monaghan. Smoothed particle hydrodynamics. *Annual Review of Astronomy and Astrophysics*, 30(1) :543–574, 1992.

[Mon05] J. J. Monaghan. Smoothed particle hydrodynamics. *Reports on Progress in Physics*, 68(8) :1703–1759, 2005.

[Moo65] Gordon E. Moore. Cramming more components onto integrated circuits. *Electronics*, 38, 1965.

[MP89] Gavin Miller and Andrew Pearce. Globular dynamics : A connected particle system for animating viscous fluids, 1989.

[MST04a] Matthias Müller, Simon Schirm, and Matthias Teschner. Interactive blood simulation for virtual surgery based on smoothed particle hydrodynamics. *Technol. Health Care*, 12(1) :25–31, February 2004.

[MST+04b] Matthias Müller, Simon Schirm, Matthias Teschner, Bruno Heidelberger, and Markus Gross. Interaction of fluids with deformable solids : Research articles. *Comput. Animat. Virtual Worlds*, 15(3-4) :159–171, July 2004.

[Mus93] Forest Kenton Musgrave. *Methods for realistic landscape imaging*. PhD thesis, Yale University, New Haven, CT, USA, 1993. UMI Order No. GAX94-15872.

[Ngu08] Hubert Nguyen, editor. *GPU Gems 3*. NVIDIA Corporation, 2008.

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

[Ols04] Jacob Olsen. Realtime procedural terrain generation - realtime synthesis of eroded fractal terrain for use in computer games. *Department of Mathematics And Computer Science IMADA University of Southern Denmark*, 2004.

[Par65] Partheniades. Erosion and deposition of cohesive soils. *Proceedings of the American Society of Civil Engineers (ASCE)*, 91 (HY1) :105–139, 1965.

[Per85] Ken Perlin. An image synthesizer. *SIGGRAPH Comput. Graph.*, 19(3) :287–296, July 1985.

[PGMG09] Adrien Peytavie, Eric Galin, Stephane Merillou, and Jerome Grosjean. Arches : a Framework for Modeling Complex Terrains. *Computer Graphics Forum (Proceedings of Eurographics)*, 28(2) :457–467, 2009.

[PH93] Przemyslaw Prusinkiewicz and Mark Hammel. A fractal model of mountains with rivers. In *In proceedings of Graphics Interface '93*, pages 174–180, 1993.

[PTM+11] Alexandre Peyrat, Olivier Terraz, Stéphane Mérillou, Eric Galin, and Djamchid Ghazanfarpour. Generating large-scale details : Altering soil surface and structure with tracks. In Jan Bender, Kenny Erleben, and Eric Galin, editors, *VRIPHYS*, pages 129–137. Eurographics Association, 2011.

[RB85] William T. Reeves and Ricki Blau. Approximate and probabilistic algorithms for shading and rendering structured particle systems. *SIGGRAPH Comput. Graph.*, 19(3) :313–322, July 1985.

[SBBK08] Ondrej Stava, Bedřich Beneš, Matthew Brisbin, and Jaroslav Krivanek. Interactive terrain modeling using hydraulic erosion. In Markus Gross and Doug James, editors, *Eurographics/SIGGRAPH Symposium on Computer Animation*, pages 201–210, Dublin, Ireland, 2008. Eurographics Association.

[SS09] Michael Schwarz and Marc Stamminger. Fast GPU-based adaptive tessellation with CUDA. *Computer Graphics Forum*, 28(2 (Proceedings of Eurographics 2009)) :365–374, March 2009.

[Sta99] Jos Stam. Stable fluids. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '99, pages 121–128, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

[Std] Khronos Group Std. The OpenCL Specification. http://www.khronos.org/opencl/.

[TPF89] Demetri Terzopoulos, John Platt, and Kurt Fleischer. From Goop to Glop : Heating and Melting Deformable Models. *Proceedings Graphics Interface*, 2(2) :219–226, June 1989.

[Tut84] W. Tuttle. *Graph theory, Encyclopedia of Mathematics and its Applications*. Addison Wesley, 1984.

[VHB87] Brian Von Herzen and Alan H. Barr. Accurate triangulations of deformed, intersecting surfaces. In *SIGGRAPH*, pages 103–110, 1987.

[VPLL06] Gilles Valette, Stéphanie Prévost, Laurent Lucas, and Joël Léonard. Soda project : a simulation of soil surface degradation by rainfall. *Computers & Graphics*, 30(4) :494–506, aug 2006.

[WCMT07] Christopher Wojtan, Mark Carlson, Peter J. Mucha, and Greg Turk. Animating corrosion and erosion. In David S. Ebert and Stéphane Mérillou, editors, *NPH*, pages 15–22. Eurographics Association, 2007.

Vì Veri Veniversum Vivus Vici.
Goethe