Revue Électronique Francophone d'Informatique Graphique, Volume 3, Numéro 1, pp. 31–50, 2009

Modèle générateur d'évolutions géologiques par animation basée sur la topologie

Pierre-François Léon, Xavier Skapin et Philippe Meseure

Laboratoire CNRS XLIM-SIC - Université de Poitiers - UMR n° 6172

Résumé

Cet article présente une nouvelle méthode d'animation basée sur la topologie. Elle consiste à animer une partition de l'espace et en assurer sa cohérence. Le système d'animation s'appuie sur un mécanisme événementiel qui détecte les incohérences topologiques : l'animation est générée à partir du traitement séquentiel de tous les événements. Ces derniers sont de deux types : des événements initiaux déterminés par l'utilisateur et des événements générés par des collisions dont les instants sont calculés à partir du mouvement des entités. Pour prendre en charge ces collisions, ces événements sont traités par rapport à leurs contextes locaux (géométrique, sémantique). Un traitement engendre des changements géométriques et topologiques et assure la cohérence entre le modèle géométrique et le modèle topologique. La méthode est illustrée par une application en géologie permettant de générer l'animation de l'évolution du sous-sol à partir de phénomènes naturels décrits dans un scénario. Dans ce cadre, un scénario est composé d'une suite de phénomènes géologiques (sédimentation, érosion, création de failles et glissement) analysée par le système d'animation 2D et chaque phénomène est traduit en un ensemble d'événements initiaux. Une fois l'animation générée, le géologue peut l'analyser et la valider grâce aux informations sémantiques et historiques fournies par le modèle.

This article presents a new topology-based animation method. It consists in animating a space partition while ensuring its consistency. The whole scene is described with a high-level language depending on the application. The animation system relies on an event approach which detects topological events and the animation itself is generated from the sequential processing of all events. Those events either are user-defined (so-called "initial events") or are generated when entities collide (date collisions are computed during the motion of entities. For processing those collisions, events are handled with respect to their local contexts (both geometrical and semantical). We use a geological application to illustrate our point, where the user can generate a subsoil evolution from some natural phenomena described by a scenario. A scenario is made of a sequence of geological phenomena (sedimentation, erosion, fault creation and sliding, each one translated into a set of initial events) analyzed by the animation system. After the animation has been generated, the geologist can analyze and validate it with the semantical and historical information given by the model.

1. Introduction

De nombreuses sciences expérimentales (biologie, botanique, géologie, etc.) sont confrontées à des structures naturelles très élaborées dont les lois de formation sont complexes et souvent mal maîtrisées. Un modèle permettant de représenter l'évolution de ces structures au cours du temps, de contrôler cette représentation, de remettre en cause les phénomènes mis en jeu et de fournir l'historique des diverses entités impliquées dans cette évolution se révélerait un outil appréciable pour vérifier les hypothèses de formation et d'évolution des structures naturelles. Dans cet article, nous présentons un modèle général *nD* générateur d'animations répondant à ces exigences. Ce modèle est basé sur un ensemble d'entités topologiques représentant la structure naturelle étudiée ; modéliser l'évolution de cette structure revient alors à contrôler l'évolution des entités topologiques sous-jacentes. Ce contrôle est assuré par la détection automatique de tous les changements topologiques intervenant entre ces entités au cours du temps et par le traitement explicite de ces modifications en termes d'opérations topologiques élémentaires.

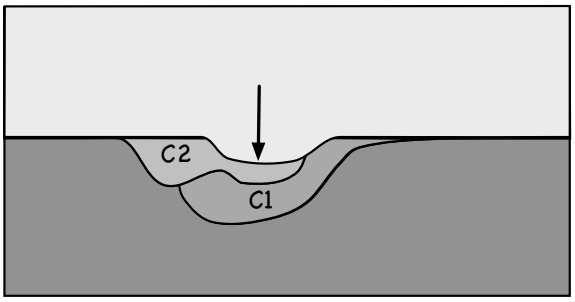

Figure 1: *Exemple de scène géologique obtenue par alternance de sédimentations (couches $C1$ puis $C2$) et d'érosions. La flèche indique le sens de l'érosion au cours du temps.*

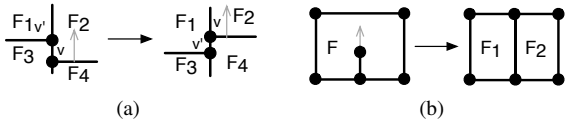

Figure 2: *Exemples de changements topologiques. (a) Glissement d'un sommet v sur un autre v' avec changement des relations d'adjacence entre les quatre faces. (b) Scission d'une face F en F_1 et F_2 causée par le déplacement du sommet.*

Cet article présente une nouvelle méthode d'animation et sa mise en œuvre dans le cas d'une application en géologie capable de modéliser et d'animer certains phénomènes géologiques responsables de la structure du sous-sol. En effet, la construction de ces structures complexes peut être réalisée à partir de l'étude d'un nombre réduit de phénomènes simples. Nous nous focalisons dans cet article sur une première version de notre système d'animation, restreinte à la dimension 2. En pratique, même si cette dimension peut paraître limitative, elle permet cependant de décrire des phénomènes suffisamment complexes pour montrer les possibilités de notre système, en particulier en géologie (en effet, le sous-sol est souvent représenté par des planches $2D$). Les phénomènes étudiés dans cet article sont la sédimentation, l'érosion, la création de failles et le glissement. Ces quatre phénomènes et leurs combinaisons engendrent de nombreuses modifications géométriques et topologiques. La figure 1 illustre un exemple d'évolution, en l'occurrence un début d'érosion (cf. Section 5.3). Si elle se poursuit, il est probable que les couches $C1$ et $C2$ finiront par être séparées. En outre, une forte érosion peut conduire $C1$ à être coupée en deux blocs $C1_1$ et $C1_2$ non adjacents. Sur le plan topologique, cette évolution peut se décrire par les événements suivants : glissement et séparation d'interfaces (Figure 2(a)), scission de face (Figure 2(b)), etc.

La suite de l'article est organisée comme suit : la section 2 présente les travaux antérieurs en rapport avec la modélisation et la description d'animation puis expose la démarche générale de notre modèle d'animation. La section 3 explique les grands principes de notre modèle d'animation de structures topologiques à base de n-g-cartes. La section 4 présente l'étude détaillée du cas de la dimension 2. La section 5 décrit les quatre phénomènes géologiques intégrés dans notre système d'animation avant de présenter des résultats montrant le processus d'écriture d'un scénario d'évolution et sa traduction en opérations menant à la génération d'une animation. Enfin, la section 6 dresse le bilan de nos travaux et leurs perspectives.

2. Animation topologique

2.1. Travaux antérieurs

Peu de travaux se sont intéressés à l'évolution de structures, même si plusieurs modèles d'animation se sont appuyés sur des modèles topologiques. À notre connaissance, les seules méthodes qui ont cherché à faire évoluer une structure topologique et ont pu représenter des modèles structurés dynamiques sont des systèmes constructifs, généralement basés sur la théorie des langages : les L-systèmes [Lin68] [PL90] et leur extension volumique [GTM*05], les map-L-systèmes [LR79], les systèmes vertex-vertex [Smi06] et MGS [GM01].

Un L-système est une grammaire formelle utilisée afin de modéliser des processus de développement et de prolifération de plantes et de bactéries. Cette grammaire formelle comprend un alphabet V, un ensemble de constantes S, un axiome de départ ω et un ensemble de règles P d'évolutions du système. Les variantes de ce modèle portent principalement sur l'application des règles (utilisation du contexte, de conditions, probabilités). Ce système a été étendu à des constructions volumiques pour modéliser la croissance interne du bois. Cette technique utilise un modèle topologique évoluant au cours du temps, comparable aux L-systèmes classiques mais en s'appliquant à des volumes au lieu d'arêtes. La croissance est guidée par l'application séquentielle de règles choisies suivant le contexte. Comme pour les L-systèmes, les transformations consistent essentiellement en des subdivisions hiérarchiques d'un maillage. Les map-L-systèmes appliquent le principe des L-systèmes sur des graphes. Les systèmes vertex-vertex sont décrits par un graphe non orienté où les sommets représentent les sommets de la structure, et les liens représentent une permutation des arêtes autour des sommets. Le système évolue par un ensemble de modifications appliquées au cours du temps. L'ensemble de ces approches reposent sur la théorie des langages, et permettent de faire évoluer des systèmes pouvant se réduire à des structures linéaires. Enfin, MGS est un langage de programmation pour la transformation de structures, essentiellement basé sur un système à base de règles de transformations.

Toutes ces structures n'apportent pas réellement de notions de cohérence géométrique et topologique.

2.2. Démarche

Le modèle proposé ne repose pas sur une structure linéaire. Au contraire, il consiste en une animation d'une subdivision de l'espace. Il se base sur un modèle topologique, les cartes généralisées, que l'on exploite dans une structure temporelle. Plus précisément, le modèle représente l'animation comme une succession de cartes généralisées, où chaque carte représente un ensemble de modifications topologiques simultanées et supposées instantanées. Pour définir les instants où une carte est nécessaire et plus généralement, garantir que le modèle topologique reste cohérent lors des mouvements de ses entités, une gestion événementielle des modifications topologiques est adjointe au modèle. Un scénario fourni par l'utilisateur spécifie les mouvements globaux de la structure. Ce scénario est traduit en série d'événements initiaux et son interprétation aboutit à la génération de l'animation. Un modèle sémantique complète notre modèle d'animation en lui fournissant un mécanisme de désignation des entités, sur lequel le scénario peut s'appuyer et qui lui permet d'être exprimé en termes « haut-niveau », *i.e.* facilement compréhensibles par l'utilisateur final. La désignation est hiérarchique afin de renseigner le modèle sur l'origine des entités. En outre, une description séquentielle des transformations que subit la carte initiale est également générée lors de l'élaboration de l'animation, et s'appuie en grande partie sur le mécanisme de désignation. Cette description séquentielle indique, sous une forme compréhensible, l'ensemble des modifications locales subies par la structure et autorise ainsi une analyse ultérieure des phénomènes reproduits.

3. Modèle événementiel d'animation topologique

Dans cette section, nous rappelons les grands principes de notre modèle d'animation événementiel, dont les détails sont fournis dans [LSM08]. Il est divisé en trois parties principales : le modèle structurel, le modèle événementiel et le modèle sémantique. Le modèle structurel porte l'information topologique et géométrique, en incluant une dimension temporelle. Le modèle événementiel vise à détecter et contrôler les modifications topologiques, entraînant la mise à jour du modèle structurel. Le modèle sémantique a pour objectif de représenter de façon immédiate l'histoire des entités topologiques. Nous détaillons ces trois modèles dans cette partie.

3.1. Modèle structurel

Pour représenter précisément les voisinages des entités topologiques, notre modèle structurel s'appuie sur un modèle topologique. Nous avons choisi les cartes généralisées (n-g-cartes) [Lie94], car elles sont définies de manière homogène en toute dimension, ce qui simplifie la définition du modèle et des opérations ainsi que l'extension du modèle dans des dimensions supérieures. De plus, l'élément abstrait manipulé, nommé brin, permet de désigner les sommets, les arêtes, les faces, les volumes sans information complémentaire. Nous rappelons les grands principes de ces structures ci-dessous et nous les exploitons dans une structure séquentielle que nous décrivons ensuite.

3.1.1. Structure topologique : n-g-cartes

Les n-g-cartes [Lie94] représentent les objets par leur bord. Elles modélisent les quasi-variétés cellulaires, orientées ou non, avec ou sans bord. Les objets géométriques sont subdivisés en cellules (sommets, arêtes, faces, etc.) reliées entre elles par des relations d'adjacence/incidence (Figure 3).

Définition 1 Une g-carte de dimension n, ou n-g-carte est un $(n+2)$-uplet $G = (B, \alpha_0, \ldots, \alpha_n)$ tel que :
– B est un ensemble fini de brins ;
– $\alpha_0 \ldots \alpha_n$ sont des involution sur B ;
– $\alpha_i\alpha_j$ [†] est une involution pour $0 \leq i < i+2 \leq j \leq n$.

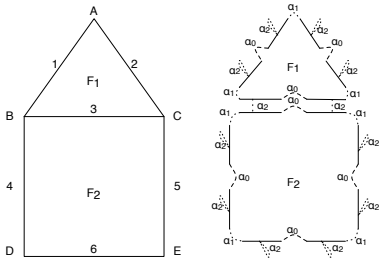

Figure 3: *Décomposition d'une maison schématique en 2-g-carte. (a) Modèle géométrique. (b) 2-g-carte associée ($\alpha_0\alpha_2$ est une involution).*

Définition 2 L'orbite $< \Phi > (b)$, pour un ensemble de permutations Φ, est l'ensemble des brins de B que l'on peut atteindre à partir de b par une composition quelconque des permutations de Φ. Si Φ est égal à l'ensemble de toutes les involutions $\alpha_0, \ldots, \alpha_n$ alors $< \Phi > (b)$ est une composante connexe de G incidente au brin b.

Dans une n-g-carte, toute i-cellule (cellule de dimension i) est obtenue par une *orbite* $< \alpha_0, \ldots, \alpha_{i-1}, \alpha_{i+1}, \ldots, \alpha_n >$.

Les orbites permettent principalement de parcourir la n-g-carte et d'associer des informations aux différentes i-cellules (Figure 4).

Définition 3 Une n-G-Carte $G = (B, \alpha_0, \ldots, \alpha_n)$ est fermée ssi $\forall i \in \{0, \ldots, n\}, b\alpha_i \neq b$.

[†] $\alpha_i\alpha_j$ est la notation qui correspond à la composition $\alpha_j \circ \alpha_i$. $b\alpha_i\alpha_j$ correspond à l'application de cette composition à un élément b de B.

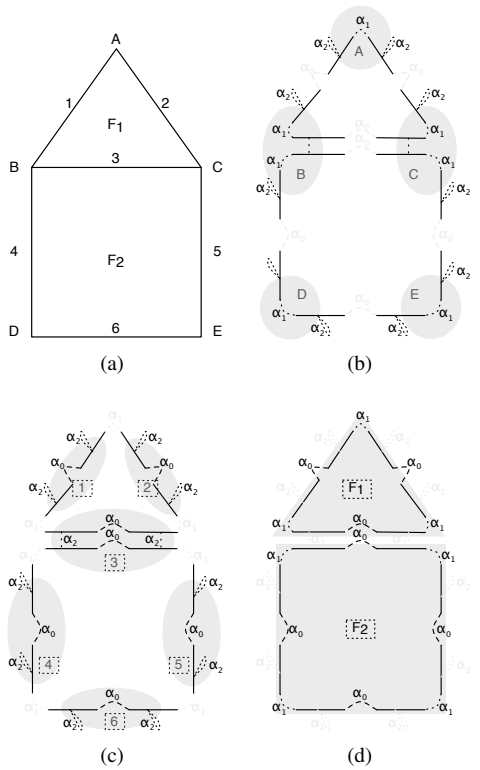

Figure 4: *Décomposition d'une maison schématique en 2-g-carte (a) en cellules de différentes dimensions. Les brins composant chaque cellule sont affichés sur fond gris. (a) Objet 2D à décomposer. (b) Orbites sommets (0-cellules). (c) Orbites arêtes (1-cellules). (d) Orbites faces (2-cellules).*

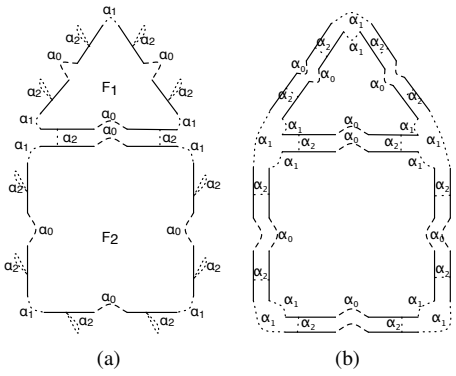

Figure 5: *(a) 2-g-carte ouverte. (b) 2-g-carte fermée.*

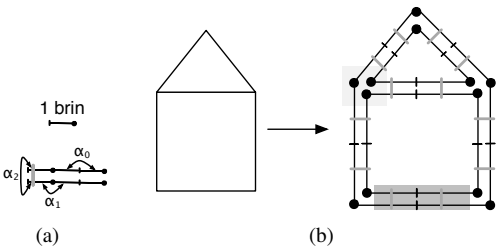

Figure 6: *(a) Convention utilisée pour la représentation d'un brin et des liaisons. (b) Exemple de deux faces collées entre elles et leur représentation par une 2-g-carte fermée. Sur fond gris clair : un exemple d'orbite sommet, sur fond gris foncé : un exemple d'orbite arête.*

La figure 5 présente une 2-g-carte ouverte par α_2 et sa version fermée. L'utilisation du modèle fermé permet de représenter une partition de l'espace. De plus, elle permet de simplifier la définition des opérations topologiques en évitant les cas particuliers.

Afin de simplifier et d'améliorer la lisibilité des figures, la convention de la figure 6 sera utilisée.

3.1.2. Modèle temporel par instants clefs

Le but de ce modèle est de représenter l'animation d'objets structurés comme une succession de modifications topologiques (Figure 7). Nous posons comme hypothèses qu'une modification topologique est instantanée et que plusieurs modifications distinctes peuvent se produire simultanément. Notre approche s'inspire largement de la technique d'animation par images-clefs [BW71] : un instant-clef correspond à un ensemble de modifications topologiques instantanées et une nouvelle n-g-carte est insérée dans le modèle pour chaque instant-clef.

Plus précisément, le modèle structurel représente l'historique des modifications topologiques au travers d'une succession de n-g-cartes fermées connexes (Figure 7) où chaque n-g-carte porte l'ensemble des modifications topologiques (supposées instantanées) à une date d. Ceci implique qu'entre deux n-g-cartes consécutives, aucune modification topologique n'intervient, seul le plongement des cellules peut changer. En théorie, la géométrie des entités est définie par une fonction $f : t \rightarrow \mathbb{R}^n$ dépendant du temps t et associée à une i-cellule. Cette approche apporte beaucoup de souplesse au niveau de la description géométrique, mais elle implique des événements topologiques délicats à détecter (voir section suivante). En pratique, nous avons opté pour un cadre géométrique plus restrictif, consistant à ne fournir un plongement qu'aux 0-cellule (les sommets), ce plongement pouvant être une fonction continue quelconque (ce choix étant motivé par des considérations de détection de collision, voir section 4.3.1). Les 1-cellules (les arêtes) sont linéaires pour un instant donné, i.e. un segment, et leur plongement temporel correspond ainsi à la surface spatio-temporelle décrite lors des mouvements et déformations du segment.

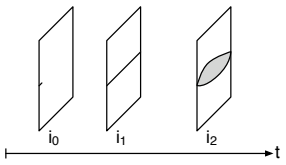

Figure 7: *Application du modèle à l'animation 2d + succession de 2-g-cartes ordonnées selon le temps t. À la date $t = i_0$, un sommet est créé sur une arête et une arête "pendante" (une extrémité de cette arête n'est pas liée au reste du modèle) lui est associée. À $t = i_1$, l'extrémité de l'arête pendante a rejoint une des arêtes bordant la 2-g-carte et fusionne avec cette arête, coupant la face originale en deux. À $t = i_2$, l'arête centrale est éclatée en face.*

3.2. Modèle événementiel

Le modèle structurel suffit à décrire une animation par une suite d'opérations topologiques et d'affectations de plongements sommets (si la géométrie du modèle est linéaire). Le problème est de déterminer les instants où ont lieu les changements topologiques, ainsi que les opérations topologiques à appliquer. Pour cela, un système à événements discrets tiré de [DZ93] est adjoint au modèle. Les événements sont générés soit à partir du scénario pour les événements initiaux déterminés par l'utilisateur, soit à partir du moteur de collisions qui prédit le contact entre entités. Chaque événement est inséré dans une file à priorité d'événements triés suivant le temps. Le traitement des événements est dépendant du contexte local (structure topologique, plongement géométrique et sémantique des entités) et se fait à l'aide d'algorithmes de reconnaissance et de remplacement de motifs topologiques et géométriques.

Dans [DZ93], un processus prédit les interactions entre objets. Pour cela, le système débute par une phase d'initialisation où il calcule, comme dans le cas d'une discrétisation du temps, toutes les interactions possibles entre couples (*acteur, cible*) d'objets. Pour chaque acteur, le système insère l'interaction potentielle entre le couple (*acteur, cible*) dont la date est la plus proche dans une *file à priorité* ordonnée suivant les dates. Cette phase d'initialisation passée, l'animation peut débuter. Le temps du système progresse par le traitement ordonné des événements, le temps courant correspondant toujours à la date du premier événement de la file à priorité. Dans ce cas, on défile l'événement et on vérifie ensuite sa validé, i.e. si les conditions de sa réalisation sont vérifiées. Si l'événement est effectivement valide, le système le traite. Le traitement se termine par un processus de détection de collisions. Il recalcule des interactions potentielles pour tous les couples ayant comme acteurs les objets modifiés.

La figure 8 présente deux cas d'événements invalides. Le premier se produit quand l'acteur (un sommet, une arête, etc.) rencontre une entité avant sa cible (dans la figure 8(a), A devrait rencontrer B en premier, mais rencontre C avant). Le second se produit quand la cible rencontre une entité avant l'acteur (dans la figure 8(b) B devrait rencontrer A, mais rencontre C). Ces événements invalides sont causés : soit par l'évaluation des collisions potentielles (dans la première figure, il est prévu que A et B rentrent respectivement en collision avec C et A) ; soit par le changement de trajectoire causé par un nouvel événement.

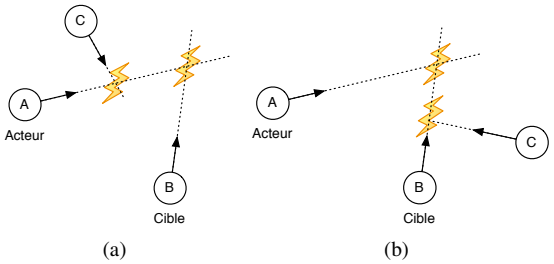

Figure 8: *Deux cas d'événements invalides. (a) L'événement entre A et B n'aura pas lieu car l'acteur A rencontre C avant B. (b) L'événement entre A et B n'aura pas lieu car la cible B rencontre C avant A.*

Nous tirons parti du fait que nous étudions ici l'évolution d'une partition de l'espace. Le collisions considérées sont donc des auto-collisions de la structure. Dans ce cas, les entités topologiques ne peuvent entrer en collision que si elles sont adjacentes ou incidentes à une même entité. Nous exploitons cette propriété pour limiter le nombre de tests de collision effectués.

3.3. Modèle sémantique

Le modèle sémantique vise deux objectifs : représenter de façon explicite les modifications subies par la structure au cours du temps et fournir une représentation de l'historique des entités topologiques. Pour remplir ces objectifs, nous utilisons un mécanisme de désignation des entités et un script d'opérations topologiques.

La désignation consiste à associer des noms aux cellules de la scène. Un nom est soit choisi par l'utilisateur lors des créations de cellules, soit déterminé automatiquement selon l'enchaînement des opérations mises en jeu. Ces noms sont utilisés comme paramètres des descriptions (fournies dans le scénario de l'utilisateur) et des opérations topologiques et géométriques issues de ces descriptions. La désignation est hiérarchique, ce qui facilite la description de la genèse d'une entité en lui associant comme parent l'entité dont elle est issue. Par exemple, l'opération d'insertion de n sommets sur une arête résulte en $n+1$ arêtes dont les désignations sont filles de la désignation de l'arête d'origine et ont, ainsi, comme préfixe la désignation de cette arête.

Le modèle structurel précédemment décrit représente

seulement le résultat géométrique et topologique de l'évolution : il ne décrit qu'implicitement les modifications subies. Pour représenter de façon explicite ces modifications, le modèle sémantique contient un script bas-niveau représentant la succession de modifications topologiques et les évolutions géométriques des entités de la structure. Ce script est généré par l'animation et retrace l'historique de construction du modèle pour permettre d'analyser, a posteriori, le résultat de l'animation. Cette fonctionnalité est utile à l'utilisateur pour retracer dans le détail les étapes de l'animation produite et déterminer quels paramètres modifier afin de produire une nouvelle animation selon d'autres hypothèses. Pour fabriquer ce script, le modèle sémantique récupère, lors du traitement des événements, la liste des opérations topologiques appliquées aux entités altérées. Cette liste forme une séquence chronologique d'opérations élémentaires qui s'appliquent sur la n-g-carte initiale au cours du temps. Chaque opération élémentaire est alors décrite dans le script par un appel de fonction, fonction qui s'applique sur une ou plusieurs entités d'une des 2-g-cartes du modèle temporel. Aucun contrôle de cohérence géométrique n'est effectué à ce niveau (ce contrôle est effectué dans l'étape précédente de génération du script), seul le modèle topologique est assuré d'être cohérent car toutes les opérations topologiques utilisées (détaillées ci-dessous) assurent la préservation de cette cohérence tout au long de l'animation.

Ce mécanisme forme ainsi un pont entre l'aspect « haut-niveau » (descriptif) de notre modèle, dans lequel l'utilisateur contrôle l'animation, et l'aspect « bas niveau » (structurel) sur lequel s'appliquent les décisions de l'utilisateur.

3.4. Génération de l'animation

La figure 9 montre le processus général de l'animation. En (a-b), l'utilisateur écrit un scénario pour décrire une animation (cette description est donc spécifique à un domaine d'application donné). Le scénario est décomposé en informations structurelles et en opérations sur des entités désignées. En (b-c), la description est analysée et convertie en une séquence d'événements. Le traitement de ces événements génère un script d'évolutions bas niveau où chaque modification topologique est enregistrée. Ce processus utilise les informations structurelles et sémantiques. En (c-d), le script bas niveau est interprété par le modèle topologique avec l'assistance du modèle sémantique qui lie les objets désignés aux entités topologiques. En (d), les opérations structurelles sont appliquées.

4. Cas $2D$

La section précédente rappelle le modèle général d'animation de structure topologique. Cette section présente une instanciation de ce modèle en deux dimensions à travers la spécification du plongement géométrique, de la désignation, des événements et du script de manipulation de la structure topologique.

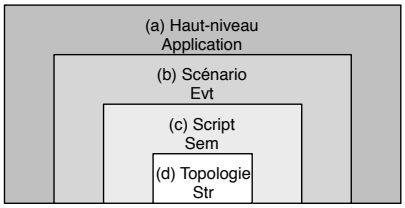

Figure 9: *Utilisation des trois modèles (événementiel, sémantique, structurel) dans le processus de création d'animations (Evt : Événementiel ; Sem : Sémantique ; Str : Structurel)*

4.1. Plongement géométrique

Comme précisé dans la section 3.1.2, nous avons choisi d'utiliser un plongement des 0-cellules. Cela consiste, en $2D$, à attribuer une fonction continue $f : t \rightarrow \mathbb{R}^2 = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix}$ à chaque sommet. La forme d'une arête liant deux sommets est déduite par interpolation linéaire de leurs coordonnées. La forme des faces est déduite par l'assemblage des arêtes la constituant.

4.2. Désignation

Les entités géométriques sont désignées par un nom. Ce nom sert de paramètre au scénario et au script bas niveau. En $2D$, seules les 1-cellules (arêtes) sont désignées car, à partir des arêtes, il est simple de désigner soit un sommet, soit une face. En effet, le modèle des n-g-cartes est de type *B-Rep*, ce qui se traduit en 2D par la propriété suivante : une arête est incidente à au plus deux faces et à deux sommets. Dans une 2-g-carte fermée, chaque arête est formée de quatre brins (voir figure 5). Un de ces brins porte l'identifiant de l'arête ce qui permet de la désigner (Figure 10). Pour désigner les sommets, nous adjoignons à la désignation la mention *begin* ou la mention *end*. Le désignation *a_name* + *begin* désigne le sommet se trouvant du côté du brin (plus précisément, le brin désigné appartient à l'orbite de ce sommet). *a_name* + *end* désigne l'autre sommet. Pour désigner les faces, nous utilisons les mentions *right* et $left$. Considérant une arête orientée du sommet *begin* vers le sommet *end*, nous pouvons considérer alors une face *a_name* + *right* à droite de l'arête et une face *a_name* + $left$ à gauche de l'arête. Pour des raisons de commodité d'implantation, seuls deux brins sur les quatre d'une arête peuvent porter une désignation (de telle sorte qu'un brin porteur d'une désignation appartient forcément à la face *right*). Ceci revient à orienter les faces dans le sens horaire. Notons que puisque la désignation s'opère à partir d'une simple arête, il existe plusieurs moyens de désigner les sommets et les faces (le nombre de façons de les désigner est respectivement égal au degré du sommet et au nombre d'arêtes formant le bord de la face).

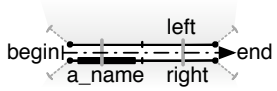

Figure 10: *Mécanisme de désignation des 0,1,2-cellules à partir du brin "a_name" portant la désignation de la 1-cellule.*

4.3. Événements

Les sommets de la scène évoluent librement dans un espace $2D$, ce qui peut entraîner des collisions entre cellules, constituant avec les événements initiaux définis par l'utilisateur l'ensemble des événements possibles. En deux dimensions, seuls deux types de collisions peuvent se produire, les collisions sommet/sommet et les collisions sommet/arête. En effet, avec un plongement temporel de sommets et un système dans un état cohérent, les collisions arête/arête n'ont pas besoin d'être détectées, car des collisions sommet/sommet ou sommet/arête sont détectées avant ; et ce, même dans le cas dégénéré où les arêtes sont parallèles (dans le cas d'un plongement non linéaire, les collisions arête/arête doivent être détectées). Nous séparons le traitement des collisions sommet/sommet, des collisions sommet/arête pour traiter les cas dégénérés de façon explicite [MKF03]. De plus, le système de collision se basant sur une résolution d'équations formelle, la différenciation entre le cas sommet/sommet et sommet/arête est immédiate. Partant de ce constat, nous pouvons définir des classes de traitement associées aux différents cas. Cette classification permet de généraliser et de simplifier la description des traitements associés aux événements.

4.3.1. Détection

La détection des auto-collisions de la structure consiste pour tout sommet sur lequel un mouvement est appliqué, à calculer ses potentielles interactions avec son voisinage direct, c'est-à-dire avec les faces qui lui sont incidentes. On utilise ainsi l'information topologique pour limiter le nombre de couples d'entités à tester. Les tests d'intersection se résument à chercher la plus proche collision (dans le temps) (1) entre chaque sommet s_0 modifié et les arêtes appartenant aux faces incidentes à s_0 (*faces_sommet*(s_0)), (2) entre chaque arête a incidente à s_0 (*aretes_sommet*(s_0)) et chaque sommet appartenant aux faces incidentes à s_0 (*sommets_face(faces_aretes(a))*), et (3) entre chaque sommet adjacent au sommet s_0 (*sommets_sommet*(s_0)) (i.e. longueur de l'arête courante devenant nulle). L'algorithme 1) récapitule ces différentes étapes. Des fonctions de la forme x_y sont utilisées, où x et y peuvent représenter des sommets, des arêtes ou des faces, et retournent l'ensemble des cellules x incidentes à une cellule y considérée. *dmintrouvee* correspond à la date de la plus proche intersection, *dtmp* correspond à la date trouvée en cas d'intersection et *inter_trouvee* correspond à la plus proche intersection trouvée. *inter_trouvee* est un triplet contenant le type d'intersection trouvé (*ve* : sommet/arête ; *vv* : sommet/sommet) ainsi que les deux cellules mises en causes.

Entrées: $s_0 \in G_2, t_{min}$
inter_trouvee ← nil
// (1) recherche les intersections du sommet s_0 avec les arêtes appartenant aux faces incidentes à s_0
dmintrouvee ← ∞
pour chaque face $f \in faces\_sommet(s_0)$ **faire**
  pour chaque arête $a \in (aretes\_face(f) - aretes\_sommet(s_0))$ **faire**
    si intersection_arete_sommet(a, s_0, dmintrouvee, dtmp) **alors**
      si dmintrouvee > dtmp **alors**
        inter_trouvee ← ("ve", s_0, a)
// (2) recherche les intersections des arêtes a incidentes au sommet s_0 par rapport aux sommets appartenant aux faces incidentes à s_0
pour chaque arête $a \in aretes\_sommet(s_0)$ **faire**
  pour chaque sommet $s \in sommets\_face(faces\_arete(a)) - sommets\_arete(a)$ **faire**
    si intersection_arete_sommet(a, s, dmintrouvee, dtmp) **alors**
      si dmintrouvee > dtmp **alors**
        inter_trouvee ← ("ve", s, a)
// (3) recherche les intersections des sommets s adjacents au sommet s_0
pour chaque sommet $s \in sommets\_sommet(s_0)$ **faire**
  si intersection_sommet_sommet(s_0, s, dmintrouvee, dtmp) **alors**
    si dmintrouvee > dtmp **alors**
      inter_trouvee ← ("vv", s_0, s)
retourner inter_trouvee

Algorithm 1: Détection de la plus proche collision potentielle (plongement sommet, $2D$).

Compte tenu du plongement affecté uniquement aux sommets, déterminer l'instant de collision entre un sommet C et une arête AB revient à détecter l'intersection entre un point et un segment de taille variable (déformable). Provot [Pro97] a proposé de détecter cette intersection en vérifiant les instants t où les points A, B et C sont alignés. En $2D$, cela revient à détecter que l'angle entre la normale $\vec{n}(t)$ à $\vec{AB}(t)$ et $\vec{AC}(t)$ doit être nul, autrement dit que $\vec{n}(t).\vec{AC}(t) = 0$. Cette formulation permet de ramener le calcul à une équation à une inconnue. De plus, l'équation peut facilement être rame-

née à une inéquation permettant de traiter les erreurs numériques en recherchant des solutions pour $\vec{n}(t).\vec{AC}(t) < \varepsilon$. Cependant, pour mener formellement le calcul à terme, il faut faire des hypothèses sur le mouvement des points, généralement un mouvement rectiligne uniforme. Cette hypothèse sur une certaine classe de mouvement des points est cependant trop restrictive. Par exemple, un point se déplaçant le long d'une arête dont les sommets ont un mouvement rectiligne uniforme n'a généralement pas un mouvement rectiligne uniforme, le point considéré sort donc de la classe des mouvements autorisés. Nous avons donc opté pour une approche plus générale : nous examinons s'il existe une intersection entre C et la droite D support de AB, puis testons si l'intersection appartient à l'arête. Il faut alors résoudre $A(t) + \alpha_1 \vec{AB}(t) = C(t)$, i.e. trouver les instant t^i tels qu'il existe α_1 vérifiant l'équation, puis vérifier pour chaque t^i que $0 \leq \alpha_1^i \leq 1$. Pour résoudre l'équation dans le cas général, nous utilisons actuellement un moteur de résolution d'équations formel.

Concernant le cas dégénéré de collision entre les sommets A et B, l'approche est similaire. Nous recherchons les instants t^i respectant $A(t) = B(t)$, via le moteur de résolution d'équations formel.

4.3.2. Traitement

Les collisions sommet/arête(Figure 11(a)) engendrent trois types de réponses. La première est de simplement coller les entités en collision (Figure 11(b) - Cas 1). La deuxième est de donner la priorité au sommet qui traverse donc l'arête en la découpant (Figure 11(c) - Cas 2). La troisième est de donner la priorité à l'arête qui sépare les arêtes incidentes au sommet (Figure 11(d) - Cas 3).

Les collisions sommet/sommet entraînent également trois types de réponses. Si les deux sommets sont liés par une arête, un glissement topologique d'un sommet sur un autre est réalisé (Figure 2(a)). Sinon, soit les entités en collision sont collées comme dans le cas d'une collision sommet/arête (Figure 12(b) - Cas 1) ; soit une priorité est donnée à l'un des deux sommets (Figure 12(c) - Cas 2). Cette seconde approche pose un problème de décision si le sommet non prioritaire est de degré supérieur à deux, car il est difficile de prévoir la structure topologique résultante ainsi que le plongement des nouveaux sommets dans le cas de trajectoires non rectilignes (Figure 12(d) - Cas 3). En effet, dans le cas d'une trajectoire rectiligne, nous pouvons calculer le résultat de l'intersection au moment $t + \varepsilon$ et en déduire la répartition des arêtes incidentes au sommet v sur les arêtes $e1$ et $e2$ (Figure 13). Dans le cas d'une trajectoire non rectiligne, le calcul des nouveaux plongements de sommets et de la topologie est délicat car la répartition des arêtes peut changer au cours du temps comme par exemple pour l'arête e (Figure 14). Pour déterminer comment séparer l'ensemble des arêtes, on peut utiliser la tangente de la trajectoire au point d'impact. Le changement de répartition d'arêtes s'effectue par glissements topologiques des sommets après collisions sommet/sommet liés par une arête.

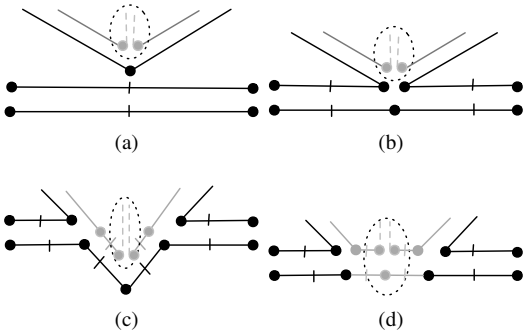

Figure 11: *Cas de collisions sommet/arête. Les pointillés représentent la répétition du motif encerclé. Par exemple, dans la figure (a), la zone en pointillés représente un nombre quelconque d'arêtes incidentes au sommet qui vont entrer en collision avec l'arête horizontale. (a) État avant collision. (b) Cas 1 : le sommet se colle à l'arête. (c) Cas 2 : le sommet est prioritaire sur l'arête. (d) Cas 3 : l'arête est prioritaire sur le sommet.*

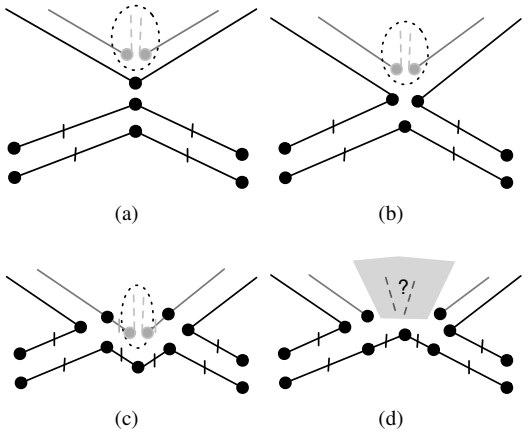

Figure 12: *Cas de collisions sommet/sommet (les pointillés représentent la répétition du motif encerclé). (a) État avant collision. (b) Cas 1 : le sommet se colle à l'autre sommet. (c) Cas 2 : le sommet supérieur est prioritaire sur l'autre de degré ≤ 2. (d) Cas 3 : le sommet inférieur est prioritaire sur l'autre de degré > 2 : les modifications topologiques du sommet supérieur dépendent du nombre d'arêtes qui lui sont incidentes et des trajectoires des sommets inférieur et supérieur.*

4.4. Script bas niveau

Les opérations appliquées au modèle sont inscrites dans le script bas-niveau via un ensemble d'opérations élémentaires que nous précisons dans la suite. Pour l'implantation,

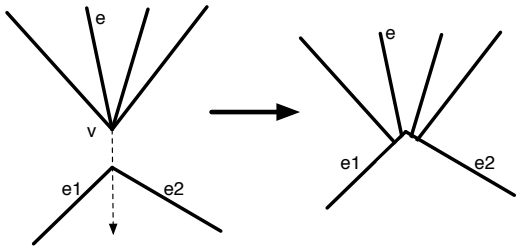

Figure 13: *Résolution du cas 3 de la figure 12(d) dans le cas rectiligne.*

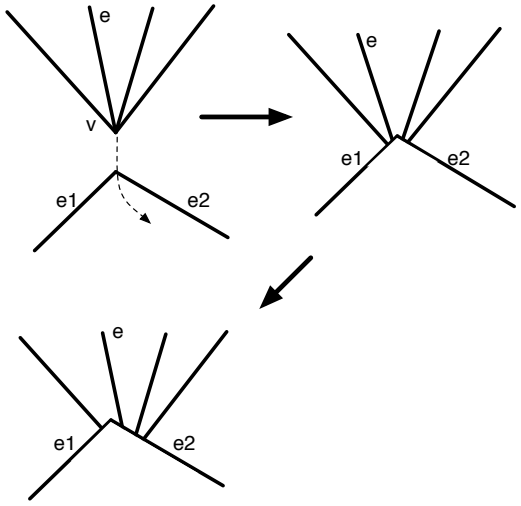

Figure 14: *Problème du cas 3 de la figure 12(d) dans le cas non rectiligne. Après l'intersection de v (dont la trajectoire est symbolisée par la flèche en pointillé) à la jonction des arêtes e1 − e2, un glissement des arêtes s'effectue en suivant la trajectoire de v.*

ce script est écrit en langage lua (http://www.lua.org) et il peut être interprété par l'application (appliqué à la carte initiale, le script permet de générer à nouveau l'intégralité du modèle structurel).

Dans le script, il est possible de préciser les plongements géométriques associés aux cellules. Ils sont de deux types : $f : t \rightarrow \mathbb{R}^2$ pour les 0-cellules ou $f : s,t \rightarrow \mathbb{R}^2$ (s une coordonnée curviligne) pour les 1-cellules [LSM06].

Nous détaillons dans la suite les fonctions de modification en $2D$ utilisables dans le script. Ces fonctions doivent permettre de créer, supprimer, fusionner les diverses entités topologiques. Nous avons identifié les transformations élémentaires nécessaires au script et permettant les transformations nécessaires à nos animations. Tout d'abord, il faut pouvoir créer une arête e, ce qui ajoute une nouvelle entrée dans l'arbre de désignation du modèle. La désignation étant hiérarchique, il est tout à fait possible de lui spécifier une entité parente. À l'inverse, on doit également pouvoir effacer une arête, ce qui la déréférence de l'arbre de désignation. Pour cet effacement, il existe deux approches duales : soit on supprime l'arête, ce qui fusionne les faces qui lui sont incidentes, soit on contracte l'arête, ce qui fusionne ses sommets extrémités. Par ailleurs, pour connecter des arêtes entre elles et plus exactement pour fixer le sommet d'une arête à un autre sommet de la carte, il faut « identifier » les sommets. Pour la détacher, il faut « déconnecter » l'arête. Ces deux fonctions modifient seulement les relations d'adjacence entre les entités. Enfin, on peut insérer n nouveaux sommets sur une arête e. Ceci a pour effet de découper l'arête désignée par e en $n+1$ arêtes désignées par e_i (avec $i \in [0;n]$).

En résumé, pour contrôler l'évolution d'une subdivision, les opérations topologiques à inclure dans le script sont les suivantes :

- création d'une arête désignée, avec, en option, le nom d'une « arête parente » ;
- contraction d'une arête ;
- suppression d'une arête ;
- identification de sommets ;
- déconnexion d'une arête ;
- insertion de n sommets sur une arête.

Ces fonctions assurent la cohérence entre les modèles topologique et sémantique. Cependant, l'utilisation seule des primitives *déconnexion* et *création* une arête pourrait engendrer un modèle qui ne remplit pas la contrainte de connexité des g-cartes. Pour cette raison, ces deux fonctions sont seulement utilisées dans des étapes intermédiaires des modifications topologiques et la contrainte de connexité est toujours à assurer à la fin de ces étapes. En effet, ces fonctions sont systématiquement associées à l'utilisation des fonctions *identification*, *suppression* et *contraction*.

5. Application à la géologie

Après avoir décrit notre modèle d'animation, nous montrons maintenant son application dans l'étude de quatre phénomènes géologiques classiques et assez intuitifs : la sédimentation, l'érosion, la création de faille et le glissement de blocs géologiques. Ces phénomènes sont composés par la suite dans un scénario. Après avoir présenté ces phénomènes, nous décrivons comment ils sont intégrés dans le modèle d'animation. Pour chaque phénomène, la méthode générale consiste à définir les événements initiaux. Ces événements permettent au système de débuter l'animation du phénomène. D'autres événements apparaissent durant l'exécution du phénomène et doivent également être définis. Les événements entraînant des modifications topologiques et/ou de plongements doivent être définis grâce à des motifs et des algorithmes de transformation. Chaque événement exige au moins un motif et sa transformation associée.

5.1. Travaux antérieurs

Un modèle du sous-sol est généralement construit à partir de mesures brutes issues de sondages, forages, etc. Ces données bruitées doivent cependant être traitées et améliorées [BNDP05]. De nombreux logiciels industriels exploitent les mesures effectuées, pour modéliser le sous-sol en une partition de cellules 3*D* (par exemple GOCAD [Goc, Mal02] et RML [RML, FHHR98]). Les modeleurs manipulent ensuite ces cellules soit en tant que "couches géologiques" réunissant les cellules partageant les mêmes propriétés géophysiques ; ou bien en tant que "blocs géologiques", i.e. des parties (éventuellement disjointes) d'une couche s'étant scindée au cours du temps.

Schneider [Sch02] délimite les couches géologiques par des intersections de surfaces paramétriques. Une autre approche pour l'intersection est proposée par Guiard [Gui06]. Elle consiste à utiliser un plongement linéaire ; les surfaces paramétriques sont alors modélisées par subdivisions de surfaces composées de faces planes aux arêtes rectilignes. Les failles et les horizons peuvent s'intersecter sans forcément couper un bloc géologique, ce qui permet par exemple de représenter des failles pendantes (i.e. des failles dont au moins une extrémité s'arrête à l'intérieur d'un bloc). Cette méthode repose sur le modèle topologique des n-G-Cartes étendu [Sch02]. Les approches de Schneider [Sch02] et Guiard [Gui06] sont statiques (i.e. décrivent seulement l'état du sous-sol à un instant donné) et ne prennent pas en compte l'historique de formation des couches géologiques et leur évolution.

Perrin [Per98] propose une description des causes et effets de l'évolution des couches géologiques, à l'aide d'une "syntaxe géologique" qui permet de formaliser des phénomènes géologiques et leur succession au cours du temps. Ses règles assurent la consistance géologique d'un modèle 3D. Elles sont intégrées dans un *Schéma d'Évolution Géologique* (SEG), décrivant une interprétation géologique du modèle à construire. Il se présente sous la forme d'un graphe acyclique orienté. Les nœuds de ce graphe représentent des surfaces délimitant les blocs géologiques ou des sous-SEG correspondant à un niveau de détail plus fin (par exemple, une faille au niveau n du SEG peut représenter un réseau de failles décrites au niveau $n+1$). Les arcs représentent soit une relation chronologique (par exemple, une faille est "antérieure à" une autre faille) ; soit une relation topologique basée sur les propriétés géologiques des surfaces qui s'intersectent (par exemple, une faille "coupe" un horizon). Les règles de syntaxe géologique permettent de préciser de quelle manière les nœuds du graphe interagissent. Les informations transportées par le SEG permettent donc de définir une chronologie des événements mais elles sont insuffisantes pour représenter une animation d'évolution de couches géologiques. En effet, les données temporelles précises ne sont pas présentes, et la chronologie fournie par le SEG ne contient pas de durée. De plus, les liens de cause à effet entre les événements décrits sont inexistants : ce modèle ne permet pas, par exemple, de déterminer comment une couche à été créée ou comment sa forme a évolué.

En résumé, les représentations courantes se contentent principalement de modéliser et de structurer un sous-sol statique. Or, le géologue se base généralement sur des hypothèses d'évolutions du sous-sol pour en déduire sa structure et sa composition, il peut ainsi rechercher la présence de couches particulières, de réservoirs, etc.

5.2. Modèle de sédimentation

La sédimentation est l'ensemble des processus par lesquels des particules en suspension se déposent. Selon les principes de stratigraphie énoncés par Stenon (1669), les sédiments se déposent en couches à peu près horizontales.

Nous considérons un modèle de sédimentation suivant une déposition strictement horizontale (nous pourrions cependant utiliser des formes non strictement horizontales en suréchantillonnant la surface de sédimentation comme nous le faisons pour l'érosion en section 5.3). La scène considérée représente une coupe transversale du sous-sol. La sédimentation débute par le remplissage des zones les plus basses. Ensuite, le niveau de sédimentation s'élève et si deux couches issues d'une même sédimentation se rencontrent, elles fusionnent.

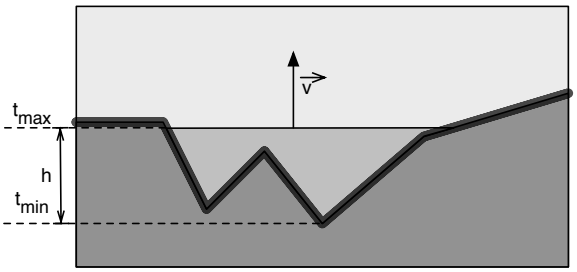

Figure 15: *Paramètres du modèle de sédimentation ($\vec{v}$ est le vecteur représentant la vitesse de sédimentation).*

La sédimentation est définie par cinq paramètres (Figure 15) : un préfixe pour désigner les différents morceaux créés par la sédimentation, une zone de sédimentation délimitée par deux sommets *(begin_interface, position_on_interface)* et *(end_interface, position_on_interface)*, une date de début *t_min*, une date de fin *t_max* et une hauteur h. Ce modèle simplifié considère que la vitesse de sédimentation est constante (ie. : $v = \frac{h}{t\_max - t\_min}$).

Ce phénomène peut être décrit dans le scénario d'évolutions géologiques à l'aide de la fonction suivante :

```
actionSedimentation(t_min, t_max − t_min,
    vertical_velocity, "prefix_interfaces",
    "begin_interface", position_on_interface,    — sommet 1
    coverage_direction,
    "end_interface", position_on_interface);     — sommet 2
```

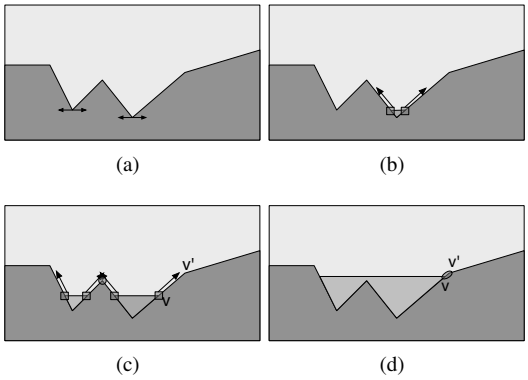

Figure 16: *Étapes d'une sédimentation. (a) Recherche des minima locaux. (b) Création de la première face et glissement de ses extrémités le long des parois de la couche. (c) Création d'une nouvelle face et glissement de ses extrémités. Le sommet v se déplace vers le sommet v'. (d) Fusion des deux faces avec glissement des extrémités. Après cette étape, v et v' rentreront en collision, puis v glissera sur v' et continuera son déplacement le long de la couche.*

Les paramètres $t\_max - t\_min$ et $vertical\_velocity$ correspondent respectivement à la durée du phénomène et à la vitesse de montée verticale de la couche en création. Ces paramètres permettent de déduire facilement la hauteur de sédimentation. Nous désignons deux arêtes et une position sur ces arêtes afin de définir les sommets 1 et 2 permettant de préciser l'interface géologique sur laquelle la sédimentation se dépose. Le paramètre "coverage_direction" permet de discriminer les deux zones possibles entre ces sommets. Ces deux zones correspondent aux deux sens de parcours d'une face. Si ce paramètre est positionné sur $forward$, le système parcourt la face de sédimentation dans le sens *begin* vers *end* de l'arête (dépendant de la désignation, voir section 4.2). Le paramètre *prefix_interfaces* permet au système de savoir comment désigner les arêtes provenant du phénomène de sédimentation.

Le traitement de ce phénomène par le système débute par la recherche des minima locaux dans la zone de sédimentation (Figure 16(a)). Ensuite, les dates de début de remplissage de ces minima sont calculées. Les événements de *création d'interfaces* associés à chaque minimum sont ajoutés à la file d'événements.

Le traitement de la création d'interface consiste à insérer une arête dans la face de sédimentation au niveau de la zone du minimum courant (Figure 16(b)). Les extrémités d'arêtes sont plongées par une fonction d'interpolation entre les coordonnées du minimum local et le sommet suivant, pour glisser le long du bord de la couche tout en tenant compte de la vitesse de sédimentation.

Lors du glissement, les sommets peuvent rencontrer de nouveaux sommets. Ces événements sont considérés comme des collisions sommet / sommet liés par une arête, et leur date est calculée et ajoutée à la file. Ces événements peuvent être traités de deux manières suivant le contexte local. Soit les deux sommets en collision sont issus du même phénomène de sédimentation (Figure 16(c)) et sont fusionnés, ce qui entraîne une fusion de faces (processus inverse de celui décrit à la Figure 2(b)) ; soit un des sommets provient de la sédimentation et l'autre appartient à la couche du sous-sol (Figure 16(d)) et dans ce cas, le premier sommet v glisse sur le second v' et continue son chemin sur l'arête suivante (Figure 2(a)). Le glissement de sommet peut entraîner d'autres événements de collisions sommet / sommet, qui sont donc insérés dans la file. Ce processus est répété jusqu'à la fin du phénomène de sédimentation.

Pour distinguer les deux cas de collisions, nous avons besoin de deux motifs de collision sommet / sommet : un pour la fusion de faces (Figure 17(a)) et un autre pour le glissement de sommet (Figure 17(b)). Dans les deux cas, la taille s de l'arête reliant les sommets impliqués dans la collision est nulle. Le premier motif correspond à la détection de deux arêtes consécutives de taille nulle qui n'appartiennent pas au processus de sédimentation (et sont associées au prédicat " !Sed") incidentes à deux arêtes, qui sont issues de ce processus (et donc associées au prédicat "Sed"). L'algorithme de transformation consiste à contracter les deux arêtes de taille nulle ($n2$, $n3$) et de fusionner les deux interfaces ($sed01\_0$, $sed01\_1$), entraînant la fusion des faces de sédimentation. Le second motif correspond à la détection d'une arête de taille nulle ($n2$). L'algorithme de transformation consiste à contracter l'arête de taille nulle et à insérer un nouveau sommet v sur l'arête suivante $n1$, coupant cette arête en deux nouvelles arêtes respectivement désignées $n1\_0$ et $n1\_1$. Ensuite, l'extrémité de l'interface de sédimentation est déliée puis identifiée à v.

5.3. Modèle d'érosion

L'érosion est l'ensemble des processus de dégradation et de transformation du relief.

Notre approche permet de traiter toutes les formes que peut prendre le phénomène d'érosion. Nous choisissons ici de considérer un profil en forme de cuve pour simplifier le paramétrage de ce phénomène. Nous supposons que toutes les couches s'érodent à la même vitesse et de façon uniforme. Pour obtenir un profil en forme de cuve, les arêtes érodées sont ré-échantillonnées suivant un pas défini par l'utilisateur. L'érosion débute par la déformation d'une surface incidente à une interface (un chemin d'arête) dont on indique les deux arêtes extrémales et des points situés sur ces arêtes délimitant la zone d'érosion. Si la surface d'érosion touche une autre surface, celle-ci est modifiée en suivant le profil de la surface d'érosion (Figure 18).

L'érosion est définie par quatre paramètres : la zone d'érosion délimitée par deux sommets (dans la figure 18, la zone

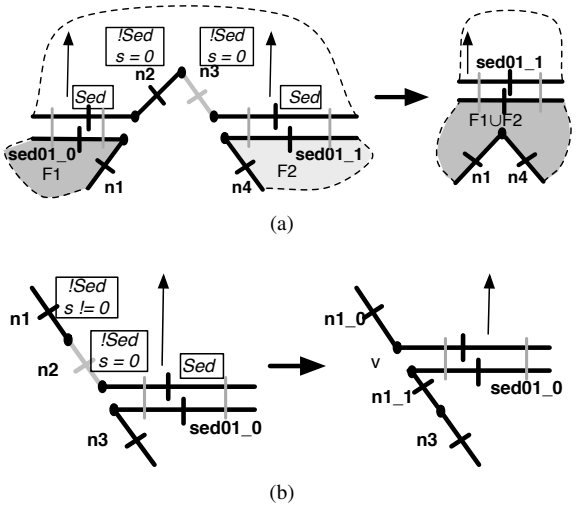

Figure 17: *Motifs de collisions pour la sédimentation (les collisions sont représentées en gris, les prédicats sont encadrés). (a) Fusion des faces F*1 *et F*2 *issues du même processus de sédimentation (les arêtes sed*01_0 *et sed*01_1 *fusionnent en sed*01_1*). Les lignes en pointillés symbolisent les faces associées aux brins représentés en trait plein. (b) Glissement d'un sommet sur un autre sommet (n*2 *se contracte, n*1 *se divise en n*1_0 *et n*1_1*).*

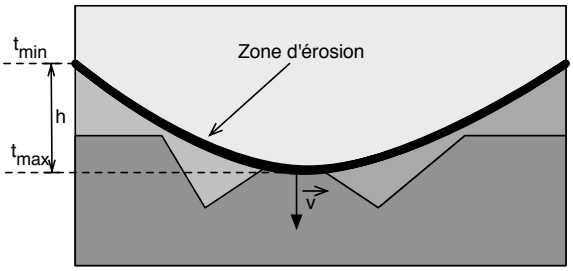

Figure 18: *Paramètres du modèle d'érosion.*

est délimitée par les bords de la scène), une date de début *t_min*, une date de fin *t_max* et une hauteur *h*.

Ce phénomène peut être décrit dans le scénario d'évolutions géologiques à l'aide de la fonction suivante :

```
actionErosion(t_min, t_max − t_min,
    "begin_interface", position_on_interface,      — sommet 1
    coverage_direction,
    "end_interface", position_on_interface,        — sommet 2
    vertical_velocity,
    "prefix_interfaces",
    sampling);
```

Les paramètres *t_max − t_min*, *vertical_velocity*, *coverage_direction* et *prefix_interfaces* ont le même sens que ceux de la fonction *actionSedimentation*. Le paramètre *sampling* correspond au pas de découpage de la surface érodée permettant d'approcher le profil d'érosion.

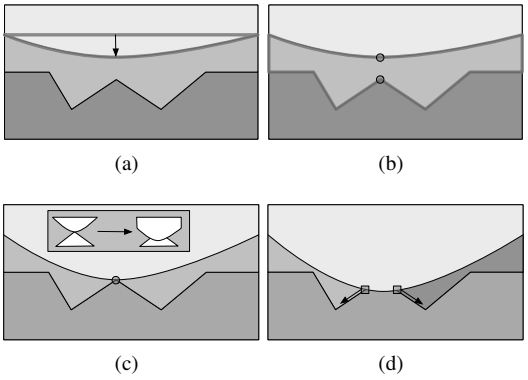

Figure 19: *Étapes d'une érosion. (a) Suréchantillonnage de la surface d'érosion et mise à jour du plongement des sommets. (b) Recherche de la date de collision la plus proche : les sommets mis en jeu dans cette collision sont symbolisés par des disques. (c) À l'instant de la collision, traitement de l'événement suivant le contexte : ici, la surface d'érosion est prioritaire sur le reste de la scène. (d) La surface d'érosion a érodé le pic : une nouvelle face est créée et les points de contact glissent le long des couches géologiques.*

Le traitement de ce phénomène débute par le suréchantillonage de l'interface d'érosion (Figure 19(a)). Pour cela, il suffit d'insérer de nouveaux sommets dont le nombre correspond au paramètre *sampling* de *actionErosion* sur l'arête considérée et de mettre à jour le plongement géométrique. Les trajectoires de ces sommets sont calculées pour donner à l'interface la forme d'une cuve. Les événements initiaux consistent donc à effectuer plusieurs insertions de sommets (si nécessaire) et à mettre à jour leur plongement.

Durant le mouvement des sommets, des collisions peuvent se produire entre la surface érodée et les couches du sous-sol. Après le calcul des trajectoires des sommets de l'interface érodée, les dates de ces collisions sont prédites et les événements associés sont ajoutés à la file. Le traitement de ces événements dépend du contexte local (Figure 19(c)). Soit l'interface d'érosion est prioritaire sur les autres interfaces du modèle et érode donc ces interfaces, soit l'interface d'érosion est éclatée en plusieurs morceaux après la collision. Cette notion de priorité est décrite explicitement dans [Per98] et adaptée à la modélisation de structure géologique dans [BPRS01]. Durant la déformation de l'interface d'érosion, les sommets peuvent rentrer en collision soit avec des sommets, soit avec des arêtes. Les Figures 20 et 21 décrivent les collisions sommet / arête possibles et leur traitement dépendant de la nature des entités mises en cause (les surfaces d'érosion ont comme prédicat "Ero", les autres " !Ero"). Les collisions sommet / sommet, où les sommets ne sont pas liés par une arête, sont traités comme des cas particuliers de ceux montrés en Figure 21. Le plongement

géométrique de chaque nouveau sommet est calculé de telle manière que le mouvement de ce sommet corresponde à un glissement le long des arêtes initiales.

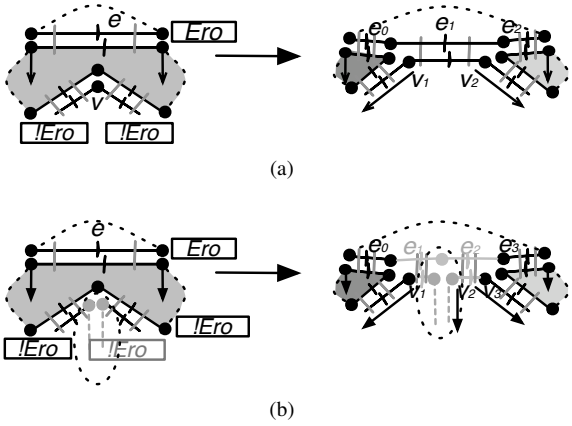

Figure 20: *Traitement de la collision entre une arête d'érosion et un sommet du modèle au niveau topologique et géométrique. a) Intersection d'une arête avec un sommet de degré 2 (deux sommets sont insérés sur l'arête e, elle se divise en e_0, e_1, e_2). b) Cas général d'intersection avec un sommet de degré $n > 2$ (e se divise en $e_0, \ldots, e_n$).*

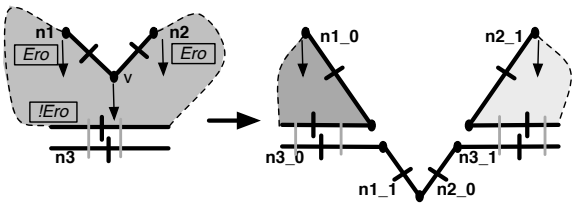

Figure 21: *Traitement de la collision d'un sommet de la surface d'érosion avec une arête du sous-sol (le degré du sommet n'a ici aucune incidence) : n1 et n2 se divisent respectivement en n1_0, n1_1 et en n2_0, n2_1.*

La figure 20 illustre le traitement d'une collision entre une arête e appartenant à la surface d'érosion et un sommet v de la scène. Si le degré de v est égal à 2 (Figure 20(a)), deux nouveaux sommets v_1, v_2 sont insérés sur e, les arêtes incidentes à v sont déliées, puis liées aux sommets précédemment insérés. Le plongement de ces sommets est mis à jour pour correspondre à un glissement le long de l'arête e. Sinon, si le degré de v est $n > 2$, n nouveaux sommets sont insérés et toutes les arêtes incidentes à v d'abord déliées puis reliées aux nouveaux sommets en suivant l'ordre donné par l'orbite sommet.

La figure 21 illustre le traitement d'une collision entre un sommet v issu d'une interface d'érosion et une arête $n3$ du sous-sol. Le traitement débute par l'insertion d'un sommet sur l'arête $n3$, ce qui découpe cette dernière en $n3\_0$ et $n3\_1$

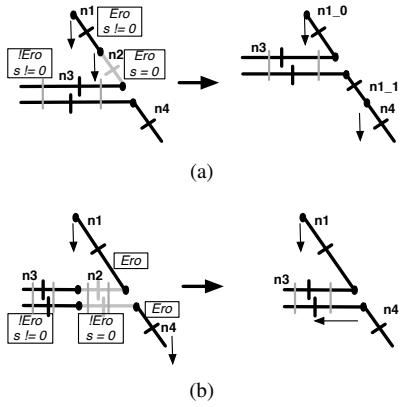

Figure 22: *Motifs de collisions de deux sommets liés par une arête dans le cas de l'érosion. (a) Glissement de sommets (n2 se contracte, n1 se divise en n1_0 et n1_1). (b) Contraction de l'arête n2.*

(dans le cas d'une collision sommet / sommet où les sommets ne font pas partie de la même arête, nous omettons simplement l'insertion d'un sommet, tout en poursuivant le reste du traitement). Ensuite, un sommet est inséré sur les deux arêtes $n1$ et $n2$ incidentes à v et qui partagent la face contenant $n3$ et v : $n1$ et $n2$ sont donc respectivement divisées en $n1\_0$ et $n1\_1$ d'une part, et en $n2\_0$ et $n2\_1$ d'autre part. Enfin, les relations topologiques entre ces arêtes sont mises à jour de manière à déconnecter $n1\_1$ (resp. $n2\_0$) de $n1\_0$ (resp. $n2\_1$) et à la relier à $n3\_0$ (resp. $n3\_1$). Enfin, nous délions les deux nouvelles arêtes de $n3$ que nous identifions avec les sommets insérés sur $n1$ et $n2$.

Notons qu'après le traitement de la collision, les sommets de contact de la surface d'érosion peuvent glisser le long de la surface d'érosion (Figure 19(d)). Cette évolution est similaire au glissement de sommets de long d'arêtes décrit dans le cas de la sédimentation. L'événement de collision sommet / sommet peut donc être généré dans le cas de l'érosion. La Figure 22 montre les motifs reconnaissant ces collisions et leur traitement. Le motif 22(a) est similaire au motif du glissement de sommet utilisé dans le processus de sédimentation (Figure 17(b)) pour la partie topologique mais diffère pour la partie plongement géométrique, qui correspond aux coordonnées temporelles de l'intersection de l'arête $n3$ avec l'arête $n1$. Le motif 22(b) représente la disparition de l'arête $n2$ par sa contraction en sommet. Un nouveau plongement géométrique est affecté à ce sommet, correspondant à l'intersection des lignes portant $n3$ et $n1 - n4$.

Il est possible de complexifier notre modèle d'erosion en supposant que les couches ont une « érodabilité » différente. Dans ce cas, il faut concevoir des traitements de collision plus riches, tenant compte des vitesses différentes d'érosion.

5.4. Modèle de création de failles

Une faille est une zone de rupture le long de laquelle se produit généralement une déformation cisaillante. Une faille peut débuter et s'arrêter n'importe où dans la scène. Elle peut donc diviser un bloc entièrement ou non. Les entités géologiques se trouvant de part et d'autre d'une faille peuvent alors glisser. La création de faille et le glissement sont dûs aux forces exercées par des contraintes tectoniques.

Dans [BPRS01,Sch02,BSP*04], les modèles géologiques 3*D* représentent les failles par des surfaces délimitant des volumes 3*D*. Par analogie, nous avons choisi d'utiliser une représentation par ligne polygonale (Figure 23) en 2*D*. Une ligne polygonale est donc subdivisée en un ensemble de segments identifiés (*seg*_0, *seg*_1 sur la figure 23). La croissance d'une ligne polygonale débute par la croissance de son premier segment. Une fois terminée, elle se poursuit par la croissance du second segment, ainsi de suite. Les segments croissants peuvent rentrer en collision avec les entités de la scène, ce qui se traduit par une décomposition en sous-segments. La désignation de ces sous-segments est réalisée par l'utilisation d'un préfixe donné par l'utilisateur et d'un identifiant numérique (*prefixe*_0 et *prefixe*_1 issus de *seg*_0, *prefixe*_2 issu de *seg*_1 sur la figure 23). Pour identifier la ligne polygonale, un parent commun à toutes les arêtes créées est inséré dans l'arbre de désignation.

La création de failles est définie par quatre paramètres : la forme de la ligne polygonale, une date de début *t_min*, une date de fin *t_max*, un préfixe pour les arêtes créées.

Ce phénomène peut être décrit dans le scénario d'évolutions géologiques à l'aide de la fonction suivante :

```
actionFaille(t_min, t_max − t_min,
  "prefix_interfaces",
  shape);
```

Le paramètre *t_max*−*t_min* correspond à la durée du phénomène. Le paramètre *prefix_interface* permet au système de savoir comment désigner les arêtes provenant du phénomène de création de failles.

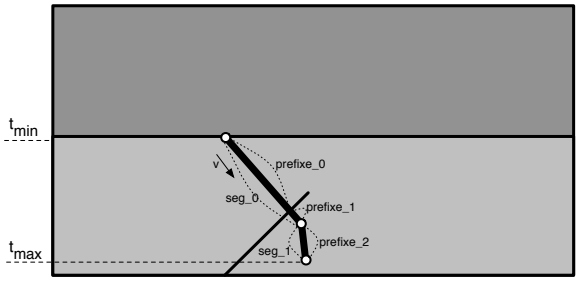

Figure 23: *Paramètres du modèle de création de failles.*

Le traitement de ce phénomène débute par la recherche de la zone où débute la faille. Cette recherche consiste à localiser l'entité la plus proche (au sens géométrique) du premier sommet de la ligne polygonale. Ensuite, deux cas peuvent se présenter pour traiter la création de la faille (Figure 24) : la croissance débute soit sur un bord (sommet ou arête), soit à l'intérieur d'une face. Dans le premier cas, si la faille débute sur une arête, un sommet est inséré sur l'arête concernée. Ensuite, ce sommet est identifié avec l'extrémité de la première arête de la faille. Enfin, cette arête grossit en fonction des paramètres définis dans action_faille (Figure 25). Dans le second cas, pour maintenir la contrainte de connexité de chaque 2-g-carte de notre modèle, il faut insérer une arête dite fictive (un booléen indique cette propriété) afin de représenter l'information d'inclusion de l'arête dans la face, et avoir ainsi une seule composante connexe par n-g-carte. Quel que soit le cas considéré, la croissance des arêtes est ensuite traitée de manière homogène.

Notons que les arêtes fictives peuvent être impliquées dans des évènements, au même titre que tout autre arête. Dans ce cas, nous traitons spécifiquement ces arêtes afin de les adapter aux modifications calculées afin qu'elles ne compliquent pas la carte inutilement (en provoquant une découpe de face en deux, par exemple). Il faut parfois les supprimer (contrainte de connexité à nouveau respectée) ou par exemple modifier leur point d'accroche lorsqu'elles sont traversées.

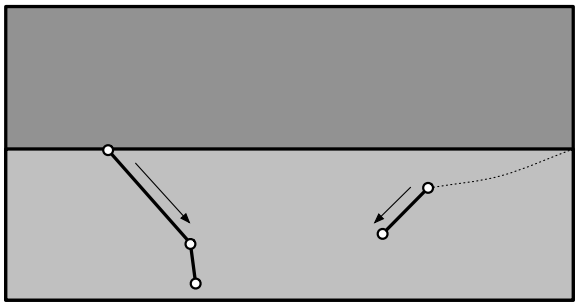

Figure 24: *Différents cas de positions de débuts de failles (en pointillé : arête fictive).*

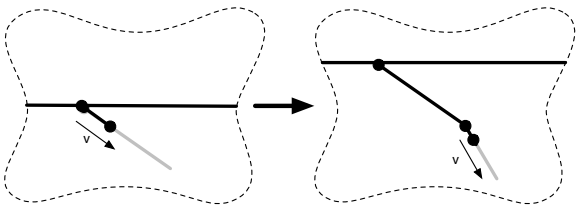

Figure 25: *Croissance d'un ensemble de segments.*

Si la ligne polygonale est constituée de plusieurs segments et que le segment en cours de croissance n'est pas le dernier constituant la faille, la date de fin de croissance du segment courant est ajoutée à la file d'événements. Cet événement

permet d'indiquer au système qu'il doit vérifier si le phénomène est toujours en cours. Dans un tel cas, le système reprend le traitement de création de segment décrit plus haut.

Comme pour les autres phénomènes géologiques, les événements de collisions sont calculés. Nous considérons qu'il existe deux types de collisions, soit « bloquante », soit « traversante ». Dans le premier cas (collision faille/faille par exemple), la croissance de la faille s'arrête au niveau de l'entité rencontrée (Figure 28(c)), ce qui se traduit par l'identification de l'extrémité croissante de la faille avec le bord du bloc courant, avec si besoin l'insertion d'un sommet sur ce bord (Figures 11(b) et 12(b)). Dans le second cas, la faille traverse l'interface rencontrée, ce qui se traduit par l'identification de l'extrémité croissante de la faille avec l'interface et l'insertion d'une nouvelle arête de l'autre côté de l'interface pour continuer le processus de croissance (Figure 26).

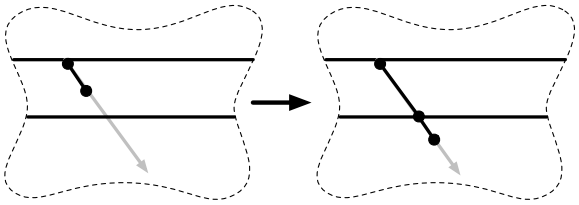

Figure 26: *Intersection d'une faille avec une interface.*

5.5. Modèle de glissement

Les failles étant causées par des tensions exercées sur les différentes couches, ces tensions entraînent le glissement de blocs le long de ces failles.

Dans notre modèle, le glissement géologique est défini par sept paramètres : une date de début *t_min*, une date de fin *t_max*, la ligne polygonale composée d'arêtes à faire glisser, un pas de subdivision minimum pour représenter la déformation introduite par le glissement, une direction relative par rapport à la première arête, une vitesse et une fonction de pondération de la vitesse *velocity_weighting_function* pour simuler une influence dépendant de la distance avec le point de glissement.

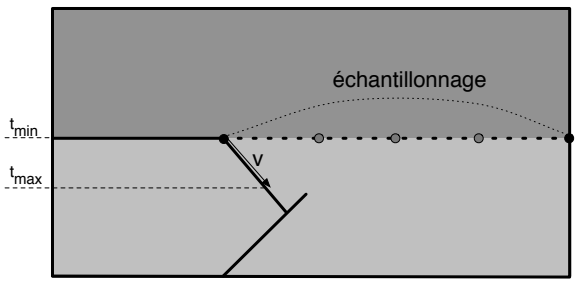

Figure 27: *Paramètres du modèle de glissement.*

Ce phénomène peut-être décrit dans le scénario d'évolutions géologiques à l'aide de la fonction suivante :

```
actionGlissement(t_min, t_max − t_min,
    interface_list, sampling,
    relative_direction,
    velocity,
    velocity_weighting_function);
```

Les paramètre *t_max*, *t_min*, *sampling* sont similaires à ceux vus précédemment. Le paramètre *interface_list* correspond à la zone qui doit glisser. Le paramètre *relative_direction* correspond à la direction relative à la première interface vers laquelle le système va effectuer le glissement.

Le traitement de ce phénomène débute par le suréchantillonnage des arêtes composant la ligne polygonale qui va glisser. L'extrémité de la première arête est "éclatée" suivant une direction relative donnée par rapport à sa désignation ($left$, $right$), ce qui se traduit par l'insertion d'un sommet sur l'arête où s'effectue le glissement et l'identification de l'extrémité de la première arête sur ce sommet (Figure 28(d)). L'arête sur laquelle la ligne polygonale glisse donne la direction du mouvement du sommet courant. Cette direction et le paramètre vitesse permettent de déduire un vecteur vitesse et donc de calculer le mouvement des sommets de la ligne polygonale. Ces sommets sont plongés en fonction de leur position courante et du vecteur vitesse pondéré par la fonction fournie par l'utilisateur (cette fonction peut être quelconque). Dans notre exemple, nous utilisons une gaussienne pour obtenir une forme de glissement arrondie.

Le phénomène de glissement peut aboutir à tous les types de collisions : sommet / sommet non liés par une arête, sommet / sommet liés par une arête, et sommet / arête. Dans le cas d'une collision sommet / sommet non liés ou sommet / arête, la priorité est donnée au sommet / arête issue du glissement, ce qui se traduit par un comportement similaire à l'érosion et reprend donc ses mécanismes. En cas d'ambiguïté, c'est-à-dire quand les entités sont de même priorité et de même nature, le système affiche un message d'alerte à l'utilisateur. Dans le cas d'une collision sommet / sommet liés par une arête, i.e. pour une arête de longueur nulle, l'arête est contractée. Si cette arête correspondait à la zone de glissement, le nouveau sommet suivant est éclaté en suivant le schéma de la figure 2.

L'utilisation d'une date de début et de fin pour les phénomènes de création de faille et de glissement a un but de démonstration et de validation du système. En effet, à l'échelle géologique, ces phénomènes peuvent être considérés comme ponctuels.

5.6. Prototype 2*D*

Afin de tester notre modèle, nous avons créé un prototype appliqué à la géologie. Pour représenter les 2-g-cartes, nous utilisons le noyau de l'application Moka (http://www.sic.sp2mi.univ-poitiers.fr/moka/). La représentation des

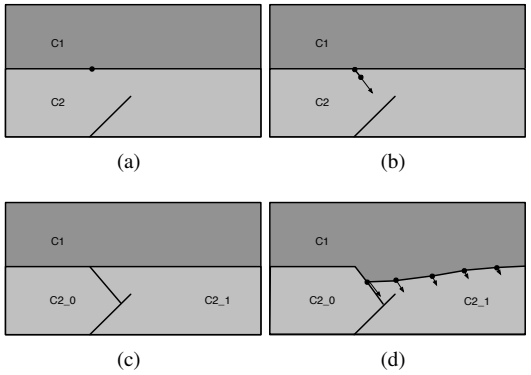

Figure 28: *Étapes d'une création de faille suivie d'un glissement. (a) La couche C2 contient une faille définie antérieurement ; la couche C1 représente l'atmosphère. Début d'une nouvelle faille par l'insertion d'un sommet sur l'interface des couches C1 et C2. (b) Création et croissance du premier segment de la faille. (c) Collision avec la faille initiale : la croissance s'arrête sur cette faille et C2 est divisée en C2_0 et C2_1. (d) Rééchantillonnage de l'interface entre C1 et $C2_1$ et déformation de cette interface suivant la direction du glissement.*

équations du 0-plongement est assurée par la bibliothèque GiNaC (http://www.ginac.de/). GiNaC est une bibliothèque permettant de réaliser du calcul symbolique et des résolutions d'équations linéaires. Durant les calculs de détection de collisions, il arrive que les systèmes d'équations ne soient pas linéaires, pour cela nous utilisons un programme externe nommé Maxima (http://maxima.sourceforge.net/). Maxima est un programme de calcul formel écrit en Lisp. L'utilisation de Maxima en tant que programme externe ralentit considérablement les calculs de collisions (mise en forme des calculs, lancement de Maxima : exécution d'un interpréter Lisp et chargement des bibliothèques et de l'interpréteur de Maxima, interprétation du résultat). Mais son choix s'est imposé par sa gratuité et sa disponibilité. Le scénario et le langage de script bas niveau sont représentés sous forme de bibliothèques pour le langage interprété Lua (http://www.lua.org/). L'utilisation de Lua évite de compiler un programme à chaque changement de script.

Le choix des bibliothèques et programmes externes a permis de réaliser un prototype fonctionnel mettant en exergue les outils nécessaires à l'implantation d'un tel modèle, ainsi que les zones de l'application pouvant être améliorées (résolution d'équation pour les détections de collisions et le calcul de nouveaux plongements).

5.7. Résultats

Cette section est consacrée à la mise en pratique de notre modèle, allant du langage de scénario composé de fonctions géologiques (figure 9(a-b)) à l'animation graphique des entités, en passant par la génération du script bas niveau (figure 9(c)) composé des opérations topologiques (figure 9(d)).

Le scénario du Listing 1 débute par la description de la construction d'un chenal via une succession de sédimentations et d'érosions. Il se termine par des créations simultanées de failles dont l'une conduit à un glissement.

Listing 1: *Ce scénario décrit 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) scene = CSceneVertex:new(gmv, 8, 6);
(2) scene:actionChenalCreation(0, "C1", "profil_7.v");
(3) scene:actionSedimentation(10, 10,
    3, "Sed01",
    "border_left_0", left,
    forward,
    "border_right_1", left);
(4) scene:actionErosion(30, 10,
    "Sed01_0", begin,
    forward,
    "Sed01_0", begin,
    3,
    "Ero01", 2);
(5) scene:actionSedimentation(40, 10,
   .4, "Sed02",
   "border_left_0_0", left,
   forward,
   "border_right_1_1", left);
(6) scene:actionFaille(60, 20,
   "faille06",
   {{1.843325, 1.92517}, {3.6, 2.5}},
   0);
(7) scene:actionFaille(60, 20,
   "faille01",
   {{1,2}, {1.5,.3}},
   0);
(8) scene:actionFaille(60, 20,
   "faille05",
   {{2.29413, 1.29558}, {2, 3}},
   0);
(9) scene:actionFaille(60, 20,
   "faille04",
   {{2.55,1.55}, {2.5,3.5}, {1, 5}},
   0);
(10) scene:actionFaille(60, 30,
   "faille03",
   {{2.55,1.54}, {2.5,.1}},
   0);
(11) scene:actionFaille(60, 10,
   "faille02",
   {{.5,2}, {1.0,.5}},
   0);
(12) scene:actionFaille(80, 20,
   "faille07",
   {{1,2},{7,3.5}},
   0);
(13) scene:actionGlissement(110, 20,
   {"Sed02_0_1"}, 1,
   right, .1);
```

Ce scénario génère une animation illustrée par les images de la figure 29, extraites de notre lecteur d'animation. Ce logiciel est utilisé pour visualiser l'animation dans son intégralité et examiner la structure topologique à chaque instant. La partie gauche de la fenêtre présente les entités regroupées dans une arborescence permettant de connaître leur origine et leurs descendants. Cette arborescence est mise à jour à chaque changement topologique. De plus, elle permet de sélectionner les entités par leur nom et de les surligner dans

la fenêtre d'animation. La fenêtre d'animation permet de sélectionner des faces, des arêtes, des sommets et de leur donner des noms. Elle peut aussi afficher une vue éclatée de la structure topologique (Figure 30). La sélection, l'affichage et la navigation temporelle sont des outils d'analyse précieux pour comprendre l'évolution des entités.

Comme nous venons de le voir, tout scénario génère une suite d'événements. Ces événements sont analysés et traduits en un script d'évolutions de structures topologiques. Les paragraphes suivants détaillent le début du script obtenu à partir du scénario du Listing 1. Ils se basent sur les figures 29 et 30 qui mettent en évidence les changements causés par le script haut niveau sur la structure, la géométrie et l'arbre de désignation. Ensuite, ils décrivent brièvement la suite du script dont les étapes se trouvent sur la figure 30.

(1-2) Ce scénario crée une nouvelle animation pour laquelle chaque image est de dimension 8*x*6, définit la première image-clef à la date 0 et positionne le « profil de » chenal $C1$ composé de dix arêtes désignées $C1\_0$ à $C1\_9$ (Figure 29(a)).

(3) Période de sédimentation entre les dates 10 et 20 avec une hauteur de 3. Cette sédimentation affecte toute la scène (du bord gauche au bord droit) et crée des interfaces disjointes dont la désignation est préfixée par *Sed*01. Ce phénomène a pour effet de créer deux nouvelles arêtes *Sed*01_0 et *Sed*01_1 dont les extrémités glissent le long du bord de $C1$ (Figure 29(b)). À l'étape de la figure 29(b), quatre sommets ont été insérés sur les quatre arêtes $C1\_1$, $C1\_2$, $C1\_7$ et $C1\_8$. Ces insertions aboutissent à la séparation de chaque arête en deux morceaux, (par exemple l'arête $C1\_8$ est divisée en $C1\_8\_0$ et $C1\_8\_1$), ce qui se traduit par un sous-arbre dans l'arbre de désignation. Au niveau du pic central, *Sed*01_0 et *Sed*01_1 se rencontrent et fusionnent en *Sed*01_0 (Figure 29(c)). Le nom donné à l'arête résultante est dépendant de l'ordre de traitement des événements.

(4) Période d'érosion de la date 30 à 40 pour une hauteur de 3 appliquée à l'interface *Sed*01_0. Ce phénomène a pour effet d'échantillonner la surface d'érosion *Sed*01_0 (en *Sed*01_0_0, *Sed*01_0_1, *Sed*01_0_2 et *Sed*01_0_3) en affectant une trajectoire de déformation aux nouveaux sommets (Figure 29(d)). Le moteur de collision prédit une rencontre entre la surface d'érosion et le pic de $C1$. La collision est traitée en donnant la priorité à la surface d'érosion (Figure 29(e)). Ce traitement insère un sommet sur les arêtes *Sed*01_0_1 et *Sed*01_0_2 ce qui les scinde respectivement en *Sed*01_0_1_0 et *Sed*01_0_1_1, et en *Sed*01_0_2_0 et *Sed*01_0_2_1. L'érosion se poursuit et supprime les arêtes $C1\_4$ et $C1\_5$ (Figures 29(e) et 29(f)).

(5-12) Période de sédimentation (préfixe *Sed*02, aboutissant à l'arête *Sed*02_0) de la date 40 à 50 (figure 30(h)) suivie d'une série de créations de failles ($faille01$ à $faille07$) (figure 30(i)) qui se traduisent par une série de créations d'arêtes et de collisions (figure 30(j)). À l'issue de ce processus, l'arête *Sed*02_0 a été découpée en *Sed*02_0_0 (partie gauche) et *Sed*02_0_1 (partie droite) par une faille. À la date 110, l'arête *Sed*02_0_1 glisse vers sa droite (figures 30(k-l).

Le script du Listing 2 est un extrait du script bas-niveau généré par l'application et il correspond au début de la première sédimentation décrite dans le scénario du Listing 1, figure 29(b). Ligne 2, le script crée une nouvelle 2-g-carte, copie de la précédente, à la date 10. Il déclare ensuite deux nouvelles interfaces, *Sed*01_0 et *Sed*01_1 (Lig.3, 6), et les identifie aux sommets $C1\_1/end$ (rappel : un sommet est désigné par un nom d'arête et sa position par rapport à elle), $C1\_2/begin$ et $C1\_7/end$, $C1\_8/begin$ (Lig.4,5 et Lig.7,8). $C1\_1/end$ et $C1\_2/begin$ (respectivement $C1\_7/end$ et $C1\_8/begin$) désignent un même sommet. Les deux nouvelles faces issues de ces identifications sont donc constituées d'une arête seulement et sont donc dégénérées. Quatre événements de glissements de sommets sont générés, ce qui se traduit par quatre insertions de sommets (Lig.9,12,15,18) sur les arêtes $C1\_1$, $C1\_2$, $C1\_3$ et $C1\_4$, quatre déconnexions (Lig.10,13,16,19) et quatre identifications des extrémités des arêtes *Sed*01_0 et *Sed*01_1 aux nouveaux sommets (Lig.11, 14, 17, 20).

Listing 2: *Cet extrait de script bas niveau représente les opérations topologiques du début de la première sédimentation du Listing 1.*

```
1  [...]
   dp1 = animation:copyLastDiscretePlane2d(10)
   dp1:createInterface("Sed01_0", 0)
   dp1:identification("C1_1", end, "Sed01_0", begin, left)
5  dp1:identification("C1_2", begin, "Sed01_0", end, left)
   dp1:createInterface("Sed01_1", 0)
   dp1:identification("C1_7", end, "Sed01_1", begin, left)
   dp1:identification("C1_8", begin, "Sed01_1", end, left)
   dp1:splitInterface({50}, "C1_1")
10 dp1:desidentification("Sed01_0", begin)
   dp1:identification("C1_1_1", begin, "Sed01_0", begin, left)
   dp1:splitInterface({50}, "C1_2")
   dp1:desidentification("Sed01_0", end)
   dp1:identification("C1_2_0", end, "Sed01_0", end, left)
15 dp1:splitInterface({50}, "C1_7")
   dp1:desidentification("Sed01_1", begin)
   dp1:identification("C1_7_1", begin, "Sed01_1", begin, left)
   dp1:splitInterface({50}, "C1_8")
   dp1:desidentification("Sed01_1", end)
20 dp1:identification("C1_8_0", end, "Sed01_1", end, left)
   [...]
```

Le résultat de l'animation du scénario du Listing 1 est composé de 2908 brins répartis sur 24 images clefs. Actuellement, nous utilisons Maxima pour résoudre sans optimisation les équations non linéaires provenant du moteur de collisions. Le temps total de calcul est donc conséquent, ici de 14 mins 26s (sur un portable de type MacBook avec un processeur à 2.16 GHz), (25s si les solutions des équations non linéaires sont mises en cache, ce qui dénote que le prototype peut être amélioré en ne passant plus par un programme externe) mais pourrait être amélioré en utilisant des bibliothèques de calculs et des techniques de détections de collisions optimisées.

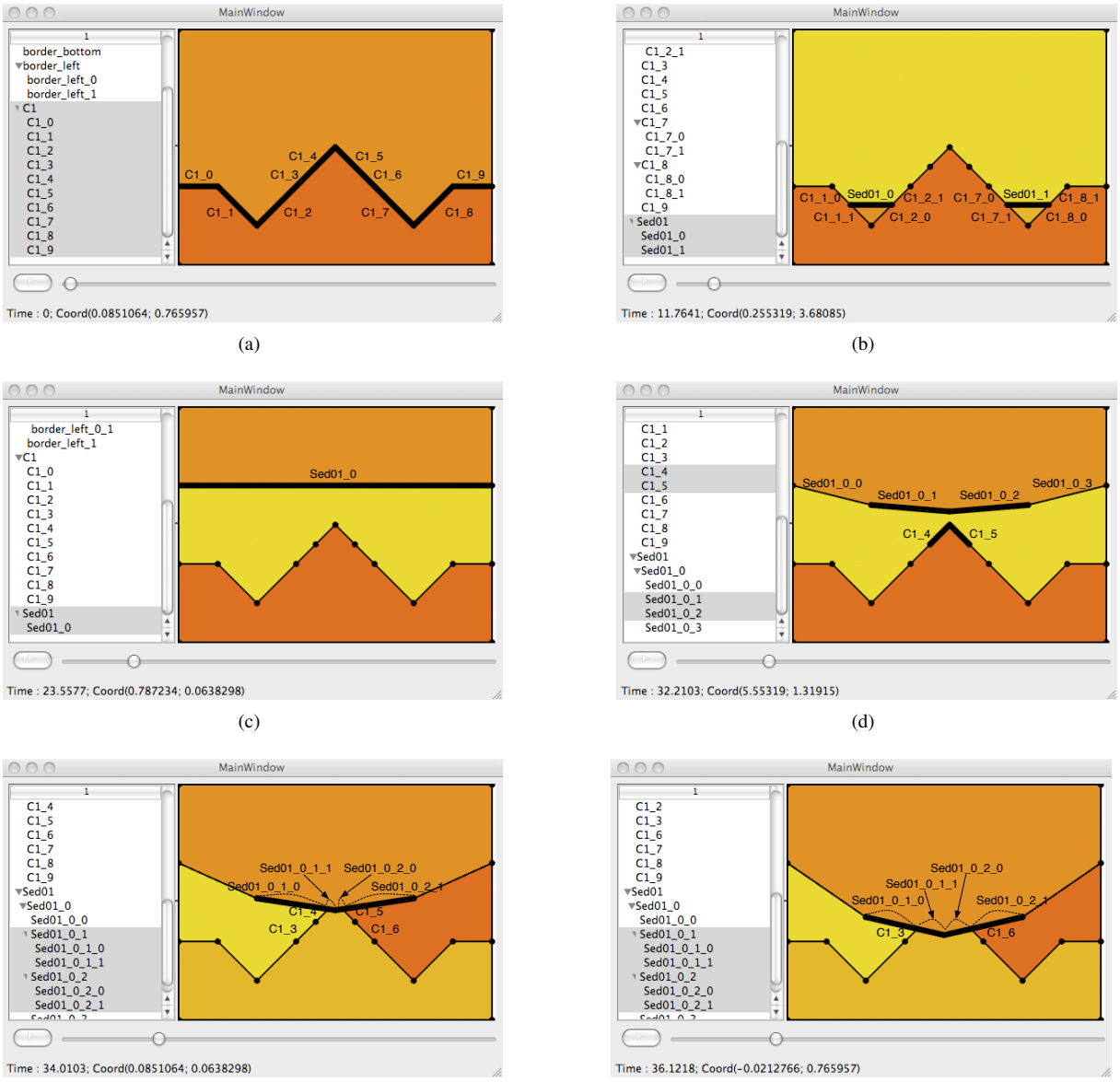

Figure 29: *Images de l'animation générée à partir du Listing 1 illustrant l'évolution des noms des entités : (a) Création du sous-sol à partir d'un profil (2-g-carte 1). (b) Début de sédimentation créant deux blocs disjoints (2-g-carte 2). (c) Fusion des deux blocs (2-g-carte 5). (d) Début de l'érosion et détection de la prochaine collision causée par le phénomène (2-g-carte 6). (e) Résultat après le traitement de la collision entre la surface d'érosion et le pic (2-g-carte 7). (f) L'érosion continue et supprime les arêtes C*1_4 *et C*1_5 *(2-g-carte 8).*

6. Extension à la 3D

L'instanciation de notre moteur en 2*D* n'est qu'une étape vers une solution gérant complètement la 3*D*. Nous avons présenté nos modèles structurel, événementiel et sémantique en dimension quelconque, l'architecture de la solution reste donc strictement identique lors du passage aux dimensions supérieures à 2. Cependant, l'ajout d'une dimension supplémentaire à la 2*D* implique tout d'abord un nombre accru de type de collisions : aux collisions sommet/arête et sommet/sommet s'ajoutent les collisions sommet/face et arête/face. Pour chacune de ses collisions, il s'agit de définir les transformations à appliquer. Les plongements temporels sont également à adapter afin par exemple qu'un sommet puisse suivre une trajectoire (pour simplifier, rectiligne) sur une face (comme en 2*D*, nous avons fait déplacer des points le long d'une arête). Enfin, c'est l'ensemble des transformations nécessaires qu'il nous faut définir, car il y a plus de transformations à considérer et leur exhaustivité nécessite une étude plus aboutie.

7. Conclusion et perspectives

Nous proposons un modèle pour créer et animer des partitions *nD* de l'espace. Nous appliquons ce modèle au cas particulier 2*D* et nous en donnons une application permettant de représenter l'évolution du sous-sol. Cette évolution est fournie à partir de la description de phénomènes sous la forme d'un scénario. Ce scénario repose principalement sur la modélisation de quatre phénomènes : géologiques, la sédimentation, l'érosion, la création de failles et le glissement, qui conduisent à la création, fusion, disparition et scission de blocs géologiques. Un scénario est analysé et traduit en événements constituant les événements initiaux de la scène. Le système construit l'animation à partir du mouvement des entités et modifie la structure et la géométrie en cas de collision. Une collision est traitée suivant le phénomène considéré et la nature des entités en jeu. L'animation et les modifications de la structure sont portées par un script d'opérations topologiques, un arbre de désignation et une liste de n-g-cartes dont les sommets sont géométriquement plongés par une fonction temporelle. L'animation du scénario peut donc être analysée pour en comprendre le résultat et sa construction.

Les contributions principales de cet article sont de fournir un modèle général d'animation de partition *nD*, d'utiliser la description explicite des liaisons de voisinages du modèle topologique pour réduire la complexité de la détection des auto-collisions en limitant le nombre de couples d'entités à tester. Ce modèle a été mis en application dans le contexte de la géologie, pour représenter l'évolution du sous-sol. D'autres domaines d'application sont envisagés : la biologie (pour la représentation de la genèse d'un organe par exemple) ou le suivi d'évolution d'objets maillés en mécanique (transformations issues de calculs éléments finis de structures homogènes par zones par exemple). Ces domaines n'ont pas pour le moment fait l'objet d'études de notre part. En effet, l'approche par scénario est trop limitative dans ces contextes. Il nous faut, au préalable, coupler notre système avec un moteur de simulation capable de provoquer des événements et piloter les évolutions de plongement géométrique des sommets.

Sur le plan des perspectives, nous avons présenté un certain nombre de problèmes liés au passage à la 3D. Ce passage est l'objet de nos études actuelles. D'autres améliorations sont cependant envisagées. Dans notre système, le traitement des divers cas de collision doit se faire en développant de nouveaux algorithmes, à la fois pour détecter et identifier les cas rencontrés puis pour appliquer les transformations (identification des brins impliqués dans la ou les opérations à invoquer). Nous souhaitons formaliser le traitement des événements en nous orientant vers un système de reconnaissance de motifs et de règles de réécritures de graphes [PCLG*07]. Cette approche devrait nous permettre d'automatiser le processus de traitement des transformations topologiques (recherche de motifs dans un jeu de règles et génération automatique du code des transformations). Par ce biais, nous renforçons l'indépendance de notre modèle d'animation par rapport à des applications particulières (et simplifions le portage pour une application donnée).

Ensuite, pour représenter fidèlement les phénomènes naturels, nous pensons améliorer le modèle structurel et permettre la manipulation des structures de données complexes et en recourant à des modèles multi-résolution. L'animation contrôlée de telles structures est un domaine qu'il nous paraît indispensable d'explorer. Enfin, l'édition interactive des scénarii pour modifier certains paramètres et rejouer l'évolution du modèle nous intéresse particulièrement. Nous comptons utiliser les méthodes de « nomination » [BAMSB05] pour résoudre ce problème.

Toutes ces améliorations ont comme objectif de fournir un modèle d'animation 3D aussi indépendant que possible du domaine d'application utilisé. Plus précisément, il s'agit de peaufiner les outils permettant de traduire les opérations haut-niveau (propres à un domaine d'application donné et/ou à une représentation spécifique des phénomènes simulés) en opérations bas-niveau exécutées quelle que soit l'application.

Références

[BAMSB05] BABA-ALI M., MARCHEIX D., SKAPIN X., BERTRAND Y. : Intégration des opérations de nomination dans un modèle géométrique 3d. In Journées de l'Association Française d'Informatique Graphique (November 2005).

[BNDP05] BAC A., NAM V. T., DANIEL M., PERRIN M. : Traitement de surfaces géologiques pour la construction de modèles 3d. In GTMG 2005 (2005).

[BPRS01] BRANDEL S., PERRIN M., RAINAUD J.-F., SCHNEIDER S. : Geological interpretation makes earth models easier to build. In EAGE 63rd Conference, Extended Abstacts (June 2001).

[BSP*04] BRANDEL S., SCHNEIDER S., PERRIN M., GUIARD N., RAINAUD J.-F., LIENHARDT P., BERTRAND Y. : Automatic building of structured geological models. JCISEV (2004).

[BW71] BURTNYK N., WEIN M. : Computer generated key frame animation. Journal of the Society of Motion Picture and Television Engineers (1971), 149–153.

[DZ93] DWORKIN P., ZELTER D. : A new model for efficient dynamic simulation. In EUROGRAPHICS Workshop on Computer Animation and Simulation (EGCAS) (Barcelone, septembre 1993), pp. 135–147.

[FHHR98] FLOATER M., HALLBWACHS Y., HJELLE O., REIMERS M. : A cad-based approach to geological modelling. In Modeling 98 Conference (Nancy, France, Juin 1998).

[GM01] GIAVITTO J.-L., MICHEL O. : Mgs : a rule-based programming language for complex objects and collections. Electr. Notes Theor. Comput. Sci.. Vol. 59, Num. 4 (2001).

[Goc] Site web gocad. `http ://www.gocad.com/`.

[GTM*05] GUIMBERTEAU G., TERRAZ O., MÉRILLOU S., GHAZANFARPOUR D., PLEMENOS D. : Internal wood growth simulation based on subdivised 3D objects. Tech. rep., Laboratoire MSI, 2005.

[Gui06] GUIARD N. : Construction de modèles géologiques 3D par co-raffinement de surfaces. PhD thesis, mai 2006.

[Lie94] LIENHARDT P. : n-dimensional generalised combinatorial maps and cellular quasimanifolds. International Journal of Computational Geometry and Applications (1994).

[Lin68] LINDENMAYER A. : Mathematical models for cellular interactions in development. Journal of Theoretical Biology. Vol. 18 (1968), 280–315.

[LR79] LINDENMAYER A., ROZENBERG G. : Parallel generation of maps : Developmental systems for cell layers. In Proceedings of the International Workshop on Graph-Grammars and Their Application to Computer Science and Biology (London, UK, 1979), Springer-Verlag, pp. 301–316.

[LSM06] LÉON P.-F., SKAPIN X., MESEURE P. : Topologically-based animation for describing geological evolution. In International Conference on Computer Vision and Graphics (September 2006).

[LSM08] LÉON P.-F., SKAPIN X., MESEURE P. : A topology-based animation model for the description of 2d models with a dynamic structure. In Virtual Reality Interactions and Physical Simulations VRIPHYS (Novembre 2008), EG.

[Mal02] MALLET J.-L. : Geomodeling. Oxford University Press (2002).

[MKF03] MESEURE P., KHEDDAR A., FAURE F. : Détection des collisions. Rapport de l'Action Spécifique CNRS Num. 90, RTP 7 (december 2003).

[PCLG*07] POUDRET M., COMET J.-P., LE GALL P., ARNOULD A., MESEURE P. : Topology-based geometric modelling for biological cellular processes. In 1st International conference on Langage and Automata Theory and Applications (LATA 2007) (Tarragona, Spain, Mars 2007).

[Per98] PERRIN M. : Geological consistency : an opportunity for safe surface assembly and quick model exploration. 3D Modeling of Natural Objects, A Challenge for the 2000's. Vol. 3 (june 1998), 4–5.

[PL90] PRUSINKIEWICZ P., LINDENMAYER A. : The Algorithmic Beauty of Plants (The Virtual Laboratory). Springer, October 1990.

[Pro97] PROVOT X. : Collision and self-collision handling in cloth model dedicated to design garments. Graphics Interface (1997), 177–189.

[RML] Site web rml. `http ://www.beicip.com/`.

[Sch02] SCHNEIDER S. : Pilotage automatique de la construction de modèles géologiques surfaciques. PhD thesis, École des Mines de Saint-Étienne, 2002.

[Smi06] SMITH C. : On Vertex-Vertex Systems and Their Use in Geometric and Biological Modelling. PhD thesis, University of Calgary, April 2006.

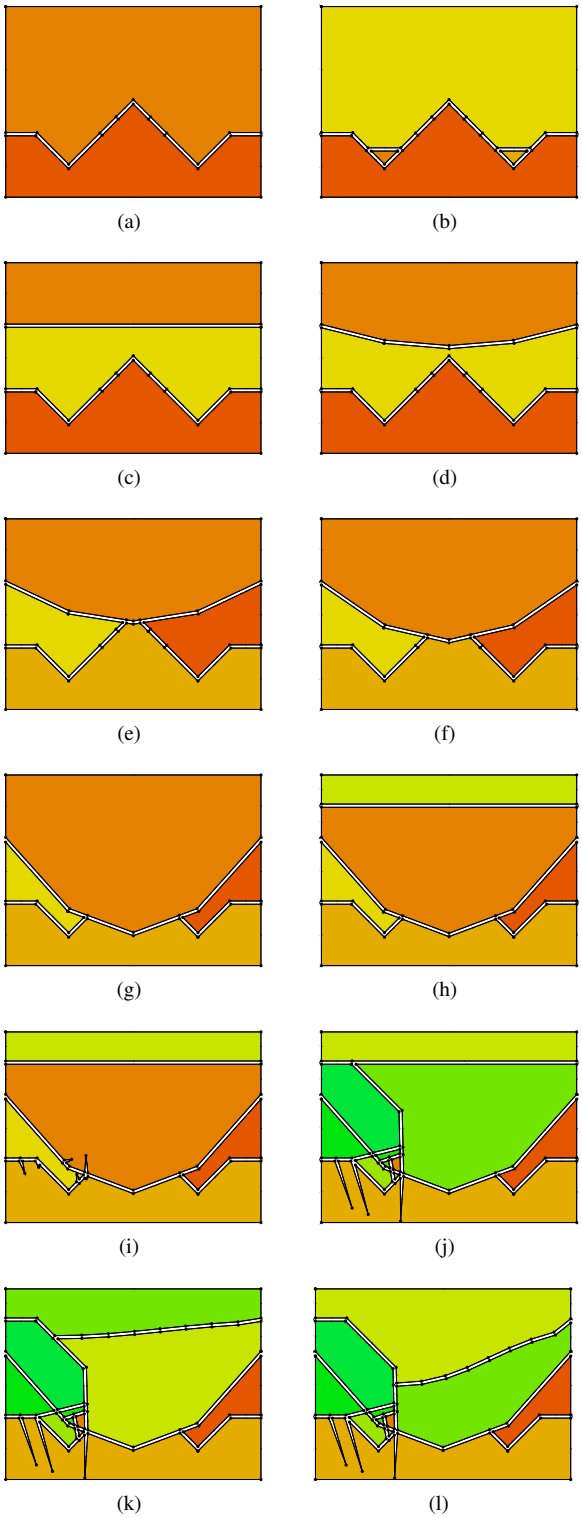

Figure 30: *Images générées à partir du Listing 1 - Vue topologique.*