Génération de contenu graphique

Nicolas Marechal

N° **127-2010** **Année 2010**

UNIVERSITÉ CLAUDE BERNARD – LYON 1

UFR INFORMATIQUE

École doctorale "Informatique et Information pour la Société" (EDIIS) de Lyon

THÈSE DE L'UNIVERSITÉ DE LYON

présentée et soutenue publiquement le 7 juillet 2010

pour l'obtention

du DIPLÔME DE DOCTORAT

(Arrêté du 7 août 2006)

Spécialité Informatique

par

Nicolas Maréchal

Génération de contenu graphique

Composition du jury

Rapporteurs :

M. Marc Neveu	Professeur, Université de Bourgogne
M. Jean-Michel Dischler	Professeur, Université Louis Pasteur de Strasbourg

Examinateurs :

M^{me} Marie-Paule Cani	Professeur, Institut polytechnique de Grenoble
M. Samir Akkouche	Professeur, Université Claude Bernard Lyon 1

Directeurs :

M. Éric Galin	Professeur, Université Lumière Lyon 2
M. Éric Guérin	Maître de conférences, Université Claude Bernard Lyon 1

Laboratoire d'InfoRmatique de l'Image et Systèmes d'information (LIRIS) – UMR CNRS 5205

Remerciements

Je tiens tout d'abord à remercier Éric Galin et Éric Guérin, mes encadrants, qui m'ont permis d'effectuer cette thèse. Je remercie également Marc Neveu et Jean-Michel Dischler d'avoir accepté d'être rapporteurs. Je remercie Marie-Paule Cani et Samir Akkouche qui ont accepté d'être examinateurs.

Merci à tous les membres du LIRIS pour la bonne ambiance, les discussions, les idées, les pauses... bref tout ce qui rend le quotidien très agréable dans ce laboratoire. Merci à mes co-bureaux de la TD6 qui m'ont supporté durant ces trois ans.

Merci Clem, Dodo pour les voyages dépaysant organisés comme nous seuls en avons le secret ;-).

Un grand merci à tous mes amis pour toutes les fêtes, les sorties, la montagne... qui nous rappellent qu'avoir une vie sociale en dehors de la thèse c'est important ! Merci Nico P, Nico Chanon, Diane, Alex, Pierre(s), Virginie, Timothée, Romain alias Metalman, Marie-Laure, Charlène et tous les autres que j'oublie...

Je tiens particulièrement à remercier toute ma famille : mes parents, mon frère et mes sœurs, ma grand-mère... qui m'ont toujours soutenu.

À mes parents.

Table des matières

Introduction

Depuis de nombreuses années, la modélisation de paysages naturels et urbains ainsi que la simulation de phénomènes naturels ont fait l'objet de nombreux travaux en informatique graphique. Le principal objectif est de créer les environnements les plus réalistes possibles en reproduisant le plus fidèlement ce que nous avons l'habitude de voir. L'industrie des loisirs numériques profite de l'augmentation de la puissance de calcul et des capacités mémoire pour produire des paysages de synthèse et simuler des phénomènes naturels de plus en plus réalistes. Ce réalisme est obtenu en introduisant de nombreux détails et beaucoup de variétés. Cependant, la création des détails et de la variété est souvent réalisée manuellement. Cette tache est fastidieuse et le temps nécessaire pour modéliser les objets est important. Dans cette thèse, nous nous intéressons au problème de génération de variété permettant d'accroître le réalisme des paysages.

Une part importante du réalisme d'une scène est obtenue grâce à la présence d'une très grande diversité ainsi que d'un très grand nombre de détails. Les variations de forme et de texture contribuent fortement à l'apparence des paysages et des objets. Dans la nature, ces changements d'apparence sont le résultat de processus et de phénomènes complexes tel que l'érosion du terrain par l'eau ou encore l'apparition de mousses, de rouille ou de salissures provoquée par la pollution. L'évolution au cours du temps et la manière dont sont créés les objets produisent l'ensemble de ces variations.

Par exemple, lors du processus de développement d'un arbre, de nombreux facteurs environnementaux et biologiques interviennent affectant sa géométrie et sa texture. Ceci donne lieu à une très grande variété d'instances d'arbres malgré le fait qu'ils appartiennent à la même espèce. De plus, au fil des années, l'écorce des arbres est colonisée par des lichens, de la mousse et du lierre. Sur une échelle de temps plus courte, les saisons ont également une influence considérable sur l'aspect des arbres. En automne, les feuilles changent de couleur et, au début de l'hiver, elles finissent par tomber. Au printemps, des bourgeons et des fleurs apparaissent pour finalement être remplacés par des feuilles et des fruits en se rapprochant de l'été.

Un très grand nombre de méthodes de générations d'objets et de techniques de vieillissement ont été proposées (Chapitre 1). Ces approches peuvent être classées en trois catégories : la génération à partir d'esquisses et d'exemples, la génération procé-

durale et la simulation.

Les arbres, les plantes, les villes et les bâtiments sont généralement créés soit par édition à l'aide d'esquisses et d'exemples, soit procéduralement en utilisant des grammaires de formes et des L-Systèmes. Cependant ces approches sont spécifiques à un type d'objets précis. Le nombre de polygones générés est souvent beaucoup trop important pour que les objets puissent être visualisés en temps réel. D'autre part, la création d'un grand nombre de variétés est très laborieuse et lorsque le nombre d'objets générés devient important, la capacité mémoire est rapidement atteinte. Pour les approches procédurales, les paramètres permettant de contrôler la génération des objets sont très nombreux et peu intuitifs pour un artiste. Une grande maîtrise de l'outil et de la méthode associée est nécessaire pour générer une forme particulière.

Les techniques de vieillissement font généralement appel à la simulation et permettent de modéliser de façon physiquement réaliste les phénomènes sous-jacents altérant les objets. Cependant, il est généralement impossible de contrôler le résultat final car celui-ci dépend de lois physiques complexes. De plus, l'utilisation de discrétisation est très coûteuse en mémoire et le coût de la résolution des équations en temps est important. Finalement, bien que l'impact du climat soit fondamental sur les changements d'apparence des écosystèmes ou sur le vieillissement des objets, celui-ci est tout simplement ignoré.

Dans cette thèse, nous avons étudié les trois types de méthodes pour la génération de contenu graphique. Notre objectif est double : premièrement, proposer des solutions aux problèmes énoncés précédemment et deuxièmement, couvrir l'ensemble du spectre pour avoir une bonne compréhension des avantages et inconvénients des différentes approches.

Nous avons proposé une méthode de génération à partir d'exemples permettant de générer rapidement un grand nombre de variétés d'un même objet (Chapitre 2). Notre approche consiste à découper les objets en morceaux appelés *fragments* dont la géométrie et la texture sont modifiées. Ces modifications engendrent la création de nouveaux fragments qui sont stockés dans des atlas. Les fragments sont ensuite ré-assemblés générant ainsi un grand nombre de variétés. Le contenu graphique que nous obtenons est figé dans l'espace, il n'évolue pas dans le temps et ne peut pas s'adapter automatiquement à des contraintes extérieures. Nous appellerons donc la variété générée *variété de contenu statique*.

Dans un contexte de génération d'objets manufacturés et plus particulièrement celui de la création d'ouvrages d'art, les variétés d'objets statiques ne peuvent pas s'adapter à des contraintes tel que le relief d'un paysage ou la largeur d'une rivière. Par conséquent, nous avons proposé une méthode de génération procédurale en utilisant des modèles paramétrés (Chapitre 2). Les modèles paramétrés sont décrits à l'aide de procédures algorithmiques et de fonctions de calcul de paramètres permettant de générer et de positionner les différents éléments architecturaux. La mise en place des modèles s'effectue à l'aide de deux types d'objets : les objets dit *terminaux* et les objets dit *récursifs*. Nous avons appliqué cette méthode à la génération de routes, de ponts

et de tunnels. Ces ouvrages s'adaptent automatiquement à la trajectoire de la route et permettent de franchir des rivières de différentes largeurs en utilisant le même modèle. Le gain de temps est donc considérable. En effet, dès lors que la trajectoire ou que la largeur d'une rivière est modifiée, il n'est pas nécessaire de re-modéliser entièrement les ouvrages. De plus, la modification des paramètres permet de générer rapidement des ouvrages de formes différentes mais similaires. Nous appellerons ce type de variété *variété de contenu adaptatif*.

Pour finir, nous avons proposé une méthode de simulation permettant de générer des paysages hivernaux évoluant en fonction des conditions climatiques (Chapitre 3). Notre approche consiste à simuler les différents échanges thermiques présents dans la nature et ainsi suivre l'évolution du manteau neigeux et de la couche de glace se formant en surface de lacs. Notre méthode nous permet de simuler la fonte de la neige, le gel et dégel de l'eau en simulant les changements d'état. L'inertie thermique dans les masses d'eau de volume différent est prise en compte en simulant la convection naturelle. Les conditions climatiques sont définies à l'aide de fonctions décrivant le climat et permettant de mettre en place des scénarios de simulation. D'autre part, le contrôle de notre simulation s'effectue à l'aide de ce scénario climatique. Le paysage généré par ce type d'approche évolue au cours du temps. Les changements d'apparence produisent des variations du contenu graphique. Par conséquent, la variété obtenue sera appelée *variété de contenu dynamique*.

Dans la suite du document, nous détaillons les méthodes existantes ainsi que les différentes approches que nous proposons.

Chapitre 1

État de l'art



Sommaire

1.1 Introduction

La génération de contenu graphique pour la modélisation de paysages est un domaine vaste regroupant de nombreux travaux.

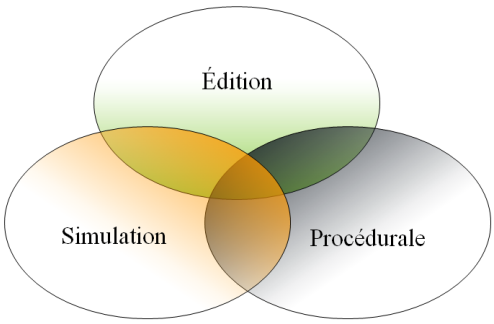


FIGURE 1.1 – Catégories de méthodes permettant de générer du contenu graphique.

Les méthodes existantes peuvent être classées en trois catégories pouvant se recouper (Figure 1.1) : la génération et l'édition à partir d'esquisses et d'exemples, la génération procédurale, la simulation. Selon l'approche utilisée, les méthodes proposées possèdent un certain nombre d'avantages et d'inconvénients.

Les méthodes de génération et d'édition à base d'esquisses et d'exemples se caractérisent par un haut niveau de contrôle. Cependant, l'intervention de l'utilisateur est souvent très importante et la modélisation de milliers d'objets devient rapidement laborieuse.

Les techniques de génération procédurale permettent de créer rapidement une très grande quantité d'objets variés et de paysages à partir de très peu d'informations. Le contrôle du résultat est plus difficile et peut aller jusqu'à rendre l'utilisation de la méthode rédhibitoire pour certains types d'utilisateurs tels que les artistes.

Enfin, les méthodes de simulation permettent de suivre l'évolution d'un phénomène au cours du temps. Le réalisme et la complexité sont reproduits et simulés. D'autre part, la dimension temporelle offre une dynamique au contenu graphique généré. Toutefois, l'utilisateur n'a que très peu de contrôle sur le résultat final puisque le phénomène est régi seulement par un certain nombre de lois physiques. De plus, les calculs complexes permettant de résoudre les équations conduisent à des temps de calcul élevés.

Il existe également des méthodes qui se trouvent aux frontières des deux ou des trois catégories et qui ont pour objectif de tirer parti des avantages de chacune des catégories. Par exemple, certaines méthodes procédurales s'inspirent de phénomènes physiques ou phénoménologiques afin de dériver des équations empiriques. La prise en compte de ces équations permet d'accroître le réalisme et d'ajouter une dimension temporelle tout en conservant le contrôle et la rapidité de génération.

1.2 Édition interactive

Ces méthodes permettent de créer rapidement du contenu graphique en essayant de limiter au maximum les interventions de l'utilisateur. Les approches consistent à proposer des algorithmes qui génèrent des objets approximant au mieux les esquisses fournies par l'utilisateur. Les interfaces sont conçues pour être intuitives et offrent

beaucoup de contrôle à l'artiste. Nous pouvons classer les techniques d'édition en trois catégories : génération par esquisses, génération à partir d'exemples en trois dimensions et génération à partir d'images.

Les esquisses effectuées en deux dimensions et représentant la forme de l'objet souhaité sont analysées et interprétées par le système pour générer le maillage en trois dimensions.

La modélisation par l'exemple consiste à créer de nouveaux objets à partir d'objets existants. Ces méthodes sont majoritairement ciblées pour des utilisateurs qui ne possèdent aucune expérience en modélisation.

Les techniques de génération à partir d'images consistent à reconstruire des objets en utilisant une ou plusieurs images prises sous différents points de vue. L'intervention de l'utilisateur est limitée au strict minimum et les résultats obtenus sont très proches des images exemples.

1.2.1 Objets génériques

Funkhouser et coll. (FKS+04) et Sharf et coll. (SBSCO06) ont proposé des méthodes similaires permettant d'assembler des morceaux de maillage pour créer de nouveaux objets. Les morceaux de maillage sont obtenus par découpage selon une esquisse effectuée par l'utilisateur. L'assemblage s'effectue en raccordant les contours de découpe. Les raccords sont finalement créés puis lissés pour obtenir une bonne continuité entre les différents morceaux. D'autres méthodes utilisent une représentation différentielle du maillage (SCOL+04, YZX+04) et permettent, entre autres, l'assemblage de différents morceaux de maillage, le transfert de textures géométriques ainsi que des opérations de déformation préservant les détails géométriques et la forme globale de l'objet.

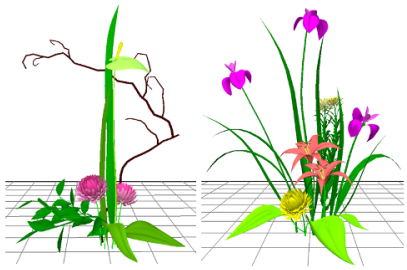
Les méthodes de métamorphose (LDSS99, ACOL00, SCFRC01) sont intimement liées à la modélisation par l'exemple. La métamorphose consiste à créer une transition la plus naturelle possible entre un maillage initial et un maillage final. La difficulté principale est d'effectuer une mise en correspondance optimale des maillages. Cette étape est en général effectuée manuellement et peut être très laborieuse lorsque les maillages comportent énormément de polygones. De plus, la métamorphose entre deux objets complexes tels que des arbres de morphologie très différente peut engendrer de nombreux recoupements des branches rendant inexploitables les objets intermédiaires générés.

1.2.2 Plantes

Depuis quelques années, cette approche a fait l'objet de nombreux travaux pour modéliser des arbres et des plantes. Une des premières méthodes a été proposée par Okabe et coll. (OOI05) et permet à l'utilisateur de définir la structure des branches de l'arbre. Les branches et les feuilles sont répliquées sur toute la structure par un mécanisme de propagation. Les auteurs ne prennent pas en compte de paramètres botaniques afin de guider la génération et les résultats peuvent parfois avoir un aspect non naturel.

La méthode proposée par Wither et coll. (WBCG09) prend en compte un certain nombre de paramètres botaniques permettant de relier les branches de façon plus réaliste et d'obtenir une meilleure distribution. La création de l'arbre s'effectue incrémentalement en dessinant la silhouette des feuillages à différents niveaux de la structure. Cette approche permet de contrôler parfaitement l'apparence globale de l'arbre. Cependant, l'utilisateur est constamment sollicité. Créer un grand nombre d'arbres devient rapidement fastidieux.

L'utilisation d'une méthode probabiliste proposée par Chen et coll. (CNX+08) permet d'obtenir des arbres complexes et réalistes à partir d'une esquisse très grossière. La création d'un nombre élevé d'arbres est alors beaucoup plus simple. L'approche consiste à extraire les paramètres initiaux de l'arbre d'une base de données en analysant la similarité entre l'esquisse et les modèles stockés. Le placement dans l'espace et la suppression des intersections de branches sont obtenus par l'optimisation d'un champ de Markov. Le modèle est automatiquement amplifié par ajouts de branches en utilisant le principe d'auto-similarité. Le feuillage est finalement généré en fonction d'un choix de feuilles effectué par l'utilisateur. Un inconvénient majeur de cette méthode est qu'il est nécessaire de créer une base de données importante pour pouvoir générer n'importe quel type d'arbre.



En 2005, Ijiri et coll. (IOOI05) ont introduit les diagrammes de structures de plantes et de fleurs afin d'organiser de façon correcte (au sens botanique) les plantes en floraison. L'ajout de ces connaissances botaniques leur permet d'avoir une méthode très réaliste et très adaptée à la création de plantes. L'arrangement de la plante reste cependant contraint par l'utilisation de diagrammes ce qui n'offre pas la possibilité de créer des plantes originales. Afin de rendre la création d'une plante plus générale, Ijiri et coll. (IOI06) ont proposé d'utiliser une hiérarchie de plans. Chaque plan permet d'effectuer l'esquisse de chacune des parties de la plante avec la possibilité de sauvegarder l'esquisse pour une utilisation ultérieure. L'arrangement de la plante reste cependant à la charge de l'utilisateur et n'est absolument pas guidé par des critères botaniques. De plus, le nombre d'interactions nécessaires est important.

Les méthodes décrites précédemment ne permettent pas de générer des variations d'une même plante. Pour cela, il est nécessaire d'effectuer l'ensemble du processus à chaque fois. Pour résoudre ce problème, Streit et coll. (SLSS06) ont proposé une interface permettant de modifier la forme globale d'une plante, fournie par l'utilisateur, à l'aide d'esquisses directement réalisées en trois dimensions. L'esquisse est ensuite utilisée pour guider une simulation biologique permettant de créer rapidement des variétés du modèle. L'inconvénient de cette méthode est qu'elle dépend de la représentation de la plante qui est, en l'occurrence, un L-Système.

De nombreuses méthodes ont été proposées pour générer des arbres de synthèse à partir d'une ou plusieurs images d'arbres réels. Par exemple, l'approche utilisée par Shlyakhter et coll. (SRDT01) consiste à créer l'enveloppe visuelle d'un arbre à partir d'un ensemble de photographies. Le squelette de l'arbre est extrait en calculant l'axe médian de l'enveloppe et l'arbre est finalement généré à l'aide du squelette et d'un L-Système contraint à la forme de l'enveloppe. Les résultats obtenus sont très proches des images fournies en entrée.

Une approche volumétrique a été proposée par Reche-Martinez et coll. (RMMD04). L'arbre est considéré comme un volume possédant des valeurs d'opacité et de couleur. L'opacité de la grille est estimée en fonction des photographies puis, pour chaque élément de la grille, un morceau de texture est assigné. Le rendu est réalisé à l'aide d'un algorithme prenant en compte le point de vue de l'observateur. Les résultats sont très réalistes. Cependant, la structure volumique est coûteuse en mémoire et il est très difficile de pouvoir éditer le résultat final. De plus, les changements d'illumination sont impossibles car l'auto-ombrage initial de l'arbre est également stocké dans la structure.

Neubert et coll. (NFD07) ont proposé une méthode pour générer des arbres à partir de plusieurs photographies et d'une simulation de particules. Un volume de voxels, contenant des valeurs de densité de feuillage, est créé en utilisant les photographies. Des particules sont ensuite initialisées en fonction des densités puis déplacées à l'aide de forces. Les brindilles et les branches sont créées à l'aide de la distribution finale des particules. L'inconvénient de leur méthode est que l'utilisateur doit intervenir pour spécifier les valeurs de densité lorsque le feuillage n'est pas disponible. De plus, il est nécessaire d'avoir une vue complète de l'arbre pour que la reconstruction soit la plus fidèle possible et cela n'est pas toujours le cas.

Tan et coll. (TZW$^+$07) ont proposé une méthode capable de reconstruire un arbre en utilisant une vue partielle de celui-ci. L'approche consiste à segmenter les photographies afin d'extraire la structure visible des branches et des feuilles. Les branches masquées par le feuillage sont ensuite reconstruites en utilisant le principe d'auto-similarité. Toutefois, l'inadéquation entre les branches reconstruites et le feuillage extrait des images a tendance à produire des feuilles flottantes. En utilisant une approche similaire, Quan et coll. (QTZ$^+$06) ont proposé une méthode permettant de reconstruire des plantes.

1.2.3 Terrains

Quelques méthodes ont été proposées pour la génération de terrains par esquisses (WI04, CHZ00). À l'heure actuelle, la méthode la plus aboutie est celle proposée par Gain et coll. (GMe09). Le terrain est interactivement généré à partir de silhouettes et de régions définies par un ensemble d'esquisses. Les esquisses sont approximées en utilisant des déformations de surface multirésolution. Des détails géométriques sont ajoutés par propagation du bruit ondelette introduit par Cook et coll. (CD05).

Pour la génération de terrains, Zhou et coll. (ZSTR07) se sont inspirés des techniques de génération de textures par l'exemple et plus particulièrement de celle proposée par Wu et coll. (WY04). Les terrains exemples sont représentés par des champs de hauteurs stockés sous forme d'images. Ces images sont découpées en morceaux rectangulaires puis assemblées à l'aide d'un algorithme minimisant le coût des différences d'altitude le long du chemin de jointures. Ces jointures sont finalement lissées en utilisant les équations de Poisson pour créer des transitions sans discontinuité. La forme globale du terrain peut également être contrôlée à l'aide de contraintes définies par l'utilisateur.

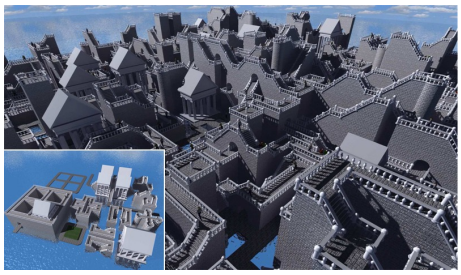
1.2.4 Architecture

Chen et coll. (CKX+08) sont les seuls à avoir utilisé un système à base d'esquisses pour créer des façades de bâtiment réalistes. La création de la façade s'effectue en analysant les jonctions entre les esquisses successives. La géométrie générée est ensuite amplifiée à l'aide d'éléments préexistants complexes et de textures stockées dans une base de données. Le choix des éléments et des textures s'effectue en analysant la similarité entre l'esquisse réalisée par l'utilisateur et les textures ou les objets stockés dans la base de données. Malgré de bons résultats, il est impossible de générer entièrement la façade du bâtiment. Par conséquent, elle peut être visualisée en utilisant seulement un nombre limité de points de vue.

Cabral et coll. (CLDD09) proposent une méthode spécialement dédiée à la création de grandes scènes architecturales construites à partir d'un petit ensemble de modèles texturés pré-existants (pièces) qui sont déformés et assemblés. La déformation des pièces s'effectue sous la contrainte de préservation des angles en modifiant la longueur des arêtes interactivement. La texture est automatiquement redimensionnée en fonction de la déformation de façon à préserver les détails et l'aspect visuel de celle-ci.

Merrell (Mer07) propose une méthode procédurale inspirée des techniques de génération de textures par l'exemple (WL00, EF01). Un ensemble de morceaux est obtenu

par découpage de modèles exemples à l'aide d'une grille. Ces morceaux sont ensuite assemblés pour créer de nouveaux objets et de vastes environnements similaires aux exemples fournis.



Cependant, l'approche utilisée limite la génération aux objets de type architecturaux qui possèdent des éléments répétitifs et qui peuvent être structurés sous forme de grille. Plus récemment, sa méthode a été adaptée pour prendre en compte des maillages de topologie arbitraire (MM08). Toutefois, les résultats illustrent principalement des scènes architecturales et démontrent l'efficacité de la méthode seulement pour cette catégorie d'objets. De plus, la forme finale des objets générés n'est absolument pas contrôlable. Cela produit des géométries parfois surprenantes.

Une méthode a également été proposée pour modéliser des façades de bâtiments à partir d'images (MZWG07). Les informations de position et la forme des éléments architecturaux sont extraits de l'image pour en déduire la grammaire de la façade. La modification de la grammaire permet de générer des variations de la façade. La géométrie est finalement générée à l'aide de la grammaire et la texture est obtenue à partir de l'image de la façade.

Toutefois, l'intégralité du bâtiment ne peut être générée, seule la façade visible est reconstruite. En utilisant une approche similaire, Aliaga et coll. (ARB07) sont capables de construire une grammaire représentant l'intégralité d'un bâtiment à partir d'une seule image. D'autres bâtiments, d'apparence similaire à l'original, peuvent être facilement générés en modifiant la grammaire.

Aliaga et coll. (AVB08) ont proposé une méthode pour générer des réseaux urbains à partir de vues aériennes de villes. La structure du réseau est extraite et étendue en utilisant des connaissances d'aménagements urbains. L'image aérienne du nouveau réseau est obtenue en assemblant des morceaux d'image découpés dans l'image initiale. L'utilisateur peut également créer de nouveaux réseaux en utilisant des opérations de fusion, d'expansion et de mélange entre deux images aériennes.

1.2.5 Conclusion

La création de contenu graphique en utilisant des méthodes à base d'esquisses est relativement rapide. En seulement quelques minutes, l'utilisateur est capable d'obtenir un résultat visuellement satisfaisant et peut exprimer librement sa fibre artistique. De plus, l'utilisateur n'a pas besoin d'avoir de connaissances, artistiques ou techniques,

spécifiques.

Cependant, cette approche a un certain nombre de limitations. Pour modéliser un type d'objets particulier, il est nécessaire de développer des algorithmes spécifiques. Nous avons vu précédemment que les méthodes proposées permettent de modéliser exclusivement des plantes ou des éléments architecturaux. Aucune méthode, à ce jour, n'est capable d'être suffisamment générique pour générer aussi bien des arbres que des maisons. De plus, une fois que l'objet a été modélisé, il n'est pas possible de créer des variations directement à partir de celui-ci. Le processus doit être réitéré du début à la fin et la création de nombreux modèles devient très rapidement laborieuse. Finalement, un objet créé correspond à un objet devant être stocké. Plus le nombre d'objets créés est important, plus l'espace mémoire nécessaire au stockage l'est également. Ces méthodes deviennent donc inexploitables dans des contextes où l'espace mémoire est limité. C'est le cas dans l'industrie du jeu vidéo.

L'utilisation d'exemples et l'assemblage de morceaux de géométrie préexistante est une bonne solution pour créer des variantes d'un objet. La majorité des méthodes présentées offre un bon contrôle sur le résultat final et l'utilisateur n'a pas besoin d'avoir de connaissances particulières en modélisation. Ces méthodes sont, en général, simples d'utilisation et le mécanisme sous-jacent n'a pas besoin d'être compris par l'utilisateur.

Néanmoins, comme pour les méthodes à base d'esquisses, un objet créé correspond à un objet stocké en mémoire. De plus, dans certains cas, la création de raccords peut engendrer un nombre de polygones très élevé. Lorsque les rendus doivent être réalisés en temps réel, le nombre de polygones est une donnée critique et doit être contrôlé avec précision.

Les résultats obtenus par ce type d'approche sont très réalistes et proches des images utilisées en entrée des algorithmes. Les méthodes proposées ne nécessitent, en général, aucune intervention de l'utilisateur. De ce fait, la création de milliers d'objets est très facile.

Malgré cela, les systèmes sont développés en fonction du type d'objets à reconstruire et ne sont pas généralisables à d'autres catégories d'objets. D'autre part, le nombre de polygones générés pour obtenir la qualité des résultats présentés est souvent très important. L'utilisation de ce type d'approche dans une plateforme de rendu en temps réel est donc délicate.

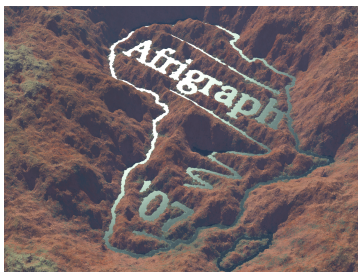
1.3 Génération procédurale

Les techniques de génération procédurale consistent à créer rapidement des modèles géométriques en utilisant un ensemble d'algorithmes et de règles. Les paramètres des procédures algorithmiques ou des grammaires permettent de générer des variétés d'objets.

1.3.1 Génération de terrains

La génération du terrain est une étape fondamentale pour la création d'une scène. Certains travaux portent sur la reconstruction de terrains à partir de données réelles. Dans cette section, nous nous intéresserons exclusivement aux méthodes de génération procédurale permettant d'obtenir des terrains artificiels.

1.3.1.1 Carte d'élévation



L'utilisation des méthodes fractales pour représenter les formes naturelles a été mise en évidence par Mandelebrot dans son ouvrage intitulé *The Fractal Geometry of Nature* (Man82). Depuis, l'utilisation des formes fractales a été adaptée à la génération de terrains. Les approches proposées consistent à effectuer des subdivisions récursives de la surface en ajoutant un facteur aléatoire de déplacement vertical. Les algorithmes *diamond-square* (Mil86) de Miller et *square scheme* (FFC82) de Fournier et coll. sont parmi les plus connus. La technique proposée par Musgrave et coll. (MKM89) utilise le bruit de Perlin (Per85) combiné à un processus d'érosion hydraulique et thermique pour donner un aspect plus naturel au terrain généré. Des algorithmes *diamond-square* prenant en compte des contraintes telles que des rivières et des crêtes ont également été proposés (PH93, BA05b, BA05a, Bel07).

La représentation utilisée par ces méthodes est dite 2.5D. Le maillage du terrain est obtenu en utilisant une discrétisation régulière sous forme de grille et les sommets sont déplacés verticalement. Avec ce type de représentation, il est impossible de modéliser certaines caractéristiques largement présentes dans la nature telles que des abris sous roches, des surplombs ou des cavernes. Ce problème a été partiellement résolu par Gamito et Musgrave (GM01) pour la création de surplombs. Leur méthode consiste à utiliser un champ vectoriel permettant d'appliquer des déformations non linéaires de la surface d'un terrain obtenu par élévation. Cependant, il est très difficile de contrôler avec exactitude la déformation et par conséquent la forme du surplomb.

1.3.1.2 Représentation volumique

Peytavie et coll. (PGMG09) ont développé un système permettant l'édition de terrains complexes multi-matériaux en s'appuyant sur une structure volumique. Les outils d'ajout et de suppression de matière proposés permettent la création rapide d'abris sous roches, de surplombs et de cavernes. Les matériaux granuleux tels que le sable et les cailloux sont stabilisés à l'aide d'un algorithme d'angle au repos. La surface du terrain est extraite

par produit de convolution et est texturée par projection et mélange de textures. Des empilements réalistes de cailloux sont également générés procéduralement. Toutefois, la représentation volumique a un coût non négligeable en mémoire. À l'inverse des approches fractales, il n'est pas possible de créer des terrains de plusieurs dizaines de kilomètres. De plus, l'approche édition peut rendre laborieuse la création de paysages de cette taille.

1.3.1.3 Conclusion

La création de terrains à l'aide d'algorithmes fractals est, à l'heure actuelle, l'approche la plus utilisée. De nombreux logiciels commerciaux utilisent ce type d'algorithmes. D'autre part, ces méthodes ont une représentation compacte. Seuls quelques paramètres ainsi que les algorithmes sont nécessaires pour générer des terrains gigantesques. Malgré cela, la représentation 2.5D ne permet pas de modéliser des caractéristiques plus complexes qui nécessitent une description en trois dimensions de l'espace. De plus, les algorithmes ont tendance à générer principalement des reliefs de type montagneux et il est difficile de contrôler correctement la forme du terrain final. Bien que l'approche volumique permette de modifier interactivement la géométrie du terrain, offrant ainsi beaucoup de contrôle, la structure doit être stockée et ne peut pas être automatiquement régénérée par des algorithmes procéduraux.

1.3.2 Génération de végétation

Les plantes et les arbres font partie intégrante des paysages. De par leur structure et leur texture complexes, ce sont des objets très difficiles à modéliser de façon réaliste.

1.3.2.1 L-Système

La plupart des méthodes proposées s'appuient sur le mécanisme de réécriture de règles et de grammaire formelle introduit par Lindenmayer (Lin68) en 1968 : les L-Systèmes. Initialement conçus pour décrire le développement d'organismes multicellulaires, ils ont ensuite été étendus pour décrire le développement de plantes. Un L-Système est composé d'un ensemble de règles et de symboles qui définissent une grammaire. Étant donnée une chaîne de symboles initiale, celle-ci est itérativement réécrite en utilisant les règles de la grammaire pour obtenir une chaîne plus complexe. Cependant, la structure générée est beaucoup trop régulière pour obtenir des arbres réalistes. Ce problème a été résolu en utilisant des L-Systèmes paramétriques introduits par Prusinkiewicz et Lindenmayer (PL90). Le principe consiste à ajouter des paramètres de contrôle aux règles ainsi qu'une dépendance au prédécesseur. Depuis, les L-Systèmes

ont fait l'objet de nombreux travaux pour accroître le réalisme des plantes générées et offrir plus de contrôle à l'utilisateur (BPF+03). Les méthodes proposées permettent, par exemple, d'animer le développement des plantes (PHM93) ou de les contraindre à un volume donné (PJM94). Bien que les résultats soient de très bonne qualité, la structure de l'arbre n'est pas toujours très réaliste. Lors de leur développement, les plantes ont deux objectifs principaux : chercher la lumière et conquérir le plus d'espace possible. Les Open L-Systèmes proposés par Měch et Prusinkiewicz (MP96) permettent de simuler la compétition entre les branches pour la lumière et l'espace en prenant en compte l'environnement. Les arbres obtenus sont beaucoup plus réalistes. Depuis, d'autres méthodes utilisant ce type d'approche ont été proposées (RLP07, PHL+09). Bien que l'approche L-Système donne de bons résultats, les paramètres sont peu intuitifs pour un utilisateur non expérimenté.

1.3.2.2 Approche géométrique

Weber et Penn (WP95) ont proposé une approche complètement différente en manipulant des paramètres purement géométriques beaucoup plus intuitifs pour un utilisateur. Ils partent du principe que les branches d'un arbre ne possèdent pas plus de quatre niveaux de récursivité. À chaque niveau, un ensemble de paramètres géométriques est spécifié dans une plage de valeurs possibles. Leur approche permet également de générer des arbres de complexité géométrique variable. Elle est donc bien adaptée aux moteurs de rendu temps réel à niveau de détails. Le système proposé par Deussen et coll. (DL97) et plus connu sous le nom de XFrog [1], tire avantage des approches géométriques et à base de règles. Dans XFrog, la plante est définie par un graphe définissant la structure et chaque nœud et feuille du graphe possèdent un ensemble de paramètres et de fonctions de modélisation. Bien que ces méthodes soient plus intuitives, le nombre de paramètres à manipuler est très important et il est difficile d'obtenir facilement le résultat souhaité.

1.3.2.3 Approche botanique

Une approche botanique a également été proposée par De Reffye et coll. (REF+88). La plante est considérée comme étant un ensemble de bourgeons pouvant devenir une fleur ou un embranchement de plusieurs tiges mais également mourir ou rester dans le même état. Le choix est guidé par un ensemble de probabilités relatif à l'espèce végétale considérée, ainsi que son stade de développement. Des facteurs environnementaux peuvent également être pris en compte (vent, attaque d'insectes, gravité, lumière, espace). Cette méthode

1. http ://www.xfrog.com/

a été incorporée dans les logiciels proposés par la société *Bionatics*[2]. Les résultats obtenus sont réalistes d'un point de vue botanique. En revanche, il est impossible de créer des plantes originales qui ne répondent pas forcément à des critères botaniques. En utilisant également une approche botanique, Streit et coll. (SFS05) ont proposé une méthode permettant de générer des variations d'une plante représentée par un L-Système. Le L-Système est modifié pour pouvoir simuler la croissance de la plante. Les variétés sont obtenues en stimulant le mécanisme de croissance biologique par des facteurs environnementaux auxquels les plantes répondent. Leur méthode permet d'obtenir rapidement de nombreuses plantes toutes légèrement différentes. Cependant, elle dépend de la représentation initiale de la plante (L-Système) et ne peut donc pas être utilisée pour générer des variétés de plantes d'un modèle initial représenté par un maillage.

1.3.2.4 Conclusion

Au fil des années, l'utilisation des L-Systèmes ainsi que la prise en compte de paramètres botaniques et géométriques ont prouvé leur efficacité pour la génération de plantes réalistes. Toutefois, les systèmes proposés sont difficiles à contrôler. En effet, le nombre de paramètres à manipuler étant très important, la moindre modification peut modifier considérablement la forme globale de la plante. Il est donc nécessaire d'avoir une bonne expertise du système pour réussir à générer la plante souhaitée. D'autre part, dans certains cas, le nombre de polygones générés peut être très important rendant le rendu de milliers d'arbres en temps réel impossible. Finalement, chaque modèle est stocké en mémoire et ceci peut devenir très coûteux lors de la création de forêts.

1.3.3 Génération de villes

La création d'une ville consiste en un ensemble de bâtiments, de quartiers, de blocs et de parcelles interconnectés par des rues (VAW$^+$10). Modéliser ces paysages denses et leur agencement de façon réaliste est un problème majeur. L'approche généralement utilisée consiste d'abord à créer un réseau de rues principales définissant les quartiers, puis de générer les rues secondaires pour former les blocs. Les blocs sont ensuite subdivisés pour former des parcelles et les bâtiments sont finalement générés et placés sur ces parcelles.

1.3.3.1 Réseau routier

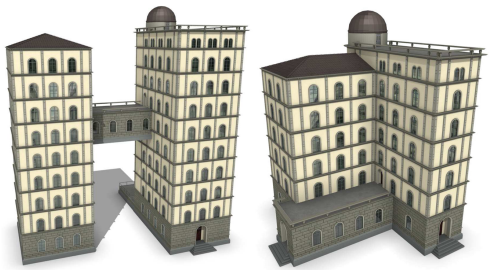
CityEngine (PM01) est le premier système permettant de générer une ville complète en trois dimensions. Le réseau routier est généré en utilisant un L-Système qui a été

2. http ://www.bionatics.com/

modifié pour prendre en compte un certain nombre de contraintes. La croissance du L-Système est guidée par un ensemble de cartes représentant des statistiques sociales (densité de population, type de quartier) et des obstacles (élévation, présence d'eau, végétation). Le réseau est ensuite divisé en parcelles et les bâtiments sont générés sur ces parcelles en utilisant également un L-Système. Une méthode récente consiste à combiner les techniques de génération procédurale à la simulation urbaine (WMWG09). Leur approche permet de suivre l'évolution d'une ville au cours du temps. L'extension du réseau routier est réalisée en simulant le trafic. La simulation de l'utilisation des zones leur permet de définir quels types de bâtiments doivent être construits (usines, gratte-ciel, résidences). L'utilisateur peut interactivement modifier les paramètres pour contrôler la simulation. Ces méthodes permettent de créer des villes réalistes.

Afin de générer exclusivement des réseaux de routes sous forme de graphes, Chen et coll. (CEW$^+$08) se sont appuyés sur un champ tensoriel qui sert à guider le processus de génération. La modification du champ tensoriel à l'aide d'outils adaptés leur permet de contrôler précisément la forme du réseau routier. L'utilisateur a également la possibilité d'éditer manuellement le graphe.

1.3.3.2 Bâtiments



La première approche permettant de générer procéduralement des bâtiments a été d'utiliser les L-Systèmes (PM01). Cependant, les bâtiments générés sont principalement des gratte-ciel et les façades sont créées à l'aide de simples textures. Afin d'accroître la diversité et la complexité des bâtiments et des façades, Wonka et coll. (WWSR03) ont introduit les *Split-Grammars*. Le bâtiment est créé en utilisant des formes simples (cube, cylindre, prisme) qui sont récursivement découpées à l'aide de plans et en fonction des règles définies par la grammaire. Toutefois, la forme des bâtiments générés reste contrainte aux formes de bases. Pour surmonter ces limitations, une nouvelle grammaire appelée *CGA shape* a été introduite par Müller et coll. (MWH$^+$06). Cette grammaire est une extension en trois dimensions des grammaires de forme de Stiny (Sti80) ainsi que des Split-Grammars (WWSR03). Des règles supplémentaires permettant des transformations géométriques (rotation, translation, mise à l'échelle) offrent plus de flexibilité. De plus, l'assemblage de formes élémentaires et la gestion des occlusions entre les formes permettent de générer des bâtiments très complexes et de correctement positionner les ouvertures. Cependant, l'écriture des règles de construction reste accessible seulement aux utilisateurs confirmés et peut être fastidieuse.

Lipp et coll. (LWW08) ont proposé un système permettant de créer et d'éditer visuellement la grammaire proposée par (MWH$^+$06), sans avoir à modifier un fichier texte. Pour permettre l'édition locale et persistante en évitant l'explosion combinatoire des règles, la grammaire a été étendue pour qu'une modification n'affectant pas l'en-

semble des instances générées par une règle devienne possible. Bien que la création et l'édition des règles soient plus intuitives, le mécanisme interne doit être compris par l'utilisateur pour complètement maîtriser le système.

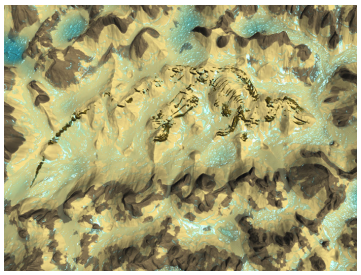
1.3.3.3 Conclusion

Les L-Systèmes et les grammaires de formes se sont avérés très efficaces pour la génération de réseaux routiers urbains et de bâtiments. Les résultats démontrent la pertinence de ce type d'approche pour la création de villes de tous types. Malgré tout, l'utilisation de ces systèmes n'est pas forcément évidente pour tous les utilisateurs. Une certaine expertise est nécessaire. Bien qu'une interface a été récemment proposée pour faciliter l'édition des règles, l'utilisateur doit cependant comprendre le fonctionnement de la grammaire pour pouvoir créer des bâtiments.

1.4 Simulation

La dimension temporelle que fournit la plupart des méthodes de simulation permet de générer du contenu dynamique. Les objets et paysages évoluent au cours du temps. Ceci se traduit par un changement d'apparence pouvant être éphémère (présence de neige en hiver) ou permanent (fissures, rouille). L'aspect synthétique, car trop parfait du contenu, devient alors beaucoup plus réaliste.

1.4.1 Érosion de terrains



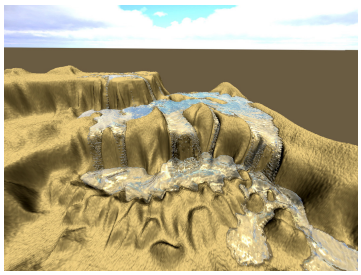
Au fil des années, le terrain est soumis à différents types d'érosion qui modifient son apparence globale. L'érosion se déroule en deux étapes. La première étape correspond au déplacement de matière provoqué par le vent, par des chocs thermiques et par l'eau. La seconde étape est la sédimentation et se traduit par le dépôt de la matière déplacée. Bien que les phénomènes d'érosion soient multiples, la plupart des algorithmes proposés utilisent seulement l'érosion hydraulique couplée à des modèles de sédimentation. En effet, l'érosion hydraulique est celle qui a le plus d'impact visuel sur le terrain.

1.4.1.1 Méthode empirique

Musgrave et coll. (MKM89) ont proposé deux algorithmes simulant l'érosion du terrain : le premier correspond au mouvement de sédiments provoqué par des chocs thermiques et le second simule le transport de sédiments et leur dépôt par l'eau. Ces algorithmes permettent de lisser le terrain, initialement obtenu par un processus fractal,

et d'obtenir un résultat plus naturel. Néanmoins, les équations utilisées sont empiriques et traduisent donc de façon approximative le phénomène.

1.4.1.2 Mécanique des fluides



Capturer le phénomène dans sa complexité nécessite de simuler le mouvement de l'eau en utilisant la mécanique des fluides ainsi qu'un modèle d'érosion, de transport et de sédimentation physique. L'érosion s'effectue en calculant la quantité de sédiments transportables par l'eau et en fonction des facteurs de dissolution des matériaux. Beneš et coll. (BTHB06) utilisent une approche eulérienne et simulent le mouvement de l'eau en utilisant les équations de Navier-Stokes ainsi que les équations du transport de sédiments pour l'érosion et la sédimentation. Krištof et coll. (KBKv09) ont proposé une méthode similaire à la différence qu'ils utilisent une approche Lagrangienne (SPH) pour la résolution des équations de Navier-Stokes. La simulation s'effectuant entièrement sur une grille en trois dimensions, la complexité en mémoire et en calcul est très importante. Par conséquent, la taille des scènes simulées reste relativement modeste.

1.4.1.3 Eaux peu profondes

L'utilisation de champs de hauteurs permet de simplifier le problème et d'obtenir des temps de simulation interactifs sur des scènes beaucoup plus grandes. Les méthodes proposées utilisent les équations du mouvement de l'eau en milieu peu profond (modèle en tube). Le champ de vélocités est calculé à l'aide de ces équations et le transport des sédiments s'effectue par advection. L'eau est évaporée de façon empirique en utilisant une constante d'évaporation (MDH07). L'approche utilisée par (vBBK08) est identique. Cependant, le terrain est représenté par plusieurs champs de hauteurs possédant différents matériaux et le sable est stabilisé à l'aide d'un algorithme d'angle au repos. L'avantage de ces méthodes est qu'elles sont extrêmement parallélisables et peuvent donc être directement implémentées sur la carte graphique. Malgré d'importantes simplifications, les résultats obtenus restent physiquement réalistes.

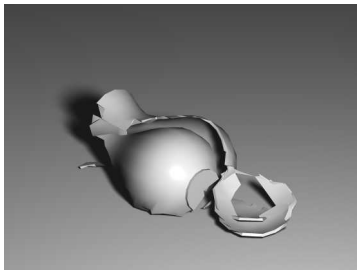
1.4.1.4 Conclusion

L'utilisation d'approches physiques pour simuler l'érosion des terrains a permis de modéliser le phénomène dans toute sa complexité. Les simplifications et les implémentations réalisées par la suite ont considérablement accéléré les temps de calcul tout en conservant une bonne réalité physique. Ce type de méthodes est très intéressant dans

le cadre de la génération de contenu graphique car celui-ci n'est plus généré de façon statique, mais de façon dynamique.

Néanmoins, l'utilisateur ne peut pas contrôler la trajectoire du fluide de façon à éroder le terrain comme il le souhaite.

1.4.2 Fissures et fractures

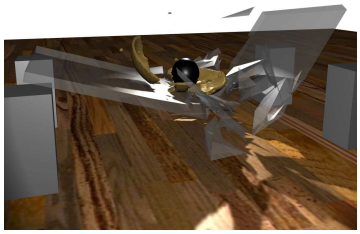


Avec le temps, la matière se fragilise et les forces de tension présentes dans le matériau conduisent à l'apparition de fissures et de fractures en surface. Les modéliser permet d'obtenir des objets beaucoup plus réalistes. Par conséquent, de nombreux travaux ont été proposés pour simuler la formation des fissures et des fractures sur les objets à l'aide d'équations physiques. En général, la résolution des équations physiques s'effectue en utilisant une des trois méthodes suivantes : les différences finies, les éléments finis ou les systèmes masses-ressorts.

1.4.2.1 Différences finies

Terzopoulos et coll. (TPBF87, TF88) ont proposé un modèle utilisant les différences finies pour représenter les comportements inélastiques (viscosité, plasticité et fractures) de la matière. Leur simulation est réalisée en deux dimensions et permet de simuler la déchirure de feuilles de papier et de tissus. Les résultats obtenus sont physiquement réalistes. Cependant, la discrétisation régulière nécessaire pour la résolution par différences finies produit des fissures fortement crénelées. De plus, les différences finies ne garantissent pas la conservation de la masse et ne peuvent être utilisées que sur des formes relativement simples.

1.4.2.2 Éléments finis



La résolution des équations en utilisant la méthode des éléments finis est une solution à ces problèmes. Le maillage des objets est représenté à l'aide de tétraèdres qui sont dynamiquement subdivisés pour augmenter la résolution le long de la fissure. Le maillage étant adaptatif, les artéfacts engendrés par la discrétisation ne sont plus visibles et les résultats obtenus sont beaucoup plus réalistes. O'Brien et Hodgins (OH99) ont utilisé ce type d'approche pour fissurer des objets. Le principe mis en œuvre est de soumettre un objet à des tenseurs de déformation exprimés par les lois de la mécanique des milieux continus. Le plan de fracture est calculé en fonction des forces de tension. Leur méthode a été étendue (OBH02) pour pouvoir prendre en

compte le cas des matériaux ductiles. Le système proposé par Federl et Prusinkiewicz (FP04) permet de modéliser l'apparition de fractures sur l'écorce d'un arbre et sur le sol. L'objet est composé de deux couches et la fissure est créée en supprimant les tétraèdres dont le stress est supérieur au seuil du matériau de la première couche. Bien que les résultats obtenus en utilisant les éléments finis soit de très bonne qualité, le nombre de tétraèdres nécessaires est important et les temps de calcul sont donc élevés.

1.4.2.3 Masses-ressorts

Les systèmes masses-ressorts sont beaucoup moins coûteux en temps de calcul et relativement simples à implémenter par rapport aux méthodes utilisant les éléments finis. Les forces de tension exercées sur le système provoquent la rupture des ressorts créant ainsi une fissure. La rupture se produit lorsque le ressort atteint son seuil maximal d'élasticité. Le système masses-ressorts est généralement attaché à un maillage représentant l'objet. Norton et coll. (NTB$^+$91) ont proposé une méthode permettant de fissurer des objets en utilisant un maillage composé de cubes. Ces cubes sont déformés pour approximer au mieux la forme des objets. La méthode proposée par Hirota et coll. (HTK98) consiste à attacher le système sur la surface de l'objet afin de simuler la formation de fissures lors de l'assèchement d'objets composés de terre. Cependant, leur modèle ne permet pas de simuler la propagation de la fissure au sein du volume. Pour résoudre cette limitation, ils ont étendu leur méthode en utilisant un maillage tétraédrique auquel est associé un réseau de ressorts (HTK00). Malgré des résultats physiquement réalistes et peu coûteux en temps de calcul, visuellement la fissure est très crénelée. Ceci s'explique par le fait que celle-ci se propage en suivant la structure du système masses-ressorts de résolution relativement faible.

1.4.2.4 Approches hybrides

D'autres approches dites hybrides ont pour objectif d'obtenir des résultats réalistes identiques à ceux obtenus par l'utilisation d'éléments finis et ce, en approchant les faibles temps de calcul des méthodes utilisant les systèmes masses-ressorts. Pour simuler le bris d'objets fragiles, Smith et coll. (SWB01) utilisent un ensemble de masses inter-connectées et structuré par un maillage de tétraèdres. Leur méthode consiste à préserver les distances entre les masses et leur permet donc de définir un système linéaire résolu par descente de gradient. Cette approche réduit considérablement les temps de simulation mais n'est valable que pour les objets fragiles. En reprenant les principes proposés par O'Brien et Hodgins (OH99) et en effectuant des simplifications, Müller et coll. (MMDJ01) sont capables de déformer et fracturer des objets en temps réel. Leurs simplifications consistent à évaluer les forces d'élasticité seulement lorsqu'une collision se produit. Cependant, les résultats obtenus ne sont pas très réalistes car les fissures sont fortement créne-

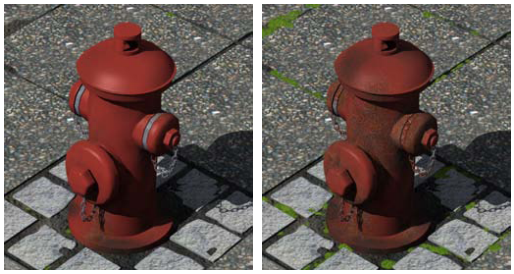
lées. L'algorithme des nœuds virtuels introduit par Molino et coll. (MBF04) permet de résoudre ce problème. Leur méthode produit des fractures dont l'orientation est totalement indépendante de la discrétisation en tétraèdres. Le principe est de dupliquer le maillage tétraédrique pour y stocker les zones dans lesquelles il n'y a plus de matière. Cette approche permet d'obtenir des résultats très réalistes.

La majorité des méthodes proposées utilisent un maillage de tétraèdres adaptatifs pour résoudre les équations. Le raffinement local du maillage permettant d'obtenir des résultats réalistes a un coût non négligeable. Pauly et coll. (PKA+05) ont proposé une méthode pour simuler le comportement d'objets plastiques et élastiques susceptibles de se fracturer sans utiliser de maillage. Pour ce faire, l'objet est échantillonné de façon adaptative par un ensemble de points auxquels ils associent une fonction de forme. La fracture est modélisée en utilisant un terme de discontinuité ajouté aux fonctions. Les résultats obtenus sont très réalistes et les temps de simulation considérablement inférieurs à ceux des méthodes utilisant les éléments finis.

1.4.2.5 Conclusion

L'utilisation des éléments finis pour la résolution des équations est l'approche qui produit les résultats les plus réalistes visuellement. Cependant, la discrétisation a un fort impact sur les temps de simulation. D'autre part, le stockage de cette discrétisation ne permet pas d'effectuer des simulations à grande échelle. L'utilisation de fonctions de forme est une approche très intéressante qui permet de se passer d'une discrétisation coûteuse. Toutefois, il est très difficile pour l'utilisateur de faire apparaître une fissure à un endroit précis. Un contrôle de la simulation plus intuitif et de plus haut niveau serait nécessaire.

1.4.3 Vieillissement



Le vieillissement des objets est provoqué par un certain nombre de phénomènes. Les conditions climatiques, la pollution et l'exposition à des substances chimiques peuvent provoquer l'apparition de salissures, de mousses, de lichens ou encore de patines. Les impacts et les rayures provoqués par des chocs ou des frottements contribuent également au vieillissement des objets. Les causes de vieillissement peuvent être classifiées en trois catégories : les attaques chimiques, les dommages mécaniques, et les attaques biologiques (MG08). Ces phénomènes ont été largement étudiés et permettent d'accroître considérablement la diversité visuelle et le réalisme des scènes.

1.4.3.1 Salissures

La méthode proposée par Dorsey et coll. (DPH96) permet de créer des salissures provenant du transport et du dépôt de saletés par la pluie. La pluie est représentée à l'aide de particules dont le déplacement est simulé en utilisant les équations du mouvement. La simulation prend également en compte les paramètres de rugosité de la surface et le taux d'absorption du matériau. Le dépôt des salissures utilise un modèle de sédimentation identique à ceux utilisés pour l'érosion de terrain. Le changement d'aspect est réalisé en modifiant la texture. La géométrie n'est donc pas modifiée. De plus, seulement les phénomènes de vieillissement par la pluie sont pris en compte.

L'importance de prendre en compte différentes sources de vieillissement a été mise en évidence par les travaux de Chen et coll. (CXW$^+$05). Leur approche permet de modéliser, entre autres, les coulures de rouille et l'apparition de mousses mais également l'accumulation de poussières et l'érosion. Le principe est le suivant : des particules appelées $\gamma - ton$ sont émises depuis la surface des objets en utilisant une approche similaire au tracé de photons. Les $\gamma - ton$ collectent ainsi les éléments responsables du vieillissement des objets. Les changements d'apparence sont obtenus en modifiant la texture et la géométrie des objets. Néanmoins, les modifications de la géométrie sont effectuées en utilisant une carte de déplacement et ne permet donc pas de modéliser la formation de trous.

1.4.3.2 Attaques chimiques

L'oxydation importante d'un matériau peut provoquer l'apparition de trous dans la surface. Pour modéliser ce phénomène, Mérillou et coll. (MDG01) ont proposé une méthode de rendu basée sur des mesures réelles. Leur simulation utilise une carte de corrosion qui permet de stocker les propriétés de la surface ainsi que la présence de trous. Un aspect intéressant de leur méthode est qu'elle ne modifie pas la géométrie de la surface pour faire apparaître les trous. Lors du rendu, l'utilisation de la carte de corrosion permet de faire apparaître les trous et de modifier localement la BRDF pour simuler l'interaction entre la lumière et la rouille. Cette approche donne des résultats très réalistes.

En 1999, Dorsey et coll. (DEJ$^+$99) ont proposé une méthode dédiée à la simulation du vieillissement de la roche. Leur approche consiste à simuler l'infiltration de l'eau en utilisant les équations de Darcy en fonction de la porosité de la roche. Ils simulent également la dissolution, la recristallisation et les transformations chimiques provoquées par l'eau infiltrée dans des minéraux. Le modèle volumique permet d'éroder la surface des objets. De plus, le moteur de rendu intègre une méthode de diffusion sous-surfacique de la lumière et augmente le réalisme des résultats.

1.4.3.3 Systèmes d'acquisition

Une autre approche consiste à capturer, en fonction du temps, l'apparence de surfaces réelles à l'aide de dispositifs d'acquisition (GTR$^+$06). Un ensemble de paramètres tels que la couleur et la réflectance sont ensuite extraits et utilisés pour créer une fonction de distribution de réflectance bidirectionnelle variant dans le temps et l'espace (TSV-BRDF). La TSV-BRDF est ensuite utilisée pour modifier la texture des objets au cours du temps. Lu et coll. (LGG$^+$07) utilisent également un dispositif de capture pour déterminer quels sont les paramètres géométriques (ou paramètres de contexte) qui influencent le plus les variations de texture pour un type de vieillissement donné. Après une série d'expérimentations, les paramètres de contexte retenus sont l'occlusion ambiante, la courbure, la direction principale ou bien encore l'épaisseur de la surface. Les textures capturées sont ensuite transférées sur d'autres objets de géométrie complètement différente, à l'aide d'un algorithme de synthèse de texture prenant en compte les paramètres de contexte.

1.4.3.4 Colonisation

Les méthodes précédentes modifient principalement la texture des objets. Cependant, dans certains cas, il n'est pas possible de représenter visuellement le phénomène par une texture. Pour simuler la propagation et la formation de lichens, Desbenoit et coll. (DGA04) ont introduit les *Open-DLA*. Leur méthode consiste à disperser des particules (spores) dans les zones de la scène qui sont protégées du vent et de la pluie. La propagation s'effectue en utilisant un algorithme d'agrégation à diffusion limitée qui prend en compte les paramètres environnementaux influençant les lichens dans leur développement. L'instantiation est réalisée à l'aide d'un ensemble de maillages texturés de lichens. Les résultats sont très réalistes. Cependant la complexité géométrique de la scène peut devenir très importante quand le nombre de lichens augmente.

1.4.3.5 Conclusion

L'impact visuel produit par le vieillissement est considérable et permet de masquer l'aspect synthétique des objets souvent trop parfaits. Les méthodes vues précédemment montrent qu'il est important de prendre plusieurs facteurs de vieillissement en considération. De plus, l'utilisation de BRDF lors des rendus permet d'obtenir des résultats beaucoup plus réalistes.

Néanmoins, le changement d'apparence est réalisé majoritairement en modifiant la texture. Les dégradations de la surface altérant la géométrie sont rarement prises en compte. De plus, les changements de température ainsi que les précipitations dégradent

également les objets et il serait important d'avoir un modèle climatique pour simuler ces phénomènes.

1.4.4 Neige et glace

En hiver, les chutes de neige, l'apparition de glace sur les lacs et la formation de stalactites changent considérablement l'aspect des paysages. De nombreuses méthodes ont été proposées pour simuler l'accumulation de la neige et peuvent être classifiées en deux catégories : les méthodes utilisant des particules et celles consistant à déplacer verticalement la surface pour représenter le manteau neigeux.

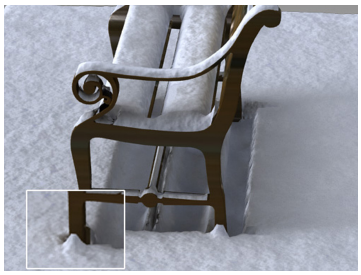
1.4.4.1 Particules

Les approches utilisant des particules consistent à calculer la distribution de la neige en évaluant la trajectoire des flocons transportés par le vent et s'accumulant sur le sol. Une des méthodes les plus abouties à l'heure actuelle et qui utilise ce principe est celle proposée par Fearing (Fea00). L'originalité de son approche est que les particules ne représentent pas les flocons mais sont utilisées pour collecter la neige. Les particules sont émises depuis la surface des objets en direction du ciel et en perturbant la trajectoire. Lorsqu'une particule atteint les nuages, celle-ci collecte une certaine quantité de neige qui est accumulée sur le sol. La neige est ensuite stabilisée en simulant des micro-avalanches. Les résultats obtenus sont très réalistes et la neige peut également s'accumuler sur des géométries très complexes telles que des arbres. Cependant, plus la complexité de la scène est importante, plus les temps de calcul augmentent.

De nombreuses améliorations de cette méthode ont été proposées afin de transporter les particules en simulant le vent. La simulation est réalisée en résolvant les équations de Navier-Stokes (FO02) ou de Boltzmann (WWXP06). Les résultats obtenus montrent une distribution de la neige beaucoup plus complexe et permet, par exemple, de modéliser la formation de congères. Récemment, Saltvik et coll. (SEN08) ont parallélisé la résolution des équations de Navier-Stokes afin de réduire la durée des temps de calcul.

Muraoka et coll. (MC00) ont développé une approche plus complète et prennent en compte les échanges thermiques par conduction. La distribution est calculée en simulant la chute de la neige à l'aide d'un champ de vortex. L'adhésion de la neige sur la surface est également simulée et s'effectue en considérant le degré d'humidité des flocons. La fonte de la neige est simulée en calculant la conduction de la chaleur. Cependant, leur article présente très peu de détails d'implémentation. De plus, lors de la fonte de la neige, seuls les échanges de chaleur par conduction sont pris en compte. Ils négligent totalement les échanges par radiations et convections qui ont un impact bien plus important sur le manteau neigeux.

1.4.4.2 Déplacement de surfaces



D'autres approches consistent à caractériser la hauteur de l'accumulation de la neige en évaluant l'accessibilité locale et l'occlusion. La surface réceptrice est ensuite déplacée verticalement en fonction de la quantité de neige accumulée. Des implémentations sur GPU ont été proposées pour accélérer les temps de calcul et prendre en compte de très grands terrains. Les régions d'accumulation de neige sont définies en calculant l'occlusion ambiante et celle-ci est dissipée en fonction de l'illumination (FB07). Ohlsson et coll. (OS04) utilisent le tampon de profondeur pour quantifier la neige qu'une surface est susceptible de recevoir. Le rendu de la neige est réalisé à l'aide d'un pixel shader spécifique. La géométrie n'est pas générée.

L'utilisation de cartes d'étendue de hauteur introduites pour l'animation interactive de matériaux granuleux par Sumner et coll. (SOH99), se sont avérées efficaces pour modéliser des objets couverts de neige. Haglund et coll. (HAH02) ont utilisé cette représentation pour modéliser le manteau neigeux en temps réel. Cependant la simplicité de leur modèle offre des résultats peu réalistes et le rendu n'est pas satisfaisant. Une amélioration majeure de ce type d'approche a été proposée par Festenberg et coll. (FG09). Ils ont développé un modèle statistique phénoménologique qui a été construit à partir d'observations réelles et qui permet d'accumuler la neige sur des objets.

1.4.4.3 Formation de glace

Kim et coll. sont les seuls à avoir proposé des méthodes permettant de simuler la formation de la glace. La croissance des cristaux de glace et la formation de stalactites de glace est un phénomène complexe impliquant des changements d'état et un processus de solidification. Dans (KL03), Kim et coll. ont proposé une méthode en deux dimensions permettant de simuler le givre sur des objets à l'aide d'une méthode de champ de phase. Cette méthode a ensuite été améliorée en combinant un algorithme d'agrégation à diffusion limitée procédural, une simulation de fluide et une méthode de champ de phase (KHL04). Ces auteurs ont également simulé la formation de stalactites de glace en trois dimensions en résolvant le problème de Stefan (KAL06). Malgré de très bons résultats, la résolution des équations mises en jeu dans ces simulations est coûteuse en temps de calcul. De plus, la méthode proposée pour la formation des stalactites ne peut pas s'appliquer à de très grandes scènes. En effet, la discrétisation précise au niveau de l'interface air-glace nécessaire pour capturer correctement le phénomène de cristallisation des gouttes d'eau est coûteuse en mémoire.

1.4.4.4 Conclusion

Les techniques de simulation consistent principalement à accumuler de la neige sur le sol et les objets. Le dépôt de neige est réalisé en simulant la trajectoire des flocons, leur transport par le vent ou en évaluant la capacité des surfaces à recevoir de la neige. Bien que ces approches permettent d'obtenir de bons résultats, la fonte de la neige est rarement prise en considération. Lorsque cela est le cas, elle s'effectue de façon empirique et rarement en simulant seulement les échanges de chaleur par conduction. Cependant, l'ensemble des échanges thermiques a un impact majeur sur le manteau neigeux. Il est donc primordial de les prendre en compte dans leur intégralité. De plus, en fonction de la configuration géométrique du paysage ainsi que des conditions climatiques, il est difficile d'évaluer quels types d'échanges thermiques auront le plus d'impact.

1.4.5 Écosystèmes

L'objectif des simulations d'écosystèmes est d'obtenir une distribution de plantes réalistes en simulant la compétition pour les ressources telles que l'eau, la lumière ou l'espace. La majorité des méthodes qui ont été développées en informatique graphique prennent en compte la compétition pour les ressources du sol. La quantité de ressources est représentée soit par des cartes de densité, soit par des cartes de présence d'eau.

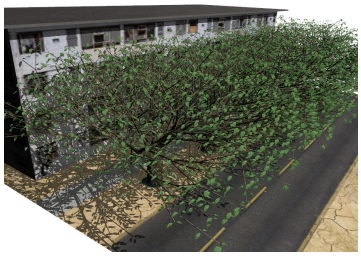
En 1998, Deussen et coll. (DHL+98) ont été les premiers à aborder ce concept. Le principe est que chaque plante est représentée par un disque définissant sa zone d'influence qui grandit avec l'âge. Au début de la simulation, des disques de tailles différentes sont distribués aléatoirement. Lorsque deux disques s'intersectent, la plante qui possède le plus petit disque meurt et est supprimée de la simulation. Ils simulent ainsi le phénomène d'éclaircissage qui correspond à l'étouffement et à la mort des jeunes plantes dominées par les plus grandes lors de la compétition pour les ressources. Les plantes qui atteignent l'âge adulte sont également supprimées. L'instantiation des plantes est réalisée en utilisant le système XFrog (DL97). Chaque plante est représentée individuellement par une instance. Ceci engendre une complexité importante de la scène. Pour la simplifier, les instances d'apparence similaire sont identifiées et remplacées par une seule instance. Malgré de très beaux résultats, la distribution des plantes est beaucoup trop uniforme. Une des raisons est que le processus de croissance des plantes ne prend pas en considération les plantes environnantes et la simulation prend seulement en compte le phénomène d'éclaircissage.

Le *multiset L-Système* introduit par Lane et coll. (LP02) est une généralisation des L-Systèmes et permet de définir des règles de croissance pour un groupe de plantes. Leur système est ainsi capable de simuler le regroupement des plantes de même espèce.

De plus, cette méthode prend en compte trois phénomènes importants dans le développement de groupes de plantes. Le premier phénomène est l'éclaircissement, le second est la succession et le troisième est la propagation. La succession correspond au remplacement d'une plante qui meurt par une plus jeune tandis que la propagation consiste à simuler l'apparition de nouvelles plantes dans le voisinage d'une plante mère. Cependant, ce système ne permet pas de simuler la compétition entre les espèces pour les ressources.

La méthode la plus aboutie à ce jour pour simuler la compétition des plantes pour les ressources a été proposée par Alsweis et Deussen (AD05). Dans leur modèle, les plantes peuvent se partager les ressources et la compétition est alors dite symétrique. Cependant, une plante peut prendre l'ascendant sur les autres et monopoliser les ressources, la plante se développe alors au détriment des autres plantes et la compétition est dite asymétrique. Le principe utilisé pour simuler l'écosystème est le même que celui proposé dans (DHL$^+$98). Les zones d'influence peuvent se superposer et le calcul de la surface de superposition permet de déduire quelle quantité de ressources doit être attribuée aux plantes en fonction de leurs rayons respectifs. La prise en compte de l'aspect symétrique et asymétrique de la compétition dans leur simulation génère des écosystèmes très réalistes.

Les méthodes précédentes ne permettent pas de contrôler la distribution ni de simuler l'impact de l'homme. Pour résoudre ce problème, Beneš et coll. (BCS03) ont couplé la simulation d'un écosystème à une simulation d'agents agissant comme des jardiniers. L'écosystème est régi par une version améliorée de l'algorithme de Lane et coll. (LP02). La simulation des agents consiste à leur affecter des tâches telles que arroser, désherber ou encore semer des plantes. Les agents peuvent également communiquer entre eux pour se répartir les tâches. Cette approche leur permet de contrôler la distribution des plantes tout en conservant l'aspect réaliste produit par la simulation d'écosystèmes.



Dans la nature, la compétition entre les espèces ne se limite pas aux ressources du sol. Chaque arbre essaye de se développer pour obtenir le plus d'espace et de lumière possible. Récemment, Beneš et coll. (BAv09) ont proposé une méthode simulant la compétition entre des arbres pour l'espace et la lumière. L'approche utilisée consiste à simuler la croissance de chaque arbre en fonction des autres arbres et des obstacles rencontrés en utilisant un modèle biologique. Les simulations sont réalisées en temps interactif à l'aide d'un algorithme parallèle. Cependant la compétition pour les ressources du sol n'est pas simulée et les arbres ne peuvent ni mourir, ni se propager. Ceci conduit à une distribution uniforme des plantes. De plus, les arbres générés sont tous de la même taille ce qui engendre des résultats peu réalistes.

Les méthodes décrites précédemment sont capables de distribuer des plantes dans un paysage de façon plus ou moins réaliste. La principale raison du manque de réalisme est

que l'ensemble des ressources telles que la lumière, l'eau et l'espace ainsi que les règles de croissance en groupes n'ont jamais été prises en compte simultanément. De plus, les changements climatiques influent énormément sur les écosystèmes. L'élaboration d'un modèle climatique couplé à la simulation d'un écosystème permettrait donc de suivre son évolution au fil des saisons.

1.5 Synthèse

Tout au long de ce chapitre, nous avons mis en évidence les avantages et les inconvénients de chacune des méthodes existantes pour générer du contenu graphique.

Les approches à base d'esquisses et d'exemples offrent, en général, beaucoup de contrôle à l'utilisateur tout en limitant le nombre d'interactions qu'il doit effectuer. De plus, aucune connaissance en modélisation n'est requise. Néanmoins, les systèmes ne traitent qu'une seule catégorie d'objets. Par conséquent, dès lors que l'utilisateur souhaite modéliser un autre type d'objet, il doit, soit changer de système, soit revenir à un outil de modélisation qu'il ne maîtrise pas forcément. Par ailleurs, une fois que l'objet a été généré, il est impossible de créer rapidement des variétés de celui-ci : le processus doit être effectué de nouveau du début à la fin ce qui peut devenir rapidement fastidieux si l'utilisateur souhaite concevoir de nombreux objets. D'autre part, le nombre de polygones générés peut être très important. Ces objets ne sont donc pas exploitables dès lors qu'ils doivent être visualisés en masse et en temps réel. Pour finir, chaque objet modélisé doit être stocké en mémoire. Dans le contexte de la création de jeux vidéo, l'espace mémoire des consoles est assez faible. Il n'est donc pas possible de créer de vastes scènes avec des milliers d'objets tous légèrement différents.

Les méthodes procédurales ont l'avantage de pouvoir générer beaucoup de contenu à partir de très peu d'informations. De plus, la création de variétés d'objets se fait simplement en modifiant les différents paramètres. Toutefois, le nombre de paramètres à manipuler est souvent très grand, et en changeant la valeur d'un seul paramètre, la forme globale de l'objet peut totalement changer et il devient difficile à l'utilisateur d'obtenir rapidement l'objet désiré. L'utilisateur doit également se familiariser avec le système avant de pouvoir correctement l'utiliser. Dans certains cas, des notions de programmation peuvent même être nécessaires. Finalement, bien qu'il soit très facile d'obtenir de nombreux objets une fois la maîtrise de l'outil acquise, les objets doivent tous être stockés en mémoire.

Pour finir, les simulations sont capables de produire du contenu graphique dynamique. Les objets et les paysages peuvent changer d'apparence avec le temps. Ceci accroît considérablement le réalisme visuel des scènes. D'autre part, la simulation des phénomènes à l'aide d'équations physiques traduit une réalité physique. Cependant, ces méthodes sont très coûteuses en temps de calcul et n'offrent, en général, quasiment aucun contrôle à l'utilisateur. Le résultat est exclusivement guidé par les algorithmes et les équations utilisés. Bien que beaucoup de méthodes utilisent des facteurs envi-

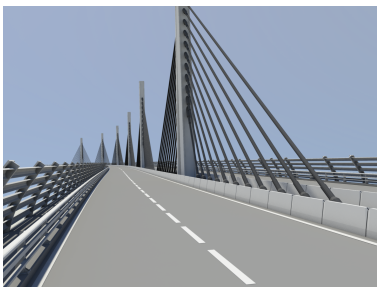
ronnementaux, aucune ne prend en compte les variations des conditions climatiques qui finalement définissent ces facteurs et permettraient de contrôler la simulation avec des paramètres plus intuitifs. Notamment, les méthodes simulant l'accumulation de la neige ne permettent pas de la faire fondre et de suivre l'évolution du manteau de façon réaliste. Les raisons sont, d'une part, que les échanges thermiques ne sont pas simulés avec précision, et d'autre part, que les variations des conditions climatiques ne sont pas prises en compte.

Dans les chapitres suivants, nous avons proposé des approches permettant de lever un certain nombre de limitations énoncées précédemment.

Chapitre 2

Génération de variété d'objets





Sommaire

2.1 Introduction

Dans le secteur du jeu vidéo, les artistes doivent faire face à des contraintes importantes de visualisation en temps réel et de faible capacité mémoire. Pour surmonter ces limitations, ils modélisent manuellement un nombre limité d'objets en faisant très attention au nombre de polygones total. Les scènes sont finalement peuplées avec ce nombre limité d'objets en prenant grand soin d'éviter au maximum les répétitions pouvant dégrader le réalisme des scènes. Ce travail est très fastidieux.

La géométrie et la texture des objets contribuent de façon importante à leur aspect visuel. Il est donc important d'introduire des variations au niveau de la géométrie et de la texture pour que le réalisme des scènes soit préservé. Néanmoins, le stockage de ces variations peut devenir rapidement problématique. Les méthodes proposées doivent donc avoir une représentation suffisamment compacte pour pouvoir être utilisable dans un contexte tel que celui du jeu vidéo.

Dans ce chapitre, nous proposons deux méthodes pour générer rapidement des variétés d'objets. Les approches utilisées permettent de s'adapter aux contraintes de visualisation en temps réel et de faible capacité de stockage.

La première méthode proposée est une approche inverse. Elle consiste à générer des milliers de variétés d'objets à partir d'un objet existant texturé, en modifiant la géométrie et la texture. Le faible surcoût mémoire introduit rend cette méthode très adaptée au contexte du jeu vidéo. De plus, son aspect générique permet de traiter tout type d'objets et ce, quel que soit la complexité géométrique des modèles. Finalement, le temps nécessaire pour créer les variations est très faible. Par conséquent, les temps de production sont considérablement factorisés. Nous avons utilisé cette approche pour générer des variétés d'arbres, de bâtiments, de rochers et de colonnes.

La seconde méthode est une approche directe. La géométrie des objets est directement générée, à la volée, à l'aide d'algorithmes procéduraux. Ceci est donc très peu coûteux en mémoire. La complexité et la forme sont contrôlées à l'aide de paramètres et de règles qui définissent le modèle paramétré commun à l'ensemble des variations d'un même type d'objet. Cette approche nous permet de générer des objets tels que des routes, des ponts et des tunnels, qui s'adaptent automatiquement aux contraintes d'une trajectoire et du relief d'un paysage.

Dans la suite de ce chapitre, nous détaillons les deux méthodes que nous avons proposées pour générer des variétés d'objets.

2.2 Génération d'objets par fragments

La création de paysages de synthèse est un problème complexe. La difficulté est de générer les différents éléments permettant de composer le paysage. De plus, dans la nature, chaque objet est différent et leur nombre est très élevé. Cette variété doit être

prise en compte pour que les scènes de synthèse soient le plus réalistes possibles.

Les méthodes procédurales se sont avérées très efficaces pour générer des variations de terrains (MKM89), de plantes (PHM93, PHL+09), de bâtiments et de villes (MWH+06, WMWG09). Cependant, le nombre de paramètres rend ces techniques difficiles à contrôler. De plus, les algorithmes développés sont conçus pour générer un type d'objets particuliers.

Les approches utilisant des exemples tels que des images (TZW+07) ou des objets géométriques (Mer07) permettent de générer de la variété. Toutefois, ces méthodes ont tendance à produire des modèles de très hautes résolutions comportant de nombreux polygones. Ceci devient problématique lorsque les scènes doivent être affichées en temps réel.

Un problème majeur de toutes ces méthodes est que chaque objet généré doit être stocké. Dès lors que le nombre d'objets devient important, des problèmes mémoire surgissent. La technique généralement utilisée pour peupler de très grandes scènes et surmonter les problèmes de mémoire est l'instanciation. Cette approche consiste à répéter plusieurs fois, dans une même scène, un même objet en appliquant des transformations affines. Néanmoins, des répétitions apparaissent et révèlent l'aspect synthétique des paysages.

Dans cette section, nous proposons une approche inverse de génération de contenu graphique permettant de créer de nombreuses variations d'un objet pour un faible surcoût mémoire. En particulier, nous nous focalisons sur les maillages de faible résolution qui sont utilisés pour les jeux vidéos. Les contraintes d'affichage et de mémoire des consoles obligent les graphistes à créer des objets comportant très peu de polygones. En effet, les architectures Xbox 360 et Playstation® 3 possèdent respectivement 512Mo et 256Mo de mémoire vidéo.

Notre méthode consiste à générer de façon semi-automatique une très grande variété d'objets à partir d'un objet initial texturé. Les variations sont obtenues en modifiant la géométrie et la texture de l'objet initial. L'approche utilisée est la suivante : le modèle initial est découpé en fragments. La géométrie et la texture des fragments sont ensuite éditées créant ainsi un ensemble de variétés de fragments. Ces fragments sont finalement recombinés pour générer de nombreuses variétés du modèle initial.

2.2.1 Le contexte du jeu vidéo

Le rendu temps réel combiné avec les contraintes de mémoire insuffisante des consoles de jeux vidéo jouent un rôle important dans la manière dont les objets sont créés par les artistes. Cette section présente les problèmes de modélisation et de texturage des objets issus de l'industrie du jeu vidéo.

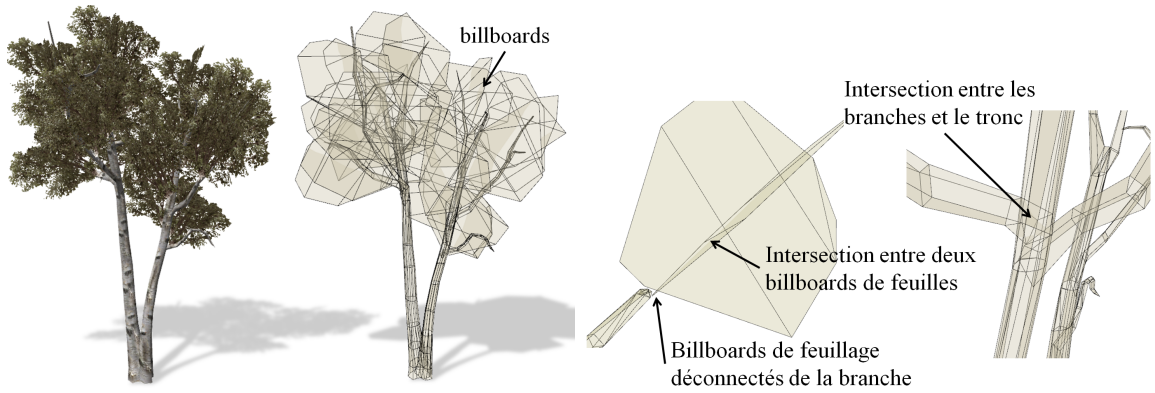


FIGURE 2.1 – Une vue texturée et en fil de fer du modèle de bouleau composé de seulement 745 sommets et 628 faces (gauche). Un gros plan mettant en évidence problèmes de connectivité et d'auto-intersections (droite).

2.2.1.1 Les modèles géométriques basse résolution

Dans les jeux vidéo, des scènes de grande taille doivent être rendues en temps réel avec des centaines d'objets et une capacité mémoire très faible. Par conséquent, les objets conçus par les artistes doivent être créés avec un nombre réduit de sommets et de faces. Par exemple, le modèle de bouleau dans la figure 2.1 (gauche) est composé de seulement 745 sommets et 628 faces incluant de nombreux *billboards* représentant le feuillage.

Afin d'atteindre ce faible nombre de polygones, les artistes modélisent leurs objets en utilisant des ensembles de maillage de faible résolution déconnectés les uns des autres et pouvant s'intersecter entre eux (Figure 2.1, droite). Les artistes passent énormément de temps à modéliser et à texturer les objets avec précision pour éviter tout artéfacts visuels et obtenir de très beaux résultats.

En outre, les objets modélisés de cette façon rendent très difficile la décomposition automatique en sous-objets à l'aide de méthodes de segmentation de maillage traditionnelles (AKM$^+$06).

2.2.1.2 Le pavage de textures

La contrainte de faible capacité mémoire des consoles de jeux vidéo oblige les artistes à utiliser une image de petite taille pour représenter la texture de l'objet. Cette texture est répétée plusieurs fois dans le but de couvrir entièrement l'objet (pavage).

La figure 2.2 illustre une branche texturée à l'aide d'un petit pavé de texture d'écorce ainsi que les coordonnées de texture correspondantes. Bien que la branche sélectionnée soit de petite taille, la texture d'écorce a dû être répétée trois fois pour que le résultat soit satisfaisant.

Les répétitions de texture sont réalisées en utilisant des coordonnées (u, v) au-delà de l'intervalle unité $[0, 1]^2$. Ceci permet à l'artiste de texturer de manière transparente la surface des objets avec une image de petite dimension au prix, bien sûr, de quelques répétitions plus ou moins visibles de motifs.

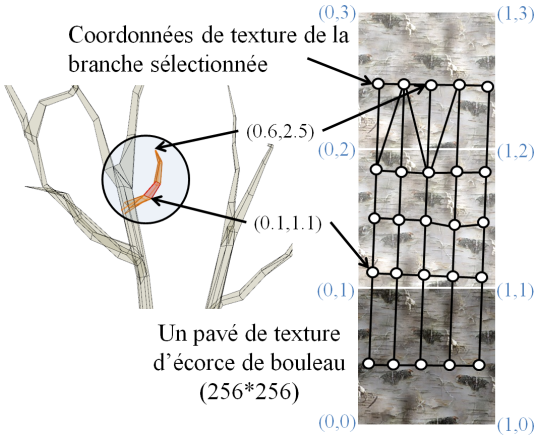


FIGURE 2.2 – Répétition d'un pavé de texture.

Une contrainte supplémentaire dans l'industrie du jeu vidéo est que la quantité d'informations qui peut être attachée aux sommets est fortement limitée. Il est donc crucial de ne pas ajouter d'informations supplémentaires et d'utiliser les coordonnées de texture fournies par les artistes lors de la modification de la texture.

2.2.2 Modification de la géométrie

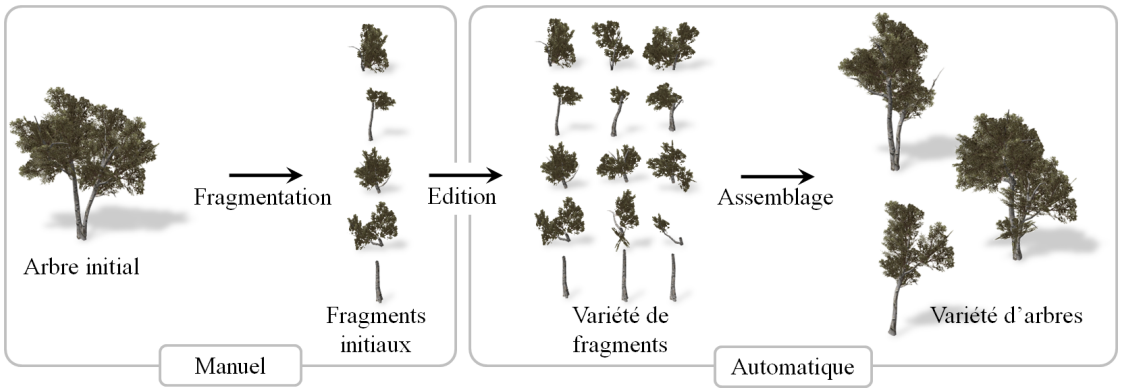


FIGURE 2.3 – Aperçu du processus de génération de variété d'objets par modification de la géométrie.

La génération de variété d'objets par modification de la géométrie s'effectue en trois étapes (Figure 2.3). Tout d'abord, l'artiste décompose un modèle initial texturé en fragments (étape de fragmentation). Ensuite il édite et modifie les fragments obtenus afin

de créer un atlas de variété de fragments (étape d'édition). Finalement, les fragments contenus dans l'atlas sont combinés automatiquement pour créer de nouveaux objets ressemblant à l'objet initial mais cependant tous différents (étape d'assemblage).

Soit O l'objet initial texturé fourni par l'utilisateur. Notre système permet à l'utilisateur de décomposer O en fragments notés F_i tel que :

$$O = \bigcup_{i=1}^{n} F_i$$

avec n le nombre de fragments obtenus après l'étape de fragmentation de l'objet initial. Les fragments F_i sont modifiés pour créer des ensembles de variétés de fragments qui sont stockés dans un atlas. Nous noterons F_i^k la $k^{\text{ième}}$ modification de F_i et $\mathcal{F}_i$ l'ensemble des variétés du fragment i.

L'atlas, noté $\mathcal{F}$, utilisé pour stocker les variétés de fragments $\mathcal{F}_i$ nous permet de générer un ensemble d'objets $\mathcal{V}$ dont l'apparence varie en assemblant les fragments F_i^k. Soit $\tilde{O} \in \mathcal{V}$ un objet généré par notre méthode. Nous définissons $\tilde{O}$ tel que :

$$\tilde{O} = \bigcup_{i=1}^{n} F_i^k$$

Soit $\#\mathcal{F}_i$ le nombre de fragments contenus dans l'ensemble $\mathcal{F}_i$. Le nombre minimal d'objets que nous pouvons générer est défini par :

$$\#\mathcal{V} \geqslant \prod_{i=1}^{n} \#\mathcal{F}_i$$

Le processus de génération de variété d'objets par modification de la géométrie permet la création d'un très grand nombre de variations d'un même objet pour un nombre très faible de fragments. Lorsque l'artiste édite des cycles locaux sur le graphe, le nombre d'objets qui peuvent être générés est supérieur à $\#\mathcal{V}$.

2.2.2.1 Décomposition des objets en fragments

Cette étape consiste à découper l'objet initial texturé O en fragments F_i. Un fragment est un morceau de l'objet initial obtenu par découpage. Notre système permet l'utilisation de méthodes de découpes quelconques. Dans notre implémentation, l'utilisateur a le choix entre deux méthodes :

– Dans le cas général, l'utilisateur dessine un chemin sur la surface de l'objet. Cette méthode ajoute de nouveaux sommets le long des arêtes existantes. Les coordonnées de texture aux nouveaux sommets sont obtenues par interpolation linéaire des coordonnées de texture des sommets de la face coupée (Figure 2.4, gauche).

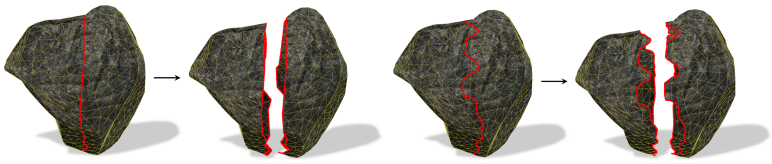


FIGURE 2.4 – Chemin de coupe ajoutant des sommets le long des arêtes (gauche). Chemin de coupe utilisant les sommets existants (droite).

– Du fait de la contrainte temps réel inhérente à la création de jeux vidéo, le nombre de sommets est fortement limité. Dans ce cas, la meilleure solution est de définir un chemin de coupe en utilisant les sommets existants, évitant ainsi l'ajout de nouveaux sommets. Nous avons appliqué cette méthode pour les objets modélisés par les artistes travaillant dans le cadre des jeux vidéo (Figure 2.4, droite).

Après l'opération de découpage, il en résulte deux fragments que nous notons F_i and F_j. Ces fragments possèdent un contour commun correspondant au chemin de coupe défini par l'utilisateur que nous appelons région de connexion et notons R_{ij}. La figure 2.5 illustre le découpage d'un bouleau en deux fragments F_i and F_j et met en évidence la région de connexion R_{ij} commune aux deux fragments.

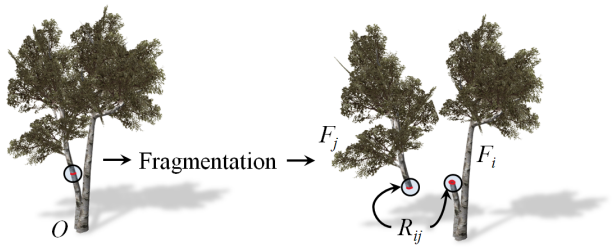


FIGURE 2.5 – Un bouleau découpé en deux fragments et la région de connexion résultante.

Lors de l'étape de fragmentation de l'objet O, nous construisons un graphe $\mathcal{G}$ associé à O et défini tel que :

$$G = (\mathcal{N}, \mathcal{A})$$

$\mathcal{N} = \{N_i\}$ représente l'ensemble des nœuds du graphe $\mathcal{G}$ et $\mathcal{A}$, l'ensemble des arcs reliant deux nœuds entre eux (Figure 2.6). Après l'étape de fragmentation, les nœuds N_i sont initialisés avec un seul fragment F_i. Un arc A_{ij} est créé entre deux nœuds N_i et N_j lorsque leurs fragments F_i et F_j ont une région de connexion R_{ij} en commun.

2.2.2.2 Création de l'atlas de fragments

L'atlas de fragments modifiés est créé par modification de l'objet fragmenté. Chaque modification engendre la création de variété de fragments notée $\mathcal{F}_i = \{F_i^k\}$. L'approche utilisée pour la gestion de l'atlas est similaire à celle décrite par Erikson et coll. (EMI01).

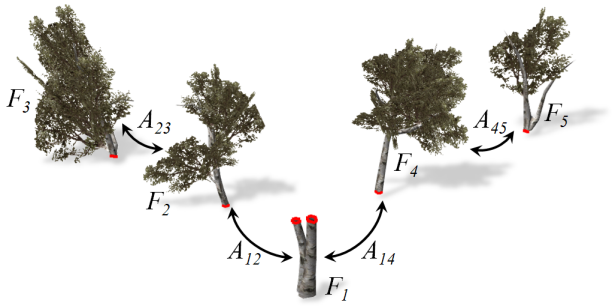


FIGURE 2.6 – Le graphe issu de la décomposition du bouleau en cinq fragments.

Lors de la création de l'atlas, toutes les variétés de fragments de $\mathcal{F}_i$ contenues dans N_i doivent rester connectables à toutes les variétés de fragments de $\mathcal{F}_j$ contenues dans N_j lorsque ces nœuds sont reliés par un arc dans le graphe $\mathcal{G}$. La réciproque doit également être vérifiée afin d'assurer l'assemblage sans aucune discontinuité de la géométrie et de la texture.

Lorsque la région de connexion R_{ij} est affectée par une modification du fragment F_i et/ou F_j, le graphe $\mathcal{G}$ est mis à jour afin de garantir la continuité de la géométrie et de la texture entre les fragments des nœuds N_i et N_j.

En fonction de l'endroit où est effectuée la modification, le graphe est mis à jour de la façon suivante.

Modification locale d'un fragment

Lorsque la modification du fragment F_i n'affecte aucun sommet d'une de ses régions de connexion, un nouveau fragment est ajouté au nœud N_i du graphe $\mathcal{G}$ (Figure 2.7). Ce type de modification, bien que conservant la combinatoire, reste limité en termes de possibilités graphiques. L'artiste utilise principalement les modifications locales pour supprimer des parties d'un fragment tels que le feuillage ou les branches.

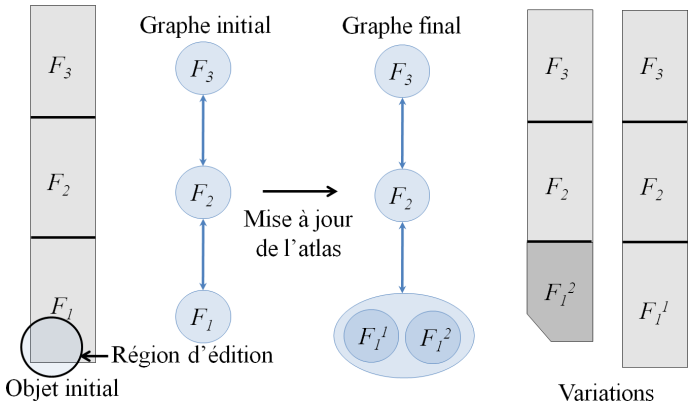


FIGURE 2.7 – Mise à jour de l'atlas lors d'une modification locale d'un fragment.

Déformation préservant la région de connexion

Une façon très efficace pour créer des variations de formes significatives consiste à déformer localement deux fragments F_i et F_j aux alentours de leur région de connexion R_{ij} tout en préservant la forme de R_{ij}. Préserver la forme de la région de connexion R_{ij} garantit la continuité de la géométrie et de la texture entre toutes les variétés de fragments contenues dans les nœuds N_i et N_j. Par ce procédé, nous évitons la division de l'atlas et maximisons la combinatoire. Ce type de modification est particulièrement utile pour la génération de variété d'objets ayant une forme arborescente tels que les arbres ou les lampadaires.

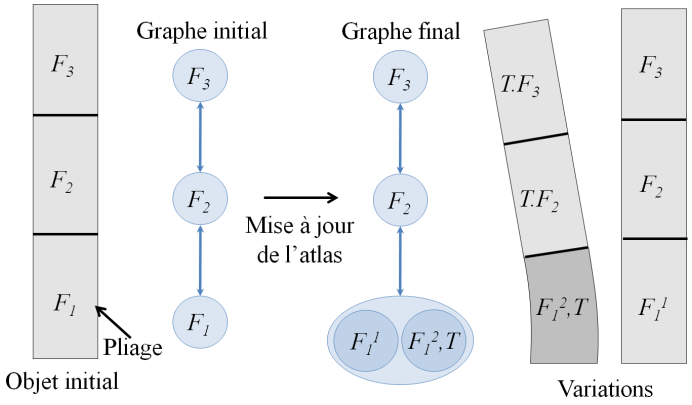


FIGURE 2.8 – Mise à jour de l'atlas lors d'une modification par transformation affine de la région de connexion.

Lorsqu'une transformation affine telle qu'une translation, rotation ou mise à l'échelle est appliquée à l'ensemble des sommets constituant la région de connexion R_{ij} du fragment F_i, nous stockons la transformation dans un repère associé à R_{ij}. Ce repère permettra la connexion sans discontinuité de la géométrie et de la texture des fragments les uns aux autres lors de l'étape d'assemblage. Par exemple, dans la Figure 2.8, si le fragment F_1^2 est sélectionné, nous devons appliquer la transformation stockée dans le repère T à F_2 et F_3.

Modification de la région de connexion

Dans le cas général, lorsque la modification déforme la région de connexion R_{ij} des fragments F_i et F_j, le graphe $\mathcal{G}$ est dupliqué pour créer un nouveau graphe $\mathcal{G}'$ dans lequel toutes les autres variétés de fragments de $\mathcal{F}_i$ et de $\mathcal{F}_j$ sont supprimées. Ceci permet d'étendre les possibilités en divisant l'atlas (Figure 2.9).

Dans certains cas, le graphe associé à l'objet peut être entièrement cyclique sans nœuds terminaux (Figure 2.10). Appliquer une transformation affine affectant la région de connexion d'un objet entièrement cyclique rend l'assemblage impossible et provoque des fissures dans l'objet. Par conséquent, nous interdisons ce type de modification

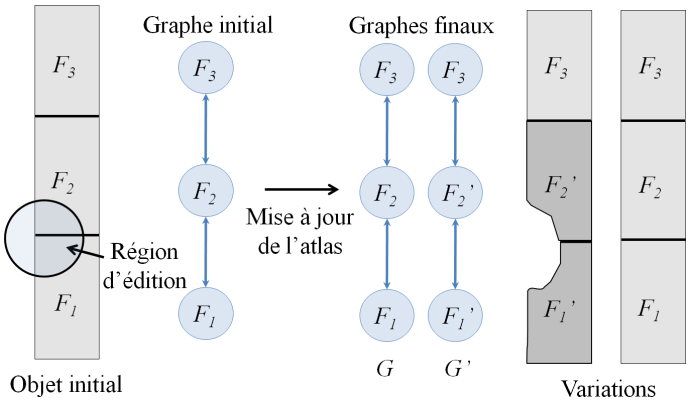


FIGURE 2.9 – Mise à jour de l'atlas lors d'une modification affectant la région de connexion.

dans notre implémentation et garantissons ainsi les propriétés d'assemblage de tous les fragments.

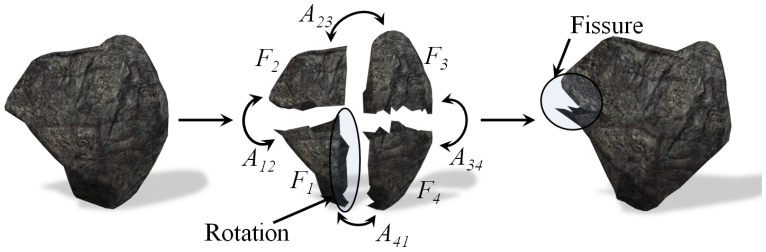


FIGURE 2.10 – Graphe entièrement cyclique issu de la décomposition d'un rocher en quatre et incidence d'une modification ayant engendré une rotation d'une région de connexion.

Régions de connexion identiques

Après l'étape de fragmentation, notre algorithme vérifie s'il existe des régions de connexion identiques entre fragments ou aux extrémités d'un même fragment. L'utilisateur a ensuite la possibilité d'éditer le graphe initial pour inclure des cycles locaux entre les nœuds dont les fragments possèdent des régions de connexion identiques. Par conséquent, certains fragments peuvent être répétés plusieurs fois offrant des possibilités similaires à celles fournies par les grammaires de formes (MWH+06) (Figures 2.11 et 2.12).

Afin de contrôler le nombre de répétitions possibles d'un fragment, un nombre maximum de répétitions est associé à chaque cycle local du graphe par l'utilisateur. Ceci permet de prévenir une éventuelle boucle infinie durant l'assemblage. L'ajout de cycles locaux permet d'accroître considérablement le nombre de variété d'objets qui peut être généré.

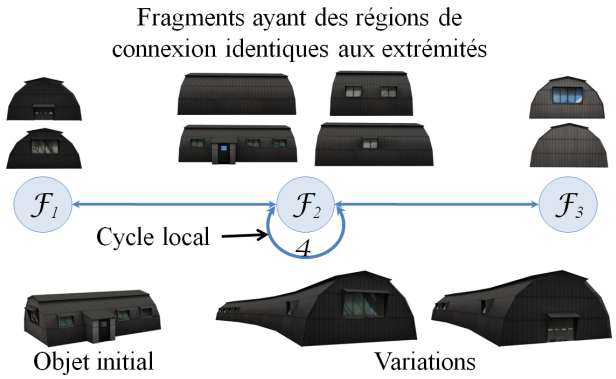


FIGURE 2.11 – Fragments ayant des régions de connexion identiques à leurs extrémités ainsi que le graphe associé.

Dans notre implémentation, l'édition de cycles locaux est restreinte aux graphes de type arborescents. Par conséquent, les transformations affines des régions de connexion des fragments situés dans un cycle local peuvent être effectuées sans aucun risque d'apparition de fissures (Figure 2.11).

Bien qu'un nœud terminal puisse avoir une région de connexion qui soit identique avec celle d'un autre fragment, la répétition d'un nœud terminal ($\mathcal{F}_1$ et $\mathcal{F}_3$ dans la Figure 2.11 et $\mathcal{F}_1$ et $\mathcal{F}_4$ dans la Figure 2.12) par édition d'un cycle l'impliquant, n'a pas de sens. Par conséquent, dans notre implémentation, nous interdisons l'édition de cycles locaux lorsqu'un nœud terminal est concerné.

La figure 2.11 illustre deux hangars générés à l'aide d'un cycle et dont un fragment a subi une rotation d'une région de connexion dans le cycle.

La figure 2.12 illustre trois variétés de bâtiments obtenus par l'édition de cycles plus complexes.

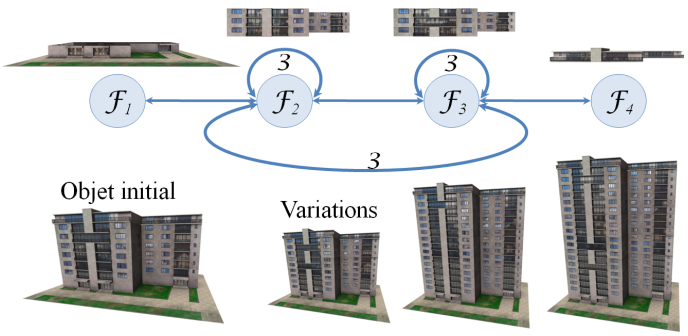


FIGURE 2.12 – Un graphe complexe composé de plusieurs cycles locaux édités par l'utilisateur ainsi qu'un ensemble de bâtiments généré à partir de ce graphe.

2.2.2.3 Assemblage

L'étape d'assemblage permet de générer un nouvel objet final $\tilde{O}$ en parcourant le graphe $\mathcal{G}$ et en sélectionnant un fragment F_i^k à chaque nœud N_i. Les fragments sont assemblés en faisant correspondre les régions de connexion R_{ij} afin de garantir la continuité de la géométrie et de la texture entre eux.

Lors de l'étape d'assemblage, il est nécessaire de recalculer la position de certains fragments lorsqu'une région de connexion a subi une transformation affine. Leur nouvelle position est recalculée en appliquant la transformation contenue dans le repère T de la région de connexion à tous les fragments du sous-arbre. Ceci garantit la continuité de la géométrie.

Dans notre implémentation, le choix des fragments F_i^k constituant l'objet final $\tilde{O}$ peut être fait manuellement par l'utilisateur ou automatiquement à l'aide d'heuristiques. Le choix manuel des fragments permet un meilleur contrôle mais reste cependant limité lorsqu'un grand nombre d'objets doivent être générés comme pour le cas de la création de forêts (Figures 2.34, 2.35).

Afin de permettre l'automatisation du choix des fragments, nous avons implémenté une heuristique permettant d'attribuer une note globale appartenant à l'intervalle $[0, 1]$, à l'objet généré. Cette note dépend d'une combinaison pondérée de deux notes définies par l'artiste et appartenant à l'intervalle $[0, 1]$, attribuées à chaque variété de fragments. Ces notes doivent être choisies de façon à refléter l'importance de la modification subie par le fragment F_i^k par rapport au fragment initial F_i. Un note de 1 indique qu'il n'a pas été modifié tandis qu'un note de 0 indique une forte modification.

Note de géométrie

La première note γ reflète l'importance des modifications appliquées sur la géométrie de l'objet. Elle est choisie directement par l'artiste lors de la modification du fragment. Afin de calculer la note globale de géométrie μ_1 d'un objet donné $\tilde{O}$, nous moyennons simplement les notes de géométrie de chaque fragment constituant l'objet :

$$\mu_1 = \frac{1}{n} \sum_{i=1}^{n} \gamma(F_i^k)$$

Note de transformation de repère

La seconde note ρ reflète l'importance de la transformation affine subie par les régions de connexion et est calculée automatiquement. ρ est une fonction qui évalue le degré de modification du repère de transformation T :

$$\rho(T) = 1 - \theta(T)/\theta_{\max}$$

avec $\theta(T)$ l'angle de rotation de T et $\theta_{\max}$ l'angle maximum autorisé. Les graphes entièrement cycliques ne sont pas concernés par cette note car toute transformation

affine est interdite sur ce type de graphes. Soit N un nœud et N_i ses n fils. La note globale de transformation de repère est calculée récursivement :

$$\mu_2(N) = \frac{1}{n}\sum_{i=1}^{n} \rho(T(N, N_i))\mu_2(N_i)$$

avec $T(N, N_i)$ la transformation de repère entre N et N_i. Intuitivement, une transformation affine appliquée à la racine de l'arbre aura un effet plus important sur la note globale de transformation de repère qu'une transformation affine appliquée à un sous-arbre. En effet, l'impact visuel sera beaucoup plus important.

Note globale

La note globale d'un objet généré $\tilde{O}$ est une combinaison pondérée des deux notes définies précédemment par un poids α :

$$\mu = \alpha\mu_1 + (1 - \alpha)\mu_2$$

α est utilisé pour donner plus ou moins d'importance à la première ou à la seconde note et est choisi dans l'intervalle $[0, 1]$. La figure 2.13 illustre l'évaluation de la note globale de deux bouleaux générés en utilisant $\alpha = 1/2$. L'utilisateur choisit un intervalle pour la note globale μ indiquant ainsi le degré de modification souhaité. Ensuite, notre méthode génère automatiquement les objets ayant une note globale appartenant à l'intervalle choisi. Ceci est très utile pour contrôler automatiquement l'importance des modifications lors de la génération d'un grand nombre d'objets.

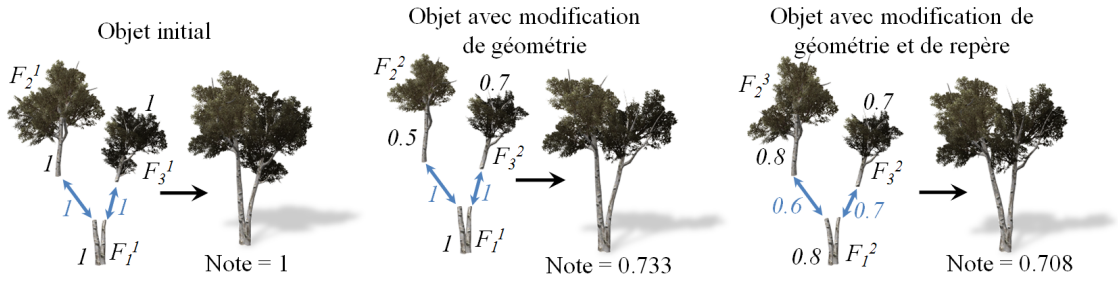


FIGURE 2.13 – Calcul de la note globale ($\alpha = 1/2$) par combinaison des notes de géométrie (noir) et des notes de transformation de repère (bleu).

Résolution des auto-intersections

Les objets initiaux issus de l'industrie du jeu vidéo peuvent s'auto-intersecter (Figure 2.1). De plus, de nouvelles auto-intersections peuvent être introduites lors de l'assemblage des objets finaux. Pour résoudre ce problème, nous détectons tout d'abord les auto-intersections initiales sur l'objet initial et étiquetons les polygones s'intersectant avec le même identifiant. Plusieurs identifiants peuvent être assignés à un même polygone. Lorsqu'un nouvel objet est généré, nous détectons les intersections entre les polygones du nouvel objet et le rejetons lorsque deux polygones qui s'intersectent ne sont pas étiquetés avec un identifiant commun.

2.2.3 Modification de la texture

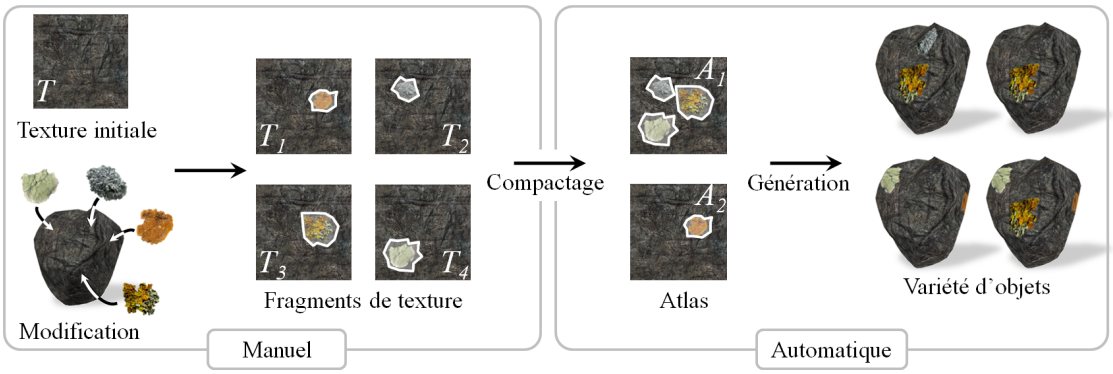


FIGURE 2.14 – Aperçu du processus de génération de variété d'objets par modification de la texture.

Étant donné un objet initial O et sa texture notée T, nous proposons une méthode pour générer un atlas de fragments de texture. Ces fragments sont combinés et assemblés afin de plaquer différentes textures sur l'objet. L'originalité de notre approche est que les coordonnées de texture fournies par l'artiste lors du placage de la texture ne sont pas modifiées. Ceci est crucial dans l'industrie du jeu vidéo car le nombre d'informations telles que les coordonnées de texture qui peuvent être stockées sur chaque sommet est très limité.

La génération de variété d'objets par modification de la texture s'effectue en trois étapes (Figure 2.14). Tout d'abord, la texture initiale T est modifiée par ajout de détails en peignant directement sur l'objet. À chaque modification, un nouveau fragment de texture noté T_k est créé (étape de modification). L'espace mémoire utilisé par les fragments de texture T_k est ensuite réduit par compactage dans un ensemble d'atlas de fragments de texture (étape de compactage). La génération de variété de textures s'effectue en sélectionnant et en combinant les fragments de texture stockés dans les atlas (étape de génération).

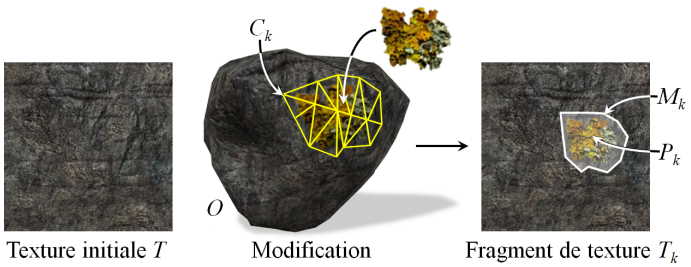


FIGURE 2.15 – Un fragment de texture $T_k = (P_k,\ M_k,\ C_k\ )$ obtenu après l'ajout d'une modification locale de la texture.

2.2.3.1 Modification de la texture

Cette étape consiste à modifier successivement la texture de l'objet. Dans notre système, celle-ci est modifiée en peignant directement les faces de l'objet. Chaque modification est stockée dans un nouveau fragment de texture (Figure 2.15).

La région de texture modifiée définit un fragment de texture noté $T_k = (P_k,\ M_k,\ C_k\ )$. P_k est le masque de pixels définissant la région modifiée dans l'espace de texture. C_k est le groupe de faces référençant les faces modifiées de l'objet. M_k est un masque de pixels plus grand que P_k tel que $P_k \subset M_k$ correspondant à la région occupée par le groupe de faces C_k projeté dans l'espace de texture.

Généralement, une modification locale sur l'objet engendre une modification locale dans l'espace de texture. Cependant, différentes faces de l'objet peuvent se superposer dans celui-ci (Figure 2.16). Dans certains cas, la texture peut se répéter cycliquement au sein d'une même face de l'objet (Figure 2.17).

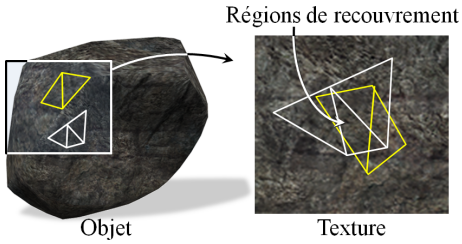


FIGURE 2.16 – Superposition de groupes de faces.

Pour résoudre le problème de superposition dû à la paramétrisation, nous créons de nouveaux fragments de texture dès lors que des modifications locales sur différentes parties de l'objet entrent en conflit par superposition dans l'espace de texture.

Un autre problème majeur est la répétition due au pavage de la texture au sein d'une même face de l'objet (Figure 2.17), provoquant la répétition indésirable de la modification à l'intérieur de la face. Ceci se produit lorsque les coordonnées de texture (u, v) de la face sortent du pavé de texture paramétré dans l'intervalle $[0, 1]^2$. Dans ce cas, nous subdivisons la face en sous-faces à l'endroit où se produit la répétition de texture dans le but de ramener les coordonnées de texture dans l'intervalle unité. La répétition de texture au sein de la face est supprimée, garantissant qu'aucune répétition indésirable de la modification locale ne puisse se produire.

Après n étapes d'édition effectuées par l'utilisateur, n fragments de texture ont été créés. Ceci est coûteux en utilisation mémoire. Par conséquent, il est nécessaire de réduire le nombre de fragments de texture en les compactant dans des atlas de fragments de texture.

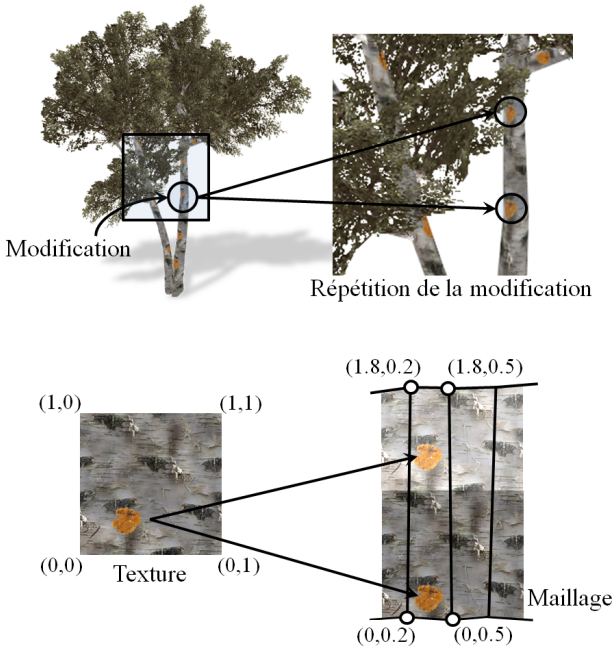


FIGURE 2.17 – Répétition de la modification engendrée par le pavage de la texture.

2.2.3.2 Compactage des fragments de texture

L'étape de compactage rassemble les fragments de texture T_k n'étant pas en conflit dans des atlas de fragments de texture notés A_i (Figure 2.18).

Soit $T_i = (P_i, M_i, C_i)$ et $T_j = (P_j, M_j, C_j)$ deux fragments de texture. T_i et T_j ne sont pas en conflit et peuvent être compactés dans le même atlas si, et seulement si, $P_i \bigcap M_j = \emptyset$ et $P_j \bigcap M_i = \emptyset$. Cette méthode de compactage n'est pas optimale en terme d'occupation mémoire comparée aux méthodes de compactage minimisant le nombre de pixels non utilisés dans les atlas de fragments de texture (LPRM02, SWG+03). En revanche, notre méthode permet de préserver les coordonnées de texture de l'objet initial et d'en éviter la création de nouvelles pour chaque atlas de fragments de texture. Ceci garantit également le placage des fragments de texture sans aucune discontinuité et préserve le placage initialement effectué par l'artiste.

Dans notre implémentation, le compactage s'effectue par une énumération exhaustive des possibilités en vérifiant les intersections entre chaque masque de pixels et chaque masque de faces pour tous les fragments de texture. Nous conservons finalement l'ensemble des atlas de fragments de texture ayant le cardinal le plus petit.

Après compactage, chaque face appartenant au groupe de faces C_k du fragment de texture T_k est étiquetée avec l'identifiant de l'atlas de fragment de texture A_i dans lequel il a été compacté. À ce stade, les masques de pixels et de faces ne sont plus nécessaires et peuvent être supprimés des fragments de texture.

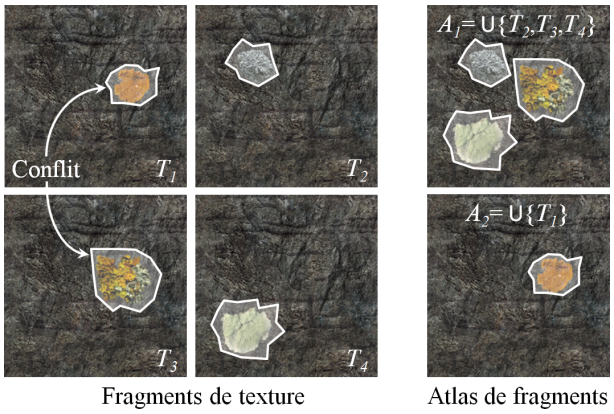


FIGURE 2.18 – Compactage des fragments de texture dans les atlas de fragments de texture.

2.2.3.3 Génération de variété de texture

L'étape de génération de variété de texture consiste à choisir quels seront les fragments de texture qui seront plaqués sur l'objet final (Figure 2.19).

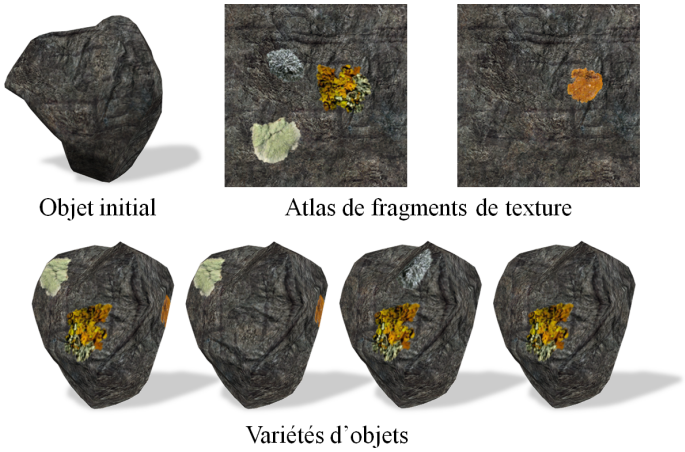


FIGURE 2.19 – Quatre variétés de rocher parmi les $2^4 = 16$ possibilités.

Dans notre implémentation, cette étape est automatisée en associant une probabilité d'apparition à chaque fragment de texture. L'utilisateur a également la possibilité de choisir directement quels seront les fragments de texture utilisés pour un objet spécifique.

Lorsqu'un fragment de texture T_k est sélectionné, nous assignons l'identifiant de l'atlas A_i le contenant, à chacune des faces de l'objet appartenant au groupe de faces C_k. Ces faces sont ensuite verrouillées afin de prévenir la sélection d'un autre fragment de texture partageant les mêmes faces. Ceci évite la superposition et le remplacement partiel d'un fragment de texture par un autre.

Lors de l'affichage de l'objet, nous utilisons l'identifiant associé à chaque face afin de charger l'atlas contenant le fragment de texture sélectionné. La modification de texture correspondante est plaquée sur l'objet en utilisant les coordonnées de texture initialement fournies par l'artiste.

2.3 Modélisation d'objets paramétrés

La création de variété d'objets par fragments peut, dans certains cas, être limitée. Les objets générés ne peuvent pas s'adapter à des contraintes extérieures, ils sont figés dans l'espace. Dans l'industrie du jeu vidéo, lorsqu'un artiste modélise un ouvrage d'art tel qu'un pont ou un tunnel, celui-ci est spécialement conçu pour permettre le franchissement d'une route à un endroit précis d'une rivière ou d'une montagne. La trajectoire initiale de la route peut être obtenue par édition ou en utilisant un algorithme de plus court chemin comme celui qui est proposé par Galin et coll. (GPMG10). Par exemple, pour des raisons de jouabilité, la trajectoire peut être amenée à être modifiée, obligeant le graphiste à modéliser de nouveau son ouvrage d'art. En effet, la largeur de la rivière ou de la montagne à franchir peut avoir diminué ou augmenté. Dans ce contexte, la modélisation manuelle des ouvrages d'art serait beaucoup trop laborieuse. De plus, appliquer une mise à l'échelle ou des déformations pour contraindre le modèle au paysage n'est absolument pas adaptée car les proportions ne seraient pas respectées. Il est donc nécessaire d'avoir une représentation générique des objets capable de s'adapter à un certain nombre de contraintes extérieures.

Une fois la trajectoire définie, nous nous sommes retrouvés confrontés à deux problèmes majeurs :

1. La géométrie de la route, des ponts et des tunnels doit être générée automatiquement et s'adapter aux contraintes de la trajectoire et du relief.
2. Le terrain doit être nivelé par des opérations d'excavation et de remblaiement afin d'accueillir correctement la route.

Le problème de génération de la géométrie est résolu en utilisant des modèles paramétrés. Ils permettent de représenter nos ouvrages d'art de façon générique et de générer la géométrie en fonction des contraintes de la trajectoire et du relief. Le nivellement du terrain est réalisé à l'aide d'un profil paramétré définissant la région sur laquelle reposera la route et une région de mélange permettant de créer les bas-cotés.

2.3.1 Structure hiérarchique

À l'inverse de la génération d'objets par fragments, les modèles paramétrés permettent de générer directement des variétés d'objets. Ces modèles sont représentés par quelques paramètres et la géométrie est générée à la volée. Dans le contexte de la création d'ouvrages d'art, l'intérêt principal d'une telle représentation est que la géométrie générée s'adapte automatiquement aux contraintes du relief du terrain.

La création d'un modèle paramétré consiste à générer la géométrie d'un objet à l'aide de procédures algorithmiques. Cela signifie que la modélisation des objets s'effectue à l'aide d'un langage de programmation. Notre approche est orientée objet et nous définissons nos modèles paramétrés à l'aide de deux types d'objets : les objets terminaux et les objets récursifs.

2.3.1.1 Objets terminaux

Les objets terminaux possèdent une procédure algorithmique permettant de générer la géométrie de l'objet. Ils constituent notre base de données d'éléments qui, par assemblage, permettront de créer des objets plus complexes (Figure 2.20). Nous définissons un objet terminal noté $O_{\mathcal{T}}$ de la façon suivante :

$$O_{\mathcal{T}} = \{\mathcal{P}, \mathcal{F}\}$$

avec $\mathcal{P}$ correspondant à l'ensemble des paramètres de l'objet, divisés en deux catégories : les paramètres de contrôle $\mathcal{C}$ et les paramètres internes $\mathcal{I}$ de l'objet. $\mathcal{F}$ représente l'ensemble des fonctions de calcul permettant de déduire les paramètres internes en fonction des paramètres de contrôle.

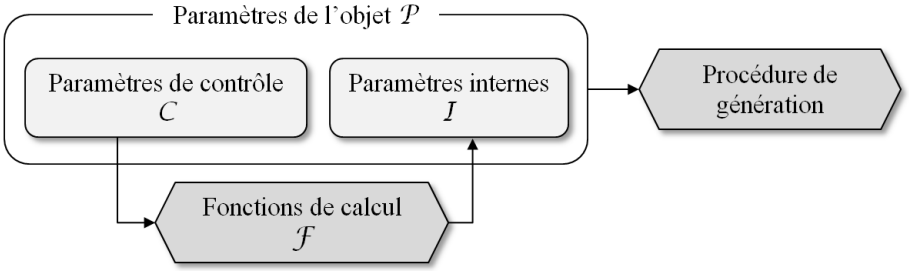


FIGURE 2.20 – Objet terminal.

Les paramètres de contrôle doivent obligatoirement être définis par l'utilisateur lors de l'appel au constructeur de l'objet. Les paramètres internes peuvent également être définis par l'utilisateur mais de manière optionnelle. Dans le cas où l'utilisateur définit ces paramètres, les fonctions de calcul permettant de les déduire automatiquement sont surchargées et ils prennent alors la valeur qui a été fournie. La valeur de chaque paramètre est choisie par l'utilisateur dans un intervalle de validité. Ces intervalles peuvent être continus ou bien discrets (Figure 2.21). Par exemple, la largeur d'une planche est définie dans un intervalle continu et peut mesurer de un à trois mètres, tandis que le rayon nominal d'un écrou est défini dans un intervalle discret dont les valeurs correspondent à des côtes standards : 4mm, 6mm, 10mm, etc...

Les fonctions de calcul permettent de déduire automatiquement les paramètres internes de l'objet à partir des paramètres de contrôle. Ces fonctions sont soit définies de façon *ad-hoc* par l'utilisateur lors de l'implémentation du modèle paramétré, soit correspondent à des fonctions réelles définies par des normes.

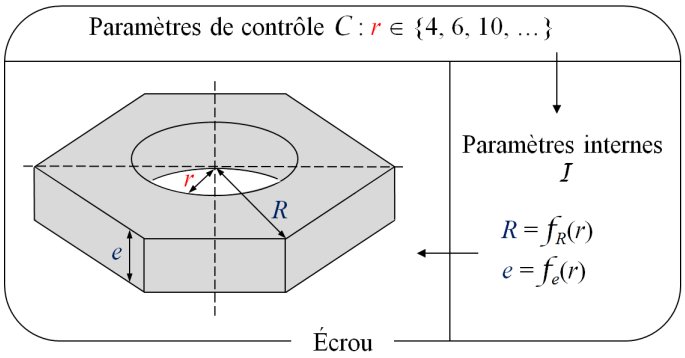


FIGURE 2.21 – Exemple d'objet terminal : écrou usuel ayant pour paramètre de contrôle son rayon nominal r. Le rayon extérieur R est défini par les fonctions $R = 2\,r$ et l'épaisseur e par la fonction $e = 0.4\,r$ définie par les normes ISO 4032, NF EN 24032 et DIN EN 24032.

2.3.1.2 Objets récursifs

À l'inverse des objets terminaux, les objets récursifs ne possèdent pas de procédure algorithmique permettant de générer la géométrie de l'objet. En revanche, ils

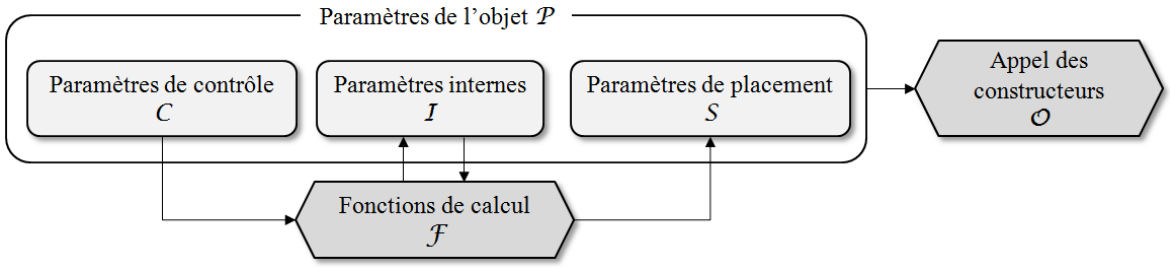


FIGURE 2.22 – Objet récursif.

appellent les constructeurs d'autres objets récursifs et terminaux après avoir calculé les paramètres internes et le positionnement des sous-objets (Figure 2.22). La création d'objets complexes se fait incrémentalement par imbrication d'objets récursifs. Nous définissons un objet récursif $O_{\mathcal{R}}$ de la façon suivante :

$$O_{\mathcal{R}} = \{\mathcal{P}, \mathcal{F}, \mathcal{O}\}$$

avec $\mathcal{P}$ les paramètres de l'objet, $\mathcal{F}$ les fonctions de calcul des paramètres internes comme précédemment définis et $\mathcal{O}$ l'ensemble des appels aux constructeurs des objets terminaux et récursifs composant $O_{\mathcal{R}}$.

Le positionnement des sous-objets s'effectue par l'intermédiaire de paramètres de placement notés $\mathcal{S}$ qui sont définis à l'aide de fonctions de calcul, des paramètres de contrôle $\mathcal{C}$ et des paramètres internes $\mathcal{I}$. Les procédures de placement sont ensuite appelées afin de créer et de positionner les sous-objets.

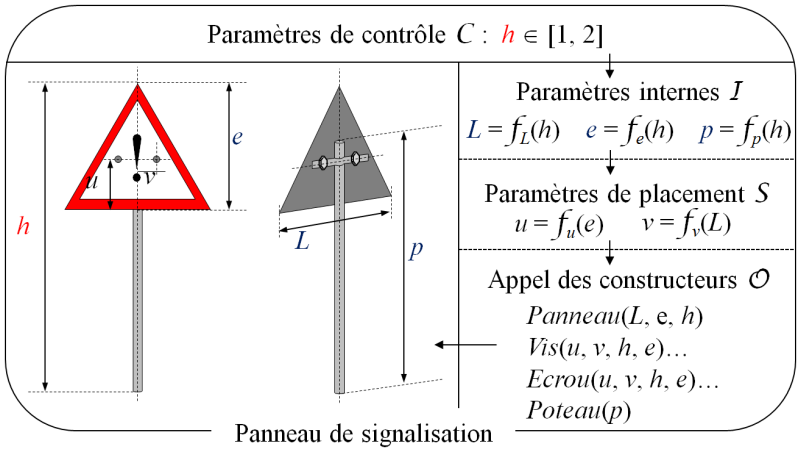


FIGURE 2.23 – Génération d'un panneau de signalisation.

Par exemple, la figure 2.23 illustre un objet récursif permettant de générer un panneau de signalisation qui est défini par sa hauteur totale h prenant sa valeur dans un intervalle continu. Les paramètres u, v et e permettent de positionner les sous-objets en fonction de h. Les procédures *Panneau*, *Vis*, *Ecrou* et *Poteau* sont ensuite appelées pour positionner et générer la géométrie des différents éléments du panneau de signalisation.

2.3.2 Application à la génération procédurale de routes

Nous avons utilisé les modèles paramétrés pour générer des ouvrages d'art. Dans les sections suivantes, nous allons voir comment créer des modèles de routes et de ponts. Les deux modèles de ponts sont choisis automatiquement en fonction du relief du terrain : le pont à piliers est utilisé pour franchir les longs obstacles et le pont Eiffel permet le franchissement de ravins de faible longueur.

2.3.2.1 Création d'un modèle paramétré de routes

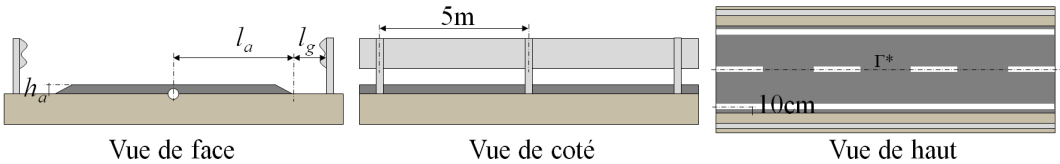


FIGURE 2.24 – Modèle paramétré de routes.

Nous avons créé un modèle paramétré permettant de générer des routes. Les paramètres de contrôle du modèle sont la largeur l_a et la hauteur h_a de l'asphalte, la

trajectoire discrétisée Γ^* et l'espacement entre l'asphalte et les glissières de sécurité l_g (Figure 2.24, gauche).

La forme de la route est créée à l'aide d'un profil dépendant de la largeur l_a et de la hauteur h_a de l'asphalte et qui est extrudé le long de la trajectoire Γ^*.

La forme des glissières de sécurité est obtenue de la même façon et elles sont positionnées à une distance $l_a + l_g$ de la trajectoire. Les poteaux soutenant les glissières sont positionnés tous les cinq mètres (Figure 2.24, milieu).

La signalisation horizontale pointillée est positionnée sur la trajectoire Γ^* à une hauteur h_a et la signalisation horizontale continue est positionnée à 10cm des bords de l'asphalte et à une hauteur h_a (Figure 2.24, droite).

Implémentation. La route est un objet récursif possédant trois procédures de placement et effectuant un appel au constructeur de l'objet terminal *asphalte*. La procédure de placement de la signalisation horizontale pointillée appelle le constructeur de l'objet terminal *ligneDiscontinue* et le positionne sur la trajectoire à une hauteur h_a. Celle de la signalisation horizontale continue effectue deux appels au constructeur de l'objet terminal *ligneContinue* et les positionne respectivement à droite et à gauche de la trajectoire à une distance $l_a - 0.1$ et à une hauteur h_a. Celle des glissières effectue deux appels à l'objet récursif *glissiere* et les positionne à gauche et à droite de la trajectoire à une distance $l_a + l_g$. L'objet glissière fait ensuite appel au constructeur de l'objet terminal *barriere* et à la procédure de placement des poteaux permettant de positionner un poteau tous les cinq mètres en effectuant des appels successifs à l'objet terminal *poteau*.

2.3.2.2 Création d'un modèle paramétré de ponts à piliers

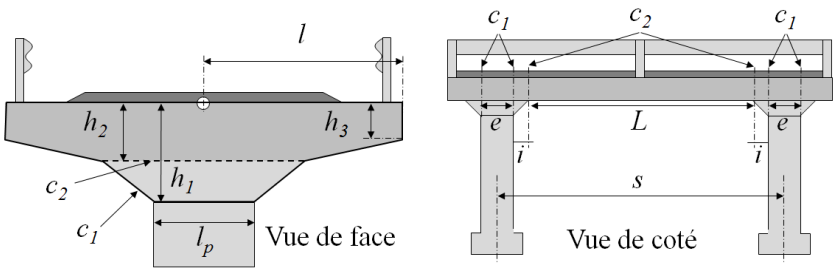


FIGURE 2.25 – Modèle paramétré du pont à piliers.

Le modèle de pont à piliers possède six paramètres de contrôle qui sont sa largeur l, l'espacement entre les piliers s, la largeur l_p, l'épaisseur e de chaque pilier, la trajectoire de la route Γ^* et le terrain. L'intervalle de validité de l est compris entre la largeur totale de la route $l_a + l_g$ et $l_a + l_g + 1$.

Le tablier est généré à l'aide de deux profils c_1 et $c2$ contrôlés par la largeur l du pont et la largeur des piliers l_p et extrudés le long de la trajectoire de la route Γ^*.

Les hauteurs h_1 du profil c_1 et h_2 et h_3 du profil c_2 sont calculées en fonction de la largeur l du pont (Figure 2.25, gauche). Le profil c_1 est extrudé sur une distance e qui correspond à l'épaisseur des piliers afin de les soutenir tandis que le profil c_2 est extrudé sur une distance L et permet de faire la jonction entre deux supports. La jonction entre les profils c_1 et c_2 est réalisée par interpolation linéaire sur une distance i. Le choix du profil à extruder dépend de l'espacement s entre chaque pilier (Figure 2.25, droite).

Un pilier est paramétré par sa hauteur h_p et est généré à l'aide de plusieurs profils qui sont paramétrés par la largeur l du pont et qui permettent de créer le socle ainsi que le pilier porteur du pilier. La hauteur h_p de chaque pilier est calculée en fonction de la différence d'altitude entre la trajectoire de la route Γ^* et la surface du terrain pour une position donnée (Figure 2.26). La géométrie est générée par extrusion des profils le long de la droite verticale de longueur égale à la hauteur h_p de chacun des piliers.

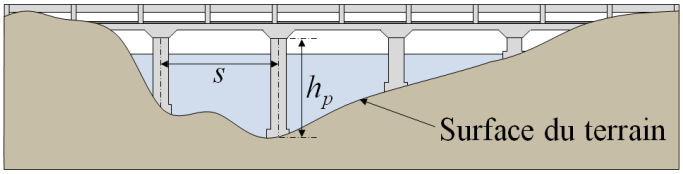


FIGURE 2.26 – Calcul de la hauteur des piliers.

Implémentation. Le pont à piliers est un objet récursif possédant deux procédures de placement. La procédure de placement des profils de contour du tablier calcule la position des profils c_1 et c_2 le long de la trajectoire en fonction de l'espacement s entre les piliers et des distances d'extrusion et d'interpolation e, L et i. Les hauteurs h_1, h_2 et h_3 des profils c_1 et c_2 sont calculées à l'aide de fonctions *ad-hoc* tel que $h_1 = l/2$, $h_2 = l/3$ et $h_3 = l/4$. L'appel au constructeur de l'objet terminal *tablier* permet de générer la géométrie du tablier. La procédure de placement des piliers calcule la position de ceux-ci ainsi que leur hauteur respective puis effectue des appels successifs au constructeur de l'objet terminal *pilier* créant ainsi la géométrie de chaque pilier.

2.3.2.3 Création d'un modèle paramétré de ponts Eiffel

Le modèle de pont Eiffel possède quatre paramètres de contrôle qui sont sa largeur l, la hauteur de l'arche h, l'espacement s entre les renforts situés entre le tablier et l'arche et la trajectoire de la route Γ^*. L'intervalle de validité de l est identique à celui du modèle de pont à piliers.

La tablier est généré à l'aide d'un profil c identique au profil c_2 du pont à piliers qui est extrudé le long de la trajectoire de la route Γ^*. Il est défini à l'aide des mêmes paramètres que ceux du profil c_2 (Figure 2.27).

Les renforts entre le tablier et l'arche sont générés à l'aide d'un profil fixe extrudé le long d'une droite verticale de longueur h_r, égale à la différence de hauteur entre le tablier et l'arche pour une position donnée. Les renforts sont placés à droite et à

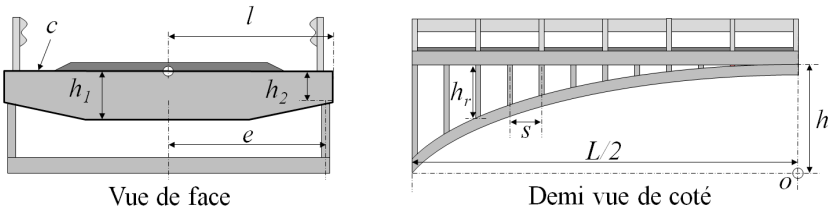


FIGURE 2.27 – Modèle paramétré du pont Eiffel.

gauche, à une distance e de la trajectoire en fonction de la largeur l du pont et espacés d'une distance s.

L'arche est également créée à l'aide d'un profil. La trajectoire d'extrusion est calculée par une ellipse d'origine o qui est paramétrée par la hauteur de l'arche h et la longueur L du pont, calculée automatiquement.

Implémentation. Le pont Eiffel est un objet récursif possédant une procédure de placement et effectuant deux appels aux constructeurs des objets terminaux *tablier* et *arche*. La procédure de placement des renforts calcule la position des renforts en fonction de leur espacement s et de la distance e à la trajectoire. La hauteur h_r est alors calculée et le constructeur de l'objet terminal *renfort* est appelé successivement afin de générer la géométrie de chaque renfort.

2.3.2.4 Ouvrages complexes

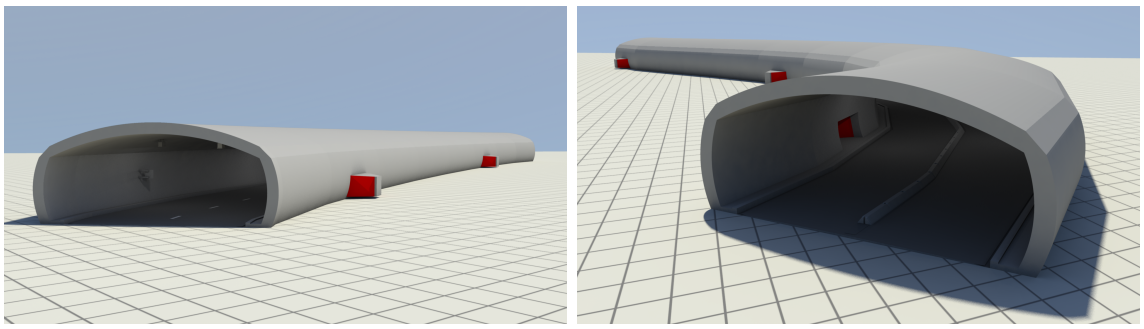


FIGURE 2.28 – Création de deux variations d'un tunnel en utilisant des modèles paramétrés.

En utilisant le principe hiérarchique des modèles paramétrés, nous avons généré des ouvrages beaucoup plus complexes en allant jusqu'à la précision du boulon. La figure 2.28 illustre la création de deux tunnels comportant plusieurs issues de secours et la figure 2.29 illustre deux variations du viaduc de Millau.

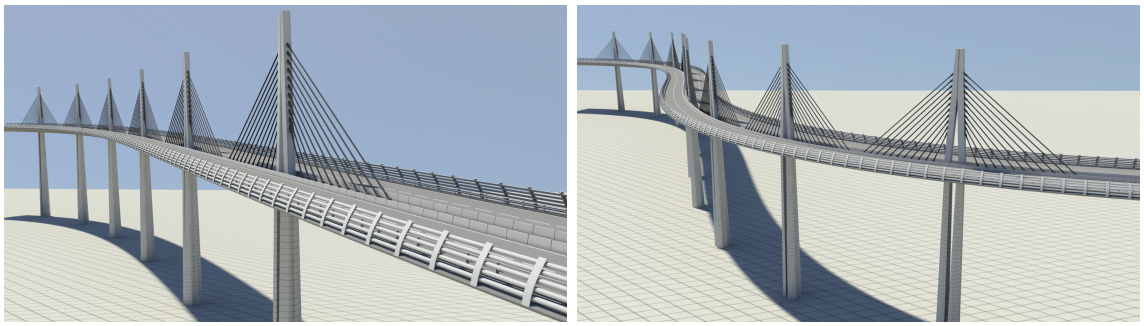


FIGURE 2.29 – Création de deux variations du viaduc de Millau.

2.3.2.5 Génération

La génération de la géométrie s'effectue à la volée en appelant le constructeur des objets récursifs permettant de générer les routes, les ponts et les tunnels. Les éléments constituant chaque modèle sont simplement affichés afin d'avoir un processus de génération rapide en évitant toute opération de raccordement de la géométrie. En effet, effectuer un raccord de géométrie entre les éléments serait très coûteux en calcul et ralentirait considérablement les temps de génération. De plus, la création du raccord modifierait la géométrie en rajoutant des polygones et leur nombre ne serait alors plus contrôlé avec exactitude. Ceci n'est pas souhaitable dans le contexte du jeu vidéo.

Les ouvrages d'art sont principalement composés de matériaux tels que du métal ou du béton. La texture de ces matériaux est facilement représentable à l'aide de textures procédurales 3D. Par conséquent, nous utilisons ce type de texture pour représenter les matériaux de nos différents éléments. De plus, les textures procédurales ont l'avantage d'avoir une représentation compacte.

2.3.3 Intégration des ouvrages dans le paysage

FIGURE 2.30 – Aperçu du processus de génération de la route.

Après avoir défini les modèles paramétrés, nous devons générer les ouvrages le long de la trajectoire et les intégrer dans le paysage. Ce processus d'intégration s'effectue en trois étapes (Figure 2.30). Étant donné un ensemble de points de contrôle définissant

la trajectoire et un terrain initial, nous créons la trajectoire à l'aide de clothoïdes par morceaux. La trajectoire est alors segmentée afin d'identifier les portions de routes, de ponts et de tunnels. Le terrain est ensuite nivelé, les entrées et les sorties des tunnels sont créées et la végétation se trouvant dans le voisinage de la route est supprimée. Finalement, la géométrie de la route, des ponts et des tunnels est générée à l'aide de modèles paramétrés.

2.3.3.1 Segmentation de la trajectoire

La trajectoire ρ de la route est fournie sous forme de points de contrôle $\rho = \{\mathbf{p}_i\}_{i\in[0,n]}$. À partir des points de contrôle $\mathbf{p}_i$, nous créons une courbe Γ à l'aide de clothoïdes par morceaux (WM05). Ce type de courbe est utilisé en voirie pour éviter au conducteur de tourner brutalement le volant et accroître le confort de conduite. Concrètement, ces courbes représentent la trajectoire d'une automobile roulant à vitesse constante et dont le volant est tourné progressivement du fait de la variation linéaire et sans discontinuités de la courbure.

La courbe Γ est alors discrétisée uniformément pour créer une courbe linéaire par morceau noté $\Gamma^* = \{\mathbf{q}_k\}_{k\in[0,m]}$. Pour chaque point $\mathbf{q}_k$ de Γ^*, nous calculons la différence signée $\delta(\mathbf{q}_k)$ entre l'altitude de la projection de $\mathbf{q}_k$ sur la surface du terrain et l'altitude de $\mathbf{q}_k$. La segmentation est alors effectuée en étiquetant chaque point $\mathbf{q}_k$ comme route, pont ou tunnel de la façon suivante : soit $\delta_{\mathcal{T}}$ la hauteur minimum nécessaire à la création d'un tunnel et $\delta_{\mathcal{B}}$ la hauteur minimum à la création d'un pont.

1. Si $\delta(\mathbf{q}_k) \geqslant \delta_{\mathcal{T}}$, étiqueter $\mathbf{q}_k$ comme point de tunnel.
2. Si $-\delta_{\mathcal{B}} < \delta(\mathbf{q}_k) < \delta_{\mathcal{T}}$, étiqueter $\mathbf{q}_k$ comme point de route.
3. Si $\delta(\mathbf{q}_k) \leqslant -\delta_{\mathcal{B}}$, étiqueter $\mathbf{q}_k$ comme point de pont (Figure 2.31).

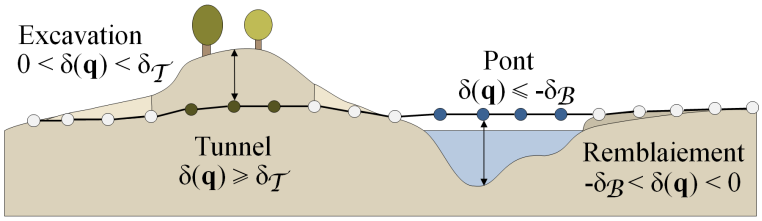


FIGURE 2.31 – Segmentation de la trajectoire de la route.

2.3.3.2 Modification du terrain

La création de clothoïdes par morceaux a pour effet de lisser la trajectoire. Par conséquent, la courbe résultante peut se retrouver au dessus ou en dessous de la surface du terrain. Une fois la courbe segmentée en portions de routes, ponts ou tunnels, nous devons donc modifier la géométrie du terrain en conséquence, afin de coller la route au

terrain. Les portions de route doivent être nivelées et les trous d'entrée et de sortie des tunnels doivent être créés.

Pour les portions de route, nous effectuons les opérations d'excavation et de remblaiement à l'aide d'une courbe de profil notée C définie par l'utilisateur. Le profil est paramétré par la largeur de l'asphalte r et par la largeur de la région de mélange R caractérisant la forme des bas-cotés (Figure 2.32).

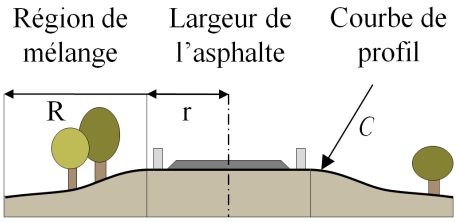


FIGURE 2.32 – Courbe de profil de la route permettant la modification du terrain.

En fonction de la forme de la courbe de profil et de sa largeur totale R, nous modifions la géométrie et supprimons la végétation aux abords de la trajectoire Γ. Nous définissons le voisinage Ω_Γ de Γ par :

$$\Omega_\Gamma = \{\mathbf{p} \,|\, d(\mathbf{p}, \Gamma) < R\}$$

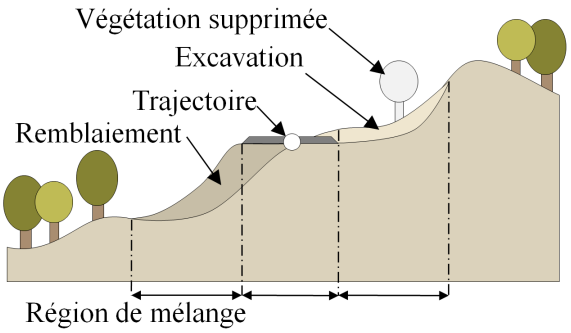


FIGURE 2.33 – Nivellement de la route par excavation, remblaiement et suppression de la végétation aux abords de la trajectoire.

Toutes les instances de végétation situées dans le voisinage Ω_Γ sont tout d'abord supprimées. Afin de palier une résolution trop faible du terrain et de capturer correctement la forme de la courbe de profil C, nous augmentons la résolution du terrain en effectuant une subdivision adaptative du maillage du terrain aux abords de la trajectoire Γ. Pour chaque sommet $\mathbf{v}$ du nouveau maillage du terrain, nous calculons la distance $d(\mathbf{v}, \Gamma)$ à la trajectoire et modifions les altitudes des sommets $\mathbf{v}$ de la façon suivante :

1. Si $d(\mathbf{v}, \Gamma) < r$ alors l'altitude du sommet $\mathbf{v}$ est fixée à l'altitude du point de la courbe de profil le plus proche.

2. Si $r < d(\mathbf{v}, \Gamma) < R$ alors nous calculons la nouvelle altitude du sommet $\mathbf{v}$ en interpolant linéairement la courbe de profil de la route $C(d(v, \Gamma))$ avec l'altitude initiale de $\mathbf{v}$ (Figure 2.33).

2.4 Résultats

Dans cette section, nous présentons les résultats que nous avons obtenus en utilisant notre méthode de génération d'objets par fragments et modèles paramétrés.

2.4.1 Génération d'objets par fragments

FIGURE 2.34 – Des centaines d'arbres différents formant une forêt de Central Park, tous les arbres ont été créés à partir d'un arbre sélectionné parmi les ressources graphiques du jeu vidéo *Alone In The Dark*™. Les rendus sont issus du prototypage de notre méthode dans le moteur du jeu.

Nous avons implémenté et intégré nos algorithmes en tant que modules dans la plateforme de développement *Twilight 2* d'Eden Games et de Widescreen Games dans le cadre du projet GENAC 2. Les ressources graphiques utilisées pour illustrer nos résultats sont issues des jeux vidéo *Alone In The Dark*™ et *Test Drive Unlimited*™ de la société Eden Games.

L'instanciation massive des objets pratiquée dans l'industrie du jeu vidéo créée des répétitions artificielles. Les résultats démontrent que notre méthode est efficace pour les supprimer et augmenter le réalisme en générant un très grand nombre d'objets différents pour un surcoût mémoire faible et avec un nombre de modifications réduit.

La figure 2.34 illustre le prototypage dans le moteur du jeu vidéo *Alone In The Dark*™ de notre méthode.

La table 2.1 présente les caractéristiques des objets initiaux que nous avons utilisés dans nos prochains résultats : nombre de sommets et de faces, taille mémoire (en

Modèle	Bouleau	Roche	Colonne	Maison
#sommets	745	578	233	568
#faces	628	1 152	312	923
Mémoire	328	146	98	186

TABLEAU 2.1 – Complexité et taille mémoire (en ko) des modèles.

ko) du modèle. La table 2.2 reporte le nombre de fragments générés après l'étape de fragmentation, le nombre de variétés par fragment, le nombre total d'objets générés et le surcoût mémoire (en ko et pourcentage) introduit.

Modèle	Bouleau	Roche	Colonne	Maison
#fragments	6	4	5	7
#variétés	3	8	4	3
#objets	729	4 096	1 024	2 187
Mémoire	+177 (54%)	+469 (321%)	+417 (425%)	+90 (48%)

TABLEAU 2.2 – Nombre d'objets générés par notre méthode et surcoût mémoire engendré en ko et pourcentage.

Central Park. Pour le jeu vidéo *Alone In The Dark™* qui se déroule autour de Central Park, 17 espèces d'arbres et 3 variétés par espèce ont été modélisées par les artistes portant le nombre total d'arbres différents à 51. Nous avons appliqué notre méthode sur ces modèles d'arbres et automatiquement généré une très grande variété d'arbres (Figures 2.35, 2.34). Les modifications appliquées aux fragments consistaient essentiellement à supprimer des branches et du feuillage. Ceci explique le faible surcoût mémoire de 54%.

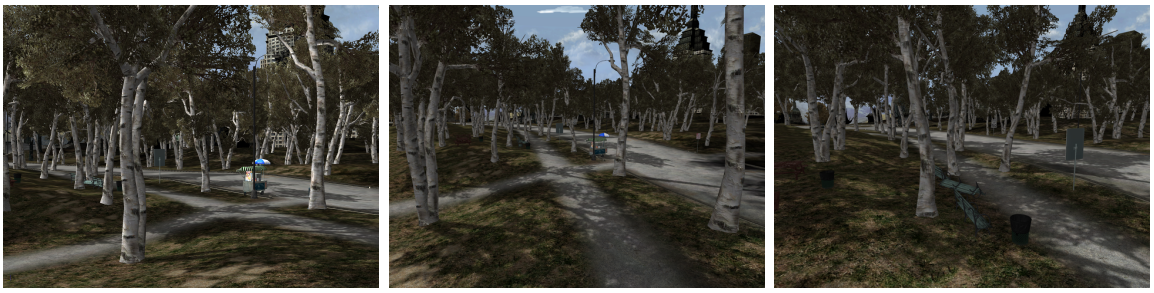


FIGURE 2.35 – Une zone de Central Park remplie de bouleaux générés à partir d'un modèle initial.

Trois heures ont été nécessaires à une personne débutante pour générer les fragments

et créer 729 variétés d'arbres en incluant une très courte période d'entraînement. À titre de comparaison, un artiste passe 8 heures afin de modéliser et texturer un seul arbre. Par conséquent, notre méthode permet de réduire les temps et les coûts de production qui sont cruciaux dans le secteur du loisir numérique.

Quartier résidentiel à Hawaï. Dans le jeu vidéo *Test Drive Unlimited™* qui se déroule sur l'île d'Hawaï, les variétés de maisons dans les quartiers résidentiels ont été conçues manuellement par les artistes qui ont ajouté des éléments de construction tels que des abris de jardin ou des garages à un bloc initial de maison. Dans la figure 2.36 nous illustrons les résultats obtenus en automatisant le procédé mis en place par les artistes en utilisant notre méthode.

FIGURE 2.36 – Quartier résidentiel à Hawaï : chaque maison est différente et a été générée à partir d'un modèle initial.

Éboulement de rochers. La figure 2.37 illustre un éboulement de rochers dans Central Park. Au départ, cet éboulement n'était pas présent dans la scène et notre méthode nous a permis d'en créer un à partir d'un seul rocher. En plus des modifications apportées à la géométrie qui permettent de générer 256 rochers, nous avons effectué 4 modifications de texture compactées dans 2 atlas de fragments de texture portant le nombre de rochers différents à 4 096. Les modifications géométriques ont été plus complexes que dans l'exemple du bouleau, expliquant le surcoût mémoire plus important (321%).

FIGURE 2.37 – Génération automatique de rochers utilisés pour créer un éboulement dans Central Park.

Hall d'entrée du musée métropolitain d'art de New York. La figure 2.38 illustre le hall d'entrée du musée métropolitain d'art de New York en ruines. Avant d'utiliser notre méthode sur les colonnes, seulement quelques-unes d'entres elles étaient endommagées. Les colonnes ont été coupées horizontalement en 5 morceaux différents qui ont été édités afin d'ajouter davantage de cassures et de fissures. De nouveaux sommets ont été insérés et plusieurs modifications de la texture ont été réalisées. C'est pour cette raison que le surcoût mémoire est plus important que dans les exemples précédents (425%).

FIGURE 2.38 – Le hall d'entrée en ruines du musée métropolitain d'art (MET) situé à New York.

2.4.2 Génération d'ouvrages d'art

Notre méthode de génération de modèles paramétrés a été implémentée en C++ sur une machine équipée d'un processeur Intel® Core™2 Duo 3GHz avec 3GB de RAM. Nous avons utilisé ces modèles pour générer des routes, des ponts et des tunnels dans la plateforme de recherche de plus court chemin anisotrope (GPMG10) afin de résoudre les problèmes d'instanciation.

Pont Eiffel. La figure 2.39 illustre l'intégration d'une route et d'un pont dans un paysage. Le pont Eiffel a été généré automatiquement pour permettre le franchissement de la rivière de faible largeur au dessus d'un grand dénivelé.

FIGURE 2.39 – Franchissement d'un canyon à l'aide d'un pont Eiffel.

Pont à piliers et tunnel. La figure 2.40 illustre la création d'un pont à piliers permettant le franchissement d'une rivière d'une grande largeur ainsi qu'un tunnel pour permettre la traversée d'une montagne.

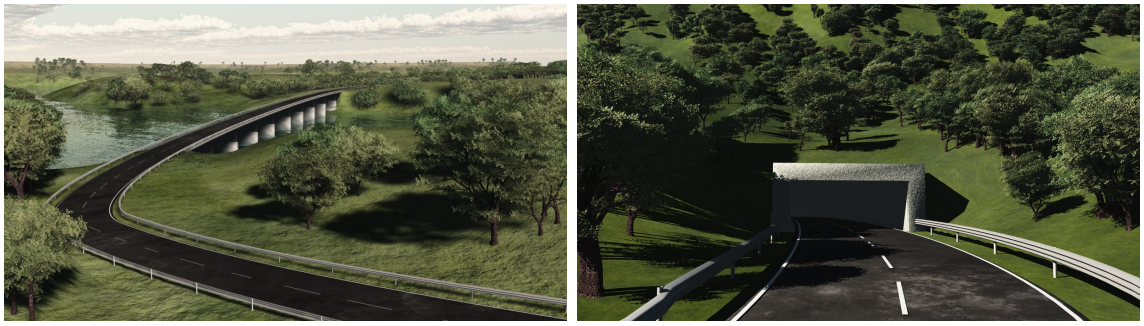


FIGURE 2.40 – Traversée d'une rivière et d'une montagne à l'aide d'un pont à piliers et d'un tunnel.

Tunnel complexe. Les modèles paramétrés ont été utilisés pour générer un tunnel complexe comportant de nombreux éléments tels que des issues de secours, des lampes et des panneaux. La figure 2.41 illustre deux tunnels générés à partir du même modèle en modifiant la trajectoire ainsi que le positionnement des différents éléments.

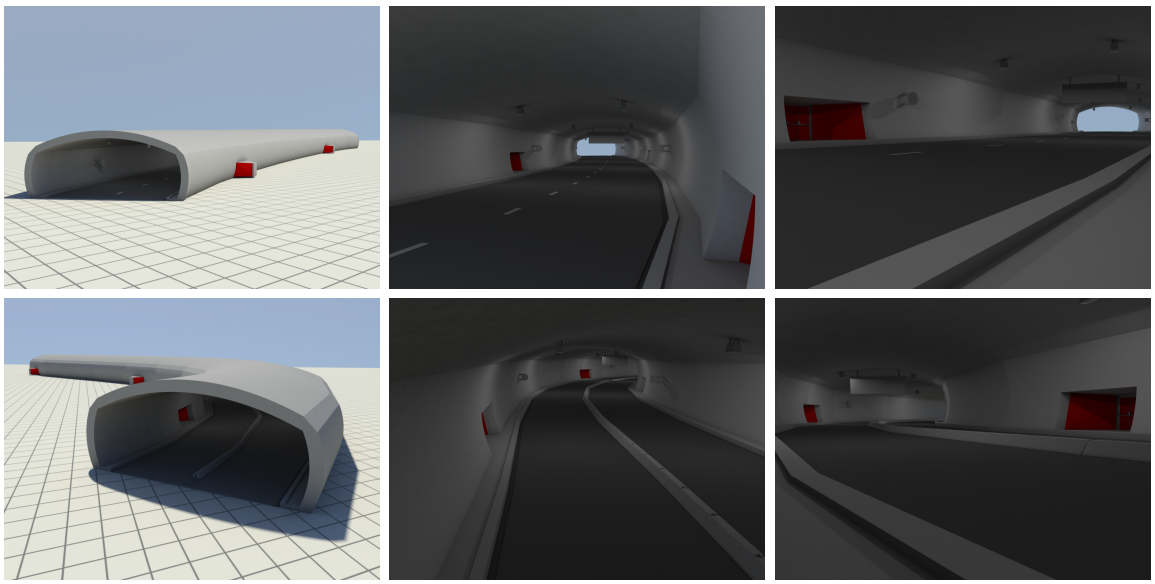


FIGURE 2.41 – Création de deux variations du tunnel.

Viaduc de Millau. Le viaduc de Millau est un pont à haubans autoroutier franchissant la vallée du Tarn, dans le département de l'Aveyron, en France. Sa longueur est de 2 460 mètres pour une hauteur de 270 mètres (Figure 2.42). Les modèles paramétrés nous ont permis de générer ce viaduc avec une très grande précision. La figure 2.43

FIGURE 2.42 – Photographie du viaduc de Millau.

illustre deux versions du viaduc de Millau s'adaptant à des trajectoires différentes. Le nombre de piliers est automatiquement calculé en fonction de la longueur du pont.

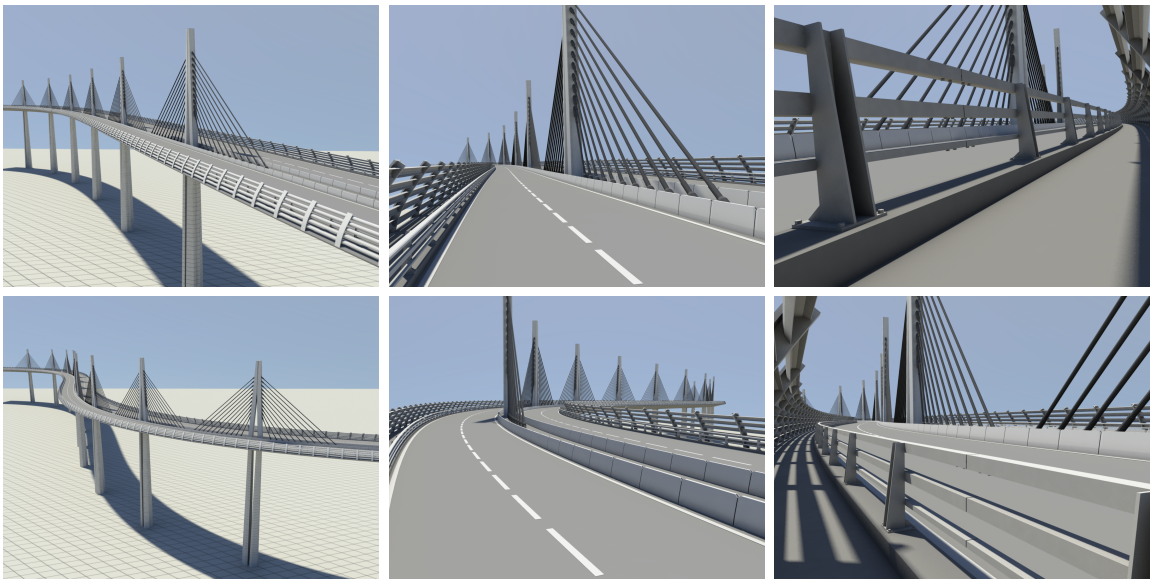


FIGURE 2.43 – Deux variations du viaduc de Millau obtenues à l'aide du même modèle paramétré.

2.5 Conclusion

Dans ce chapitre, nous avons proposé deux approches complémentaires pour générer rapidement et en grande quantité du contenu graphique.

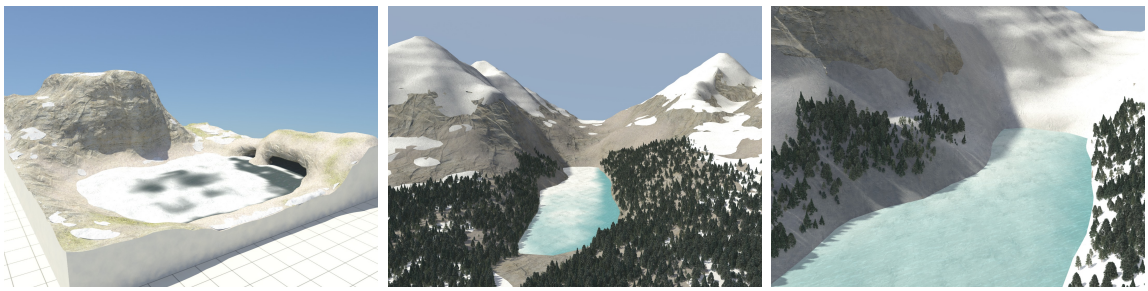
Les résultats obtenus en utilisant la méthode de génération d'objets par fragments démontrent l'intérêt d'introduire de la variété pour augmenter le réalisme des scènes.

L'outil étant bien adapté aux habitudes des artistes, sa prise en main est donc rapide. Les modifications sont également réalisées par les artistes ; le contenu graphique généré correspond donc à leurs attentes et leur style graphique est respecté. Finalement, le faible surcoût mémoire engendré rend notre méthode très adaptée au contexte du jeu vidéo. D'autre part, le temps nécessaire pour obtenir un nombre très important d'objets est faible. Le gain en temps de production est considérable pour une société de création de jeux vidéos. À l'heure actuelle, les fragments sont obtenus à partir d'un objet initial. Pour plus de flexibilité, il serait intéressant de développer un outil permettant de modéliser directement les fragments, voire même de les générer.

Bien que cette méthode soit utilisable pour un très grand nombre de type d'objets, ceux-ci restent figés dans l'espace. Dans une problématique de création d'ouvrages d'art, il est impossible de créer des objets qui s'adaptent à des contraintes extérieures telles que le relief d'un paysage ou la largeur d'une rivière. Les modèles paramétrés sont une solution efficace à ce problème. Les résultats présentés illustrent la puissance de ce type d'approche pour la génération d'objets manufacturés. De plus, l'utilisation d'algorithmes procéduraux est très peu coûteuse en mémoire et le nombre de polygones générés ainsi que la création de niveaux de détails sont facilement contrôlables. Cependant, l'artiste doit avoir des notions importantes en programmation. Il est donc nécessaire de créer une interface intuitive simplifiant la mise en place des modèles paramétrés.

Chapitre 3

Création de paysages hivernaux



Sommaire

3.1 Introduction

La simulation des phénomènes naturels et la création de paysages réalistes ont fait l'objet de nombreux travaux. L'utilisation de simulation s'est avérée être une approche très efficace pour simuler les phénomènes de vieillissement (DEJ$^+$99), les écosystèmes (AD05) et l'érosion de terrains (vBBK08).

Dans ce chapitre, nous nous intéressons à la simulation de paysages hivernaux en fonction des conditions climatiques. En hiver, l'apparition de neige et de glace change considérablement l'aspect visuel des paysages. Le manteau neigeux et les couches de glace évoluent à des vitesses différentes selon les conditions climatiques. Les méthodes précédemment proposées consistent généralement à déterminer la quantité de neige pouvant s'accumuler sur le sol et les objets (Fea00, HAH02, FB07). D'autres approches simulent le transport des flocons de neige par le vent (WWXP06, SEN08). Finalement, des méthodes ont été proposées pour simuler la formation des cristaux de glace (KL03, KHL04) ainsi que la formation des stalactites (KAL06). Néanmoins, aucune méthode ne prend en compte de façon précise les différents échanges thermiques qui ont un impact important sur la neige et la glace. Par conséquent, de nombreux phénomènes naturels tels que la fonte de la neige, le gel et le dégel de la glace ne sont pas simulés.

Suivre les changements d'apparence des paysages naturels couverts de neige en fonction de la météo et des caractéristiques de l'environnement est un problème très complexe. Lors des premières chutes de neige, la température du sol est souvent trop élevée pour que la neige puisse s'accumuler. Il faut attendre que le sol ait une température suffisamment froide. De plus, la neige exposée au soleil fond beaucoup plus rapidement que celle se trouvant constamment à l'ombre.

Pour prendre en compte ces phénomènes, notre méthode consiste à simuler l'ensemble des échanges thermiques intervenant dans la nature et nous permet de suivre l'évolution du paysage au cours du temps. À notre connaissance, nous sommes les premiers à proposer une simulation complète des échanges thermiques pour la création de paysages hivernaux. Afin de prendre en compte les conditions climatiques et de contrôler la simulation, nous avons défini un scénario climatique représenté à l'aide de fonctions. Ces fonctions correspondent à la température de l'air et du point de rosée, les précipitations de neige, les cycles jour/nuit et la couverture nuageuse. L'inertie thermique des masses d'eau est prise en compte en simulant la convection naturelle. La fonte de la neige, le gel et le dégel de l'eau s'effectuent par simulation des changements d'état. L'ensemble des simulations est réalisé en utilisant une grille de voxels régulière et les échanges thermiques sont résolus en utilisant une méthode de volumes finis. D'autre part, notre méthode d'instanciation permet de créer rapidement et efficacement la géométrie de la neige et de la glace.

Dans la suite de ce chapitre, nous détaillons la structure de données mise en place, l'ensemble des simulations réalisées et notre méthode d'instanciation de la neige et de la glace.

3.2 Aperçu, structure de données et algorithmes

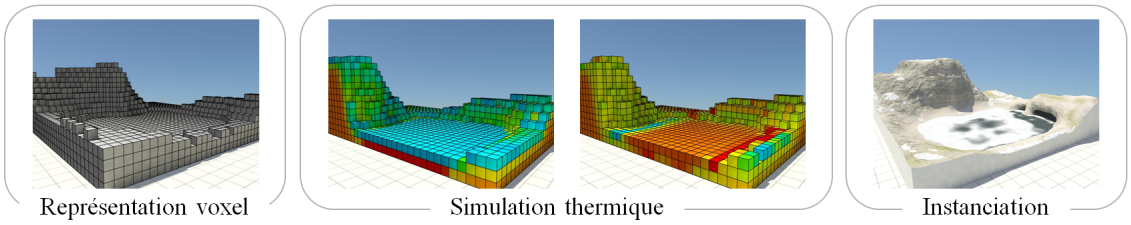


FIGURE 3.1 – Aperçu de notre simulation de paysages hivernaux.

Dans cette section, nous présentons notre méthode ainsi que la structure de données mise en place pour simuler les différents échanges thermiques et pour générer la neige et la glace.

Modéliser et simuler l'évolution du manteau neigeux et de la couche de glace dans les scènes naturelles est un problème très complexe. Capturer les petits détails géométriques sur une très grande scène par simulation en utilisant une grille de voxels précise (un voxel étant un élément de volume cubique au sein de la grille) serait beaucoup trop coûteux en mémoire et en temps de calcul. Par conséquent, nous utilisons une grille de voxels de résolution relativement faible mais suffisante à l'échelle de la scène. Notre simulation est guidée par un scénario climatique permettant de définir l'évolution temporelle des conditions météorologiques. Les surfaces et les textures de la neige et de la glace sont obtenues procéduralement par interprétation des données physiques calculées (Figure 3.1).

Étant donné un modèle initial de terrain, nous générons tout d'abord la représentation en voxels de celui-ci. Nous effectuons ensuite une simulation pour évaluer l'évolution de la température de chaque voxel en fonction du scénario climatique. Celui-ci définit entre autres les variations de température de l'air ainsi que la quantité des précipitations de neige au cours du temps.

La simulation s'effectue en utilisant la méthode des volumes finis qui nous permet de calculer les échanges conductifs, convectifs et radiatifs entre les éléments de la scène. La méthode des volumes finis consiste à résoudre des équations aux dérivées partielles en utilisant des approximations d'intégrales. Ces équations sont résolues de manière approchée sur la grille de voxels constituant le domaine d'étude. Les intégrales de volume d'un terme de divergence sont transformées en intégrales de surface en utilisant le théorème de flux-divergence plus connu sous le nom de théorème de Green-Ostrogradski. Les flux sont alors évalués aux interfaces entre les volumes finis. Les équations utilisées dans notre simulation s'expriment directement sous forme de flux. Cette méthode est, par conséquent, la plus adaptée.

Le processus d'instanciation consiste à générer un maillage texturé représentant la neige et la glace à partir des informations stockées dans la grille de voxels. Le maillage représentant le terrain est obtenu par convolution en utilisant la méthode proposée

par Peytavie et coll. (PGMG09). Cette représentation implicite garantit la continuité de la surface et le maillage triangulaire correspondant peut être généré facilement par les techniques de polygonisation standard de surface implicite. La surface de la neige est générée à l'aide d'un champ de hauteurs en fonction de la quantité de neige stockée dans les voxels (HAH02, FG09). Le maillage de glace est généré en utilisant une méthode de déplacement de surface spécifique. Finalement, la surface est perturbée procéduralement à l'aide d'un bruit pour obtenir une apparence naturelle plus réaliste.

3.2.1 Structure de données

La scène est représentée en utilisant une décomposition en grille de voxels noté $\mathcal{V}_{ijk}$. Chaque voxel possède un identifiant caractérisant le matériau (Figure 3.2) et des informations sur les propriétés thermiques du voxel telles que la quantité d'énergie échangée à chaque pas de temps et sa température. Nous stockons également un pourcentage permettant de connaitre la quantité exacte de chaque matériau pour les voxels contenant deux matériaux. Ce pourcentage est utilisé lors des changements d'état. Dans notre implémentation, les voxels peuvent être de la roche, de la neige, de l'air, de l'eau, de la glace, de la terre et du sable.

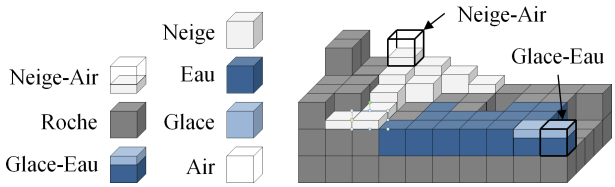


FIGURE 3.2 – Représentation en voxel de la scène avec des voxels comportant un ou deux matériaux.

Certains voxels peuvent contenir des matériaux différents (Figure 3.2) que nous appellerons par la suite voxels deux-matériaux. Ces voxels deux-matériaux sont nécessaires afin de simuler les changements d'état. Par exemple, les voxels eau-glace permettent de simuler la fonte de la glace en eau. La quantité de chaque matériau est indiquée en utilisant un pourcentage du volume du voxel.

Dans notre implémentation, deux types de voxels peuvent contenir deux matériaux : les voxels air-neige et eau-glace. Ces voxels nous permettent de connaître l'exacte quantité de neige, d'air, de glace et d'eau lors des changements d'état et de la chute de neige.

3.2.2 Algorithme de simulation

Notre simulation s'effectue en quatre étapes (Figure 3.3) :

1. calcul des paramètres de l'environnement en utilisant le scénario climatique.

2. Simulation de la chute et l'accumulation de la neige sur le sol en fonction des paramètres de l'environnement définis à l'aide du scénario climatique.
3. Calcul des échanges thermiques entre les voxels $\mathcal{V}_{ijk}$ de la scène en simulant les différents flux de chaleur passant au travers des faces des voxels. La variation de température est calculée à partir de la variation d'énergie $\mathrm{d}\,Q$ et des caractéristiques du matériau de chaque voxel.
4. Calcul des changements d'état de chaque voxel $\mathcal{V}_{ijk}$ et effectuons la mise à jour de la température des voxels d'eau en simulant la convection naturelle.

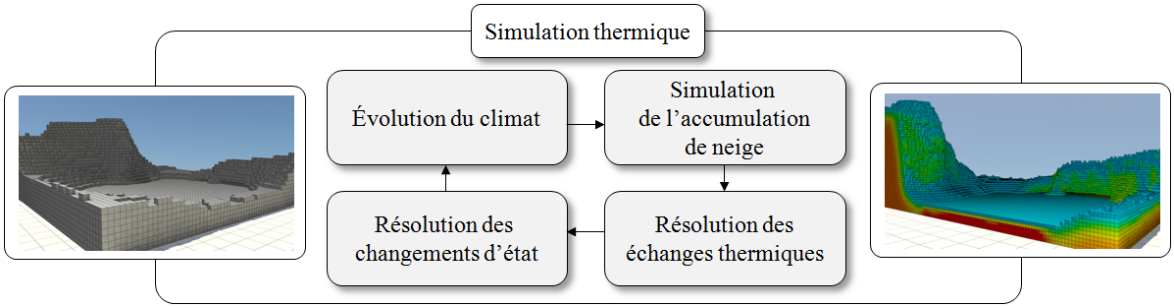
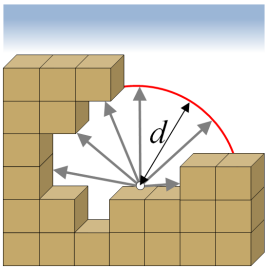


FIGURE 3.3 – Algorithme principal de la simulation.

3.2.2.1 Accumulation de la neige

L'accumulation de la neige sur le sol est effectuée en utilisant la méthode décrite par Foldes et coll. (FB07). La neige est ensuite stabilisée à l'aide d'un algorithme d'angle au repos.



Nous calculons l'accumulation de la neige en fonction de l'accessibilité locale. Celle-ci est calculée en émettant n rayons sur l'hémisphère depuis la surface du sol. Nos rayons collectent de la neige dès lors qu'ils n'intersectent pas le terrain sur une distance inférieure à d. Soit $P(t)$ la quantité de chute de neige à l'instant t. La quantité de neige a s'accumulant sur le sol est calculée de la façon suivante : pour chaque rayon émis, nous vérifions si celui-ci intersecte ou non le terrain. Si c'est le cas, nous calculons la distance entre l'origine d'émission et le point d'intersection. Si la distance est inférieure à d alors la quantité de neige a s'accumulant est incrémentée de $P(t)/n$.

Cette méthode d'accumulation est rapide à mettre en place et peu coûteuse en temps de calcul. Néanmoins, elle ne permet pas de simuler la formation de congères et l'érosion de la neige. Ces caractéristiques sont produites par le vent qui joue également un rôle très important sur la répartition de la neige et peut également la faire sublimer. Dans notre simulation, le vent n'a pas été pris en compte. Cependant notre méthode est compatible avec celle décrite par Wang et coll. (WWXP06).

Après avoir accumulé la neige, nous la stabilisons à l'aide d'un algorithme d'angle au repos. La neige possède un angle au repos noté α pouvant varier entre 15 et 90°. Dans notre implémentation, l'angle est choisi par l'utilisateur.

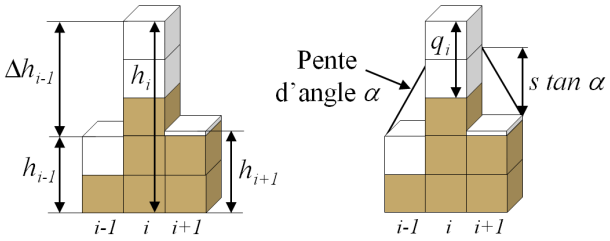


FIGURE 3.4 – Notation pour la stabilisation de la neige.

Soit h_i l'altitude de la pile de voxels centrale, q_i la hauteur de neige de la pile centrale et h_j avec $j \in [1, 4]$ la hauteur des 4 piles de voxels voisines. La différence d'altitude est définie par $\Delta h_j = h - h_j$ (Figure 3.4). La pile est considérée comme instable lorsque $\Delta h_j > s \tan \alpha$ avec s la résolution des voxels. La neige est alors stabilisée en répartissant le surplus de neige sur les piles de voxels voisines. La quantité de neige q_j à déplacer sur la pile j est définie par :

$$q_j = \begin{cases} q_i \text{ si } q_i < \Delta h_j - s \tan \alpha \\ \Delta h_j - s \tan \alpha \text{ sinon} \end{cases}$$

L'angle au repos que nous avons utilisé dans nos simulations est de 80° et correspond à celui de la neige fraîche.

3.2.2.2 Résolution des échanges thermiques

Les échanges thermiques sont évalués sur la structure en voxels présentée précédemment. La résolution des échanges conductifs, convectifs et radiatifs s'effectue par une méthode de volume finis en fonction des propriétés thermiques de chaque élément de la scène. Dans la section 3.4.1, nous présentons en détail l'ensemble de cette simulation.

3.2.2.3 Changement d'état et inertie thermique

Pour simuler la fonte de la neige ainsi que le gel et le dégel de l'eau, il est nécessaire de résoudre les changements d'état. Nous détaillons comment calculer un changement d'état dans la section 3.4.2. L'inertie thermique est également prise en compte en simulant la convection naturelle (Section 3.4.3).

Dans les sections suivantes, nous présentons le scénario climatique ainsi que les détails permettant de simuler les échanges thermiques, les changements d'état et la convection naturelle.

3.3 Scénario climatique

Notre scénario climatique permet de définir l'évolution des conditions météorologiques au cours du temps. Le climat est caractérisé par la température de l'air $T_{\text{air}}(t)$, la température de point de rosée $T_{\text{dp}}(t)$, la quantité de précipitation de neige $P(t)$ (mmh^{-1}), la couverture nuageuse $C(t)$ et les cycles jour/nuit $D(t)$.

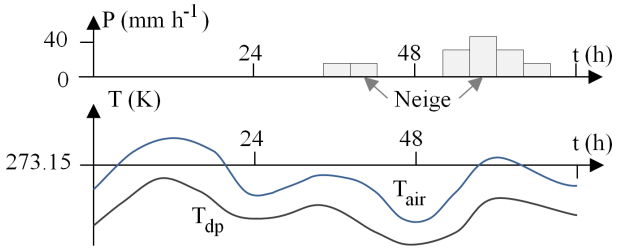


FIGURE 3.5 – Représentation graphique des paramètres de notre scénario climatique.

Les fonctions $T_{\text{air}}(t)$, $T_{\text{dp}}(t)$, $P(t)$ et $C(t)$ sont obtenues à partir de données météorologiques ou contrôlées par l'utilisateur. Les valeurs de la fonction définissant la couverture nuageuse $C(t)$ sont prises dans l'intervalle unité $[0, 1]$, 0 correspondant à un ciel dégagé et 1 un ciel avec une couverture nuageuse dense. La fonction représentant les cycles jour/nuit $D(t)$ est définie procéduralement et dépend de la localisation géographique de la scène (caractérisée par la latitude, la longitude et le jour de l'année). Pour définir cette fonction, nous calculons l'heure de lever et de coucher du soleil en temps solaire (Annexe C). À chaque pas de temps, si l'heure en temps solaire de simulation est comprise entre l'heure de lever et de coucher du soleil alors la fonction $D(t)$ est égale à 1 et à 0 dans le cas contraire. Le temps solaire est la durée entre deux retours successifs du soleil au méridien local (lorsque le soleil est au zénith). Cette durée varie au cours de l'année.

La figure 3.5 illustre les fonctions représentant les températures de l'air et du point de rosée (respectivement T_{air} et T_{dp}) ainsi que la quantité de chute de neige P. Dans notre implémentation, la chute de neige se produit lorsque la température de l'air est inférieure à 274K.

3.4 Simulation thermique

Afin d'obtenir une représentation uniforme des différents échanges thermiques, nous modélisons ces échanges sous forme de flux de chaleur au travers de la surface de contact de nos voxels en utilisant la méthode des volumes finis. Le flux de chaleur ϕ ($\text{J}\,\text{s}^{-1}$) est égal à $\phi = S\,\varphi$ avec S (m^2) l'aire de la surface d'échange et φ la densité de flux de chaleur ($\text{J}\,\text{s}^{-1}m^{-2}$). Après chaque itération, la variation de chaleur $\text{d}Q$ (J) est obtenue par $\text{d}Q = \phi\,\text{d}t$ avec $\text{d}t$ le pas de temps (s).

Capacité thermique	C_p	$\mathrm{J\,kg^{-1}K^{-1}}$
Masse volumique	ρ	$\mathrm{kg\,m^{-3}}$
Conductivité	λ	$\mathrm{J\,s^{-1}m^{-1}K^{-1}}$
Coefficient de transfert convectif	h_{conv}	$\mathrm{J\,s^{-1}}m^{-2}\mathrm{K^{-1}}$
Température de fusion	T_{g}	K
Chaleur latente de fusion	L_{f}	$\mathrm{Jkg^{-1}}$

TABLEAU 3.1 – Notations et unités de grandeurs physiques utilisées dans notre simulation.

Matériau	C_p	ρ	λ	ϵ	a
Roche	1600	2300	2.1	0.45	0.3
Eau	4186	999	0.602	0.95	0.05
Neige	2090	110	0.05	0.82	0.9
Glace	2050	915	2.22	0.97	0.8
Air	1006	1.188	0.026	1	0

TABLEAU 3.2 – Valeurs de certaines grandeurs physiques pour différents matériaux.

Soit $\mathrm{d}V$ un élément de volume élémentaire, ρ la masse volumique du matériau et C_p sa capacité thermique spécifique à pression constante. La conversion entre la variation de chaleur $\mathrm{d}Q$ et la variation de température $\mathrm{d}T$ est obtenue à l'aide de l'équation $\mathrm{d}Q = \rho\, C_p\, \mathrm{d}V\, \mathrm{d}T$. Pour résoudre l'équation non-homogène de la chaleur, nous calculons les différents flux de chaleur entre les éléments de la scène en utilisant les coefficients thermiques du matériau correspondant. La variation de température d'un voxel donné est définie comme suit :

$$\frac{\mathrm{d}T}{\mathrm{d}t} = \frac{S}{\rho\, C_p\, \mathrm{d}V} \sum_i \varphi_i$$

3.4.1 Simulation des échanges thermiques

Chaque matériau présent dans la scène est caractérisé par un ensemble de constantes définissant ses propriétés physiques : la capacité thermique C_p, la masse volumique ρ, la conductivité thermique λ, l'émissivité ϵ, la température de fusion T_{g}, la chaleur latente de fusion L_{f} et l'albédo a. Les fluides tels que l'air et l'eau sont également caractérisés par leur coefficient d'échange convectif noté h_{conv}. La table 3.1 rappelle quelques propriétés thermiques des matériaux ainsi que les notations et unités correspondantes. Pour rappel, l'albédo a et l'émissivité ϵ sont des coefficients sans dimension. La table 3.2 rappelle les valeurs des propriétés physiques que nous avons utilisées lors de nos simulations. La température de fusion de l'eau, de la glace et de la neige est de 273.15 K et

la chaleur latente de fusion est de $333.10^3 \mathrm{J\,kg^{-1}}$.

Un échange thermique se produit dans un milieu et à l'interface entre deux milieux lorsque le gradient de température ∇T est différent de 0. L'échange se produit à partir de la région la plus chaude vers la région la plus froide. Il existe trois grandes catégories d'échanges thermiques :

- les échanges thermiques par **conduction** qui se produisent à l'intérieur d'un milieu ou entre deux milieux en contact. Il n'y a pas de déplacement de la matière constituant le milieu, la chaleur est diffusée progressivement par agitation moléculaire.
- les échanges thermiques par **convection** qui se produisent dans les matériaux fluides tels que l'air et l'eau et à l'interface entre un milieu fluide et un milieu solide. À l'inverse de la conduction, la convection s'effectue par déplacement de la matière. La chaleur est transportée par le mouvement du fluide.
- les échanges thermiques par **radiation** qui sont émis sous forme de rayonnement électromagnétique depuis la surface de tous matériaux ayant une température supérieure à 0 K en contact avec l'air ou le vide.

La figure 3.6 illustre l'ensemble des échanges thermiques qui sont pris en compte dans notre simulation. Dans les paragraphes suivants, nous détaillons ces différents échanges thermiques.

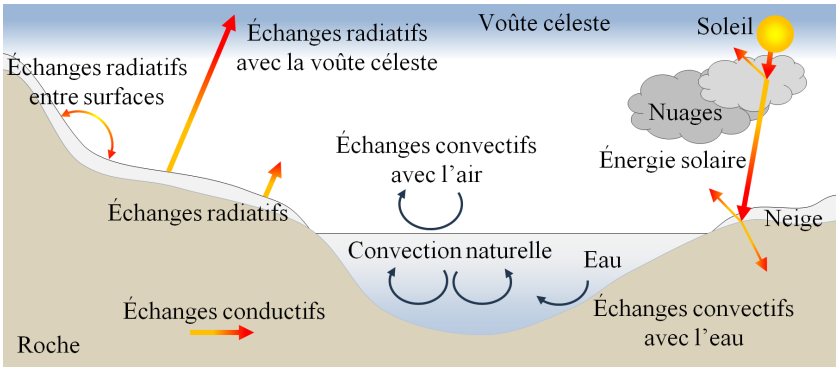
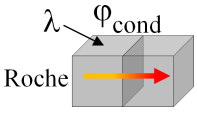


FIGURE 3.6 – Les échanges thermiques pris en compte dans notre simulation.

3.4.1.1 Échanges thermiques par conduction



La loi de la conduction a été introduite par Joseph Fourier qui était un mathématicien et physicien français, connu pour ses travaux sur la décomposition de fonctions périodiques en séries trigonométriques convergentes appelées séries de Fourier et leurs applications au problème de la propagation de la chaleur. Il a établi que la densité de flux de chaleur φ_{cond} $(\mathrm{J\,s^{-1}m^{-2}})$ est proportionnelle au gradient de température ∇T. Soit λ la conductivité thermique du

matériau. La loi s'écrit de la façon suivante :

$$\varphi_{\text{cond}} = -\lambda \nabla T \tag{3.1}$$

Dans le cas général où le matériau est non-homogène, λ dépend de la position $\mathbf{p}$ et de la température T à cette position. La conductivité $\lambda(\mathbf{p}, T)$ s'exprime alors sous forme de tenseur :

$$\lambda(\mathbf{p}, T) = \begin{vmatrix} \lambda_{xx}(T) & \lambda_{xy}(T) & \lambda_{xz}(T) \\ \lambda_{yx}(T) & \lambda_{yy}(T) & \lambda_{yz}(T) \\ \lambda_{zx}(T) & \lambda_{zy}(T) & \lambda_{zz}(T) \end{vmatrix}$$

Cependant, la majorité des matériaux sont quasi-homogènes. Par conséquent, nous considérons le matériau comme complètement homogène et approximons la conductivité par $\lambda = \lambda(T)$. La valeur de λ reste liée à la température du matériau. Néanmoins, dans notre simulation, les variations de température restent faibles. Il est donc raisonnable d'approximer la valeur de la conductivité par une constante qui est généralement choisie pour une température de 293K.

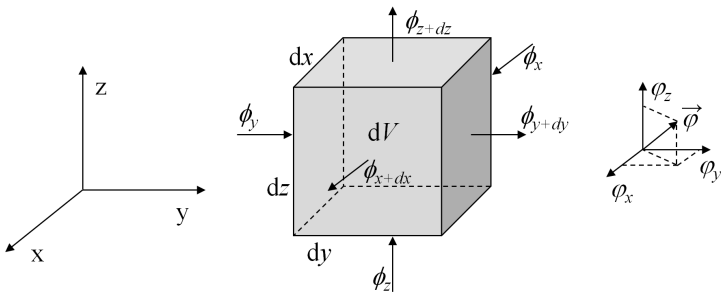


FIGURE 3.7 – Bilan énergétique du volume élémentaire.

L'équation générale différentielle de la conduction de la chaleur est déduite de l'équation introduite par Fourier en effectuant un bilan énergétique sur un volume élémentaire (Figure 3.7) et en invoquant le premier principe de la thermodynamique (conservation de l'énergie). La variation d'énergie interne $\mathrm{d}Q$ est égale à la différence entre l'énergie entrante (provenant de l'extérieur) et l'énergie sortante auxquelles s'ajoute l'énergie dégagée par les sources de chaleur internes s. Ceci ce traduit de la façon suivante :

$$\mathrm{d}Q = \mathrm{d}\phi \, \mathrm{d}t + s \, \mathrm{d}V \, \mathrm{d}t \tag{3.2}$$

Dans le cas qui nous intéresse, les sources de chaleur interne sont nulles ($s = 0$). La conduction est dite *morte*. En développant l'équation 3.2 et en la simplifiant, nous obtenons :

$$\rho \, C_p \, \mathrm{d}T \, \mathrm{d}V = [(\phi_x - \phi_{x+\mathrm{d}x}) + (\phi_y - \phi_{y+\mathrm{d}y}) + (\phi_z - \phi_{z+\mathrm{d}z})] \, \mathrm{d}t \tag{3.3}$$

Or, d'après la loi de Fourier et pour la composante x on a :

$$\phi_x = -\lambda \frac{\partial T}{\partial x} \mathrm{d}y \, \mathrm{d}z$$

et

$$\phi_{x+dx} = \phi_x + \frac{\partial \phi_x}{\partial x}\mathrm{d}x = \left[-\lambda\frac{\partial T}{\partial x} + \frac{\partial}{\partial x}\left(-\lambda\frac{\partial T}{\partial x}\mathrm{d}x\right)\right]\mathrm{d}y\,\mathrm{d}z$$

soit

$$\phi_x - \phi_{x+dx} = \lambda\frac{\partial^2 T}{\partial x^2}\,\mathrm{d}V$$

En procédant de même sur les composantes y et z puis en substituant les termes dans l'équation 3.3 et en simplifiant, nous obtenons l'équation générale différentielle de la conduction de la chaleur :

$$\frac{\mathrm{d}T}{\mathrm{d}t} = \frac{\lambda}{\rho C_p}\nabla^2 T \tag{3.4}$$

Dans notre système, nous avons préféré utiliser l'équation 3.1 de la conduction qui s'exprime sous la forme de densité de flux plutôt que l'équation 3.4 exprimant les échanges de façon différentiels. Les autres échanges que nous calculons s'exprimant également sous forme de densité de flux, cela nous permet de calculer tous les échanges en une seule passe.

3.4.1.2 Conduction à l'interface entre deux matériaux solides

Dans le cas général, le contact entre deux matériaux solides est imparfait. Typiquement, l'interface possède de nombreux interstices remplis d'air ainsi que des points de contact. La présence d'air dont la conductivité est faible (Table 3.2) agit comme une résistance et rend les échanges thermiques par conduction inefficaces. Au niveau des interstices, les échanges thermiques sont dit *interstitiels* tandis qu'au niveau des points de contact, ils sont dit *solides* (Figure 3.8).

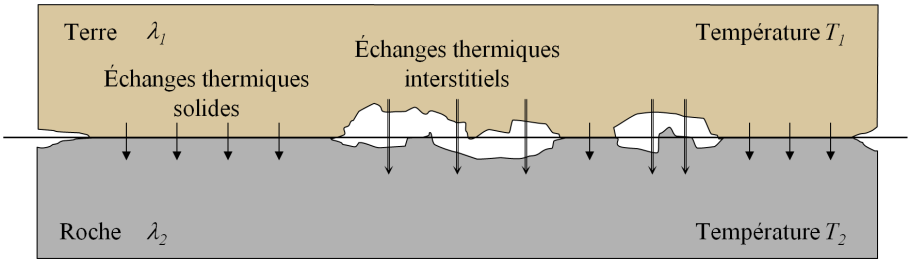


FIGURE 3.8 – Échanges thermiques entre deux matériaux solides en contact.

La résistance créée au niveau de l'interface est prise en compte à l'aide de la conductance interstitielle h_c $(J\,s^{-1}\,m^{-2}\,K^{-1})$ et la densité de flux est évaluée de la façon suivante :

$$\varphi_{\mathrm{cond}} = h_c\,(T_1 - T_2)$$

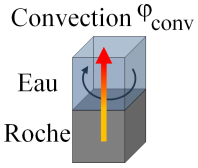
Les valeurs de h_c sont définies expérimentalement et dépendent de nombreux facteurs : la finition de la surface, le type des matériaux en contact, la pression exercée

sur les solides, le type de substance se trouvant dans les interstices et la température à l'interface. Déterminer h_c avec exactitude est donc très complexe. Par conséquent, nous approximons le contact entre deux matériaux et le considérons comme étant parfait. La conductivité λ doit alors être calculée en fonction des conductivités respectives λ_1 et λ_2 des matériaux en utilisant l'analogie des résistances électriques en série :

$$\lambda = \frac{1}{\frac{1}{2\lambda_1} + \frac{1}{2\lambda_2}}$$

La densité de flux à l'interface est alors évaluée en utilisant la loi de Fourier (Equation 3.1) et la conductivité précédemment calculée.

3.4.1.3 Échanges thermiques convectifs aux interfaces avec un fluide



La loi de Newton permet d'évaluer la densité de flux de chaleur à l'interface entre un fluide et un autre matériau en fonction de l'agitation du fluide. Elle a été introduite par Isaac Newton qui était un philosophe, mathématicien, physicien, alchimiste et astronome anglais. Figure emblématique des sciences, il est actuellement surtout reconnu pour sa théorie de la gravitation universelle et la création, en concurrence avec Leibniz, du calcul infinitésimal.

Soit h_{conv} le coefficient d'échange convectif, T_{fluide} la température moyenne du fluide et T_{surf} la température de la surface du matériau. La loi de Newton s'écrit de la façon suivante :

$$\varphi_{\text{conv}} = h_{\text{conv}}(T_{\text{surf}} - T_{\text{fluide}})$$

Le degré d'agitation du fluide est exprimé par le coefficient d'échange convectif h_{conv}. Plus ce coefficient est grand, plus le fluide est agité et par conséquent la quantité d'énergie apportée ou arrachée par le fluide est importante.

Pour calculer la valeur du flux, il faut tout d'abord déterminer la valeur de h_{conv} qui s'exprime de la façon suivante :

$$h_{\text{conv}} = \frac{\lambda_{\text{fluide}}}{L} Nu_L \tag{3.5}$$

avec Nu_L le nombre de Nusselt (ou Nusselt), λ_{fluide} la conductivité du fluide et L la longueur caractéristique du corps considéré (diamètre d'une canalisation, longueur d'une plaque plane). L'objectif est donc de calculer le nombre de Nusselt.

Dans le cas général, déterminer le Nusselt analytiquement ou par une méthode numérique est très complexe et dépend du type de convection considéré (forcée ou naturelle).

– La convection forcée se manifeste lorsque le mouvement du fluide est une conséquence d'une action extérieure (présence de vent, écoulement d'eau le long d'une pente provoqué par la différence d'altitude).

– La convection naturelle est provoquée par les différences de température dans le fluide entrainant, sous l'effet de la poussée d'Archimède, un mouvement ascendant de fluide chaud et un mouvement descendant de fluide froid. Ceci est dû au fait que sa masse volumique évolue en fonction de sa température. La figure 3.9 illustre le changement de masse volumique de l'air en fonction de la température.

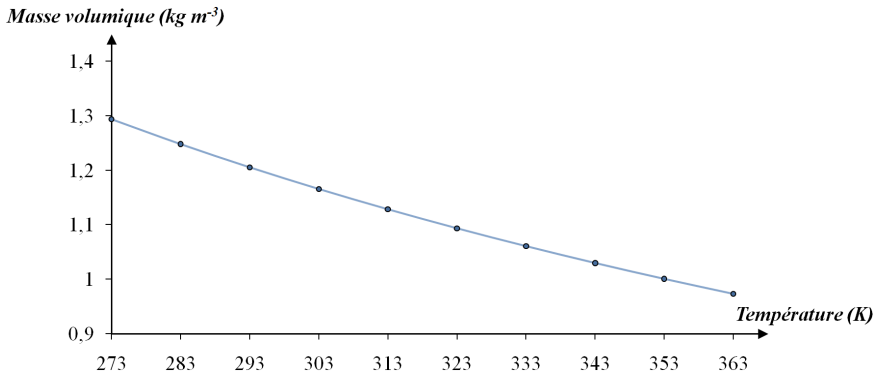


FIGURE 3.9 – Évolution de la masse volumique de l'air en fonction de la température.

En convection forcée, le Nusselt peut être calculé dans un certain nombre de cas particuliers. Pour une surface plane de longueur caractéristique L soumis à un flux laminaire, transitionnel et turbulent, le Nusselt s'exprime de la façon suivante (Chu76) :

$$Nu_L = 0.45 + (0.6774\,\kappa^{1/2}) \left[1 + \frac{(\kappa/12500)^{3/5}}{\left[1 + (\kappa_u/\kappa)^{7/2}\right]^{2/5}} \right]^{1/2} \tag{3.6}$$

avec

$$\kappa \equiv Re_L\, Pr^{2/3} \left[1 + \left(\frac{0.0468}{Pr} \right)^{2/3} \right]^{-1/2}$$

et $\kappa_u \approx 1.875\,\kappa$ pour $Re_L = 5.10^5$ qui correspond au nombre de Reynolds critique traduisant le passage du régime laminaire au régime turbulent. Le coefficient d'échange convectif h_{conv} est alors obtenu en injectant l'équation 3.6 dans l'équation 3.5. Pr est le nombre de Prandtl et il permet de comparer la rapidité des phénomènes thermiques et des phénomènes hydrodynamiques dans un fluide. Il s'exprime par $Pr = v/\gamma$, avec v la viscosité cinématique du fluide et $\gamma = \lambda_{\text{fluide}}/\rho\, C_p$ la diffusivité thermique. Re_L est le nombre de Reynolds et représente le rapport entre les forces d'inertie et les forces visqueuses. Il s'exprime par $Re_L = U\, L/v$ avec U la vélocité du fluide. L'apparition du terme U engendre la nécessité de simuler le champ de vélocités du fluide.

Dans notre système, les masses d'eau sont considérées comme étant statiques. Par conséquent, les courants pouvant exister au sein du volume d'eau sont exclusivement provoqués par la convection naturelle. Cependant, la prise en compte de rivières ou de torrents est possible et nécessite donc de calculer h_{conv} à l'aide de l'équation 3.6. De plus, nous n'avons pas simulé la présence de vent. Les mouvements de l'air résultent donc

seulement de la convection naturelle. Or, notre méthode est compatible avec celle de Wang et coll. (WWXP06) qui permet de simuler le vent. De même que pour les rivières ou les torrents, en présence de vent, h_{conv} doit être calculé en utilisant l'équation 3.6.

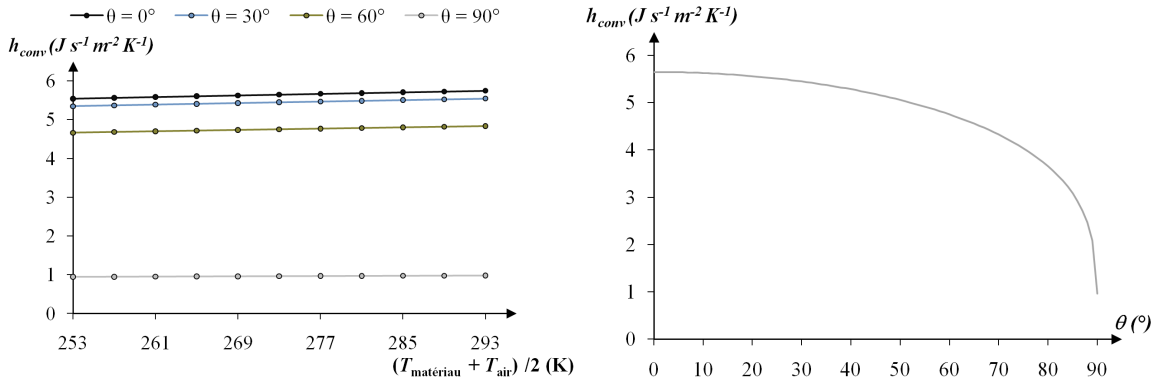


FIGURE 3.10 – Évolution du coefficient convectif de l'air en convection naturelle. En fonction de la température pour différentes inclinaisons de surface (gauche) et en fonction de l'inclinaison de la surface pour une température de 273K (droite).

En convection naturelle, le nombre de Nusselt Nu_L s'exprime en fonction du nombre de Rayleigh Ra_L et du nombre de Prandlt Pr. Le nombre de Rayleigh caractérise le transfert de chaleur au sein d'un fluide en convection naturelle. En dessous d'une valeur critique spécifique au fluide, les échanges de chaleur dans le fluide s'effectuent principalement par conduction et au-delà, ils s'effectuent principalement par convection. Le nombre de Rayleigh s'exprime de la façon suivante :

$$Ra_L = \frac{g\,\tau\,[(T_{\text{roche}} + T_{\text{fluide}})/2]\;L^3}{\gamma\,v} \tag{3.7}$$

avec g la constante gravitationnelle et τ le coefficient de dilatation thermique. Pour une surface plane verticale, le Nusselt s'exprime de la façon suivante (CC75) :

$$Nu_L = 0.68 + 0.67\,Ra_L^{1/4}\left[1 + \left(\frac{0.492}{Pr}\right)^{9/16}\right]^{-4/9}$$

Pour une surface inclinée d'un angle θ par rapport à la verticale, le Nusselt peut s'exprimer en fonction de l'équation précédente en remplaçant la constante de gravitation g de l'équation 3.7 par $g\,\cos(\theta)$ (Ric53).

Les figures 3.10 et 3.11 illustrent les courbes de variation du coefficient convectif h_{conv} de l'air et de l'eau en fonction de la température et de l'inclinaison de la surface. Les coefficients ne varient presque pas en fonction de la température et peuvent donc être considérés comme constants. Cependant, l'inclinaison de la surface impacte énormément sur la valeur du coefficient. Par conséquent, dans notre simulation, nous calculons la valeur de h_{conv} en fonction de l'inclinaison de la surface au point considéré.

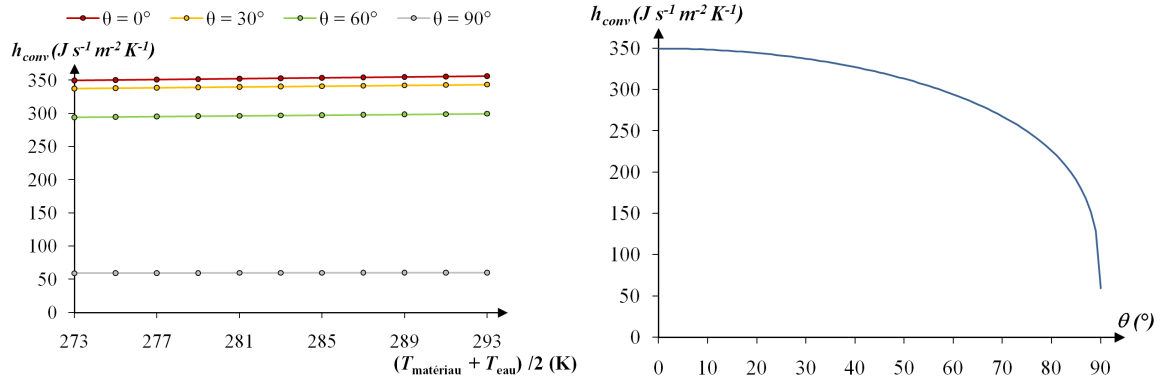
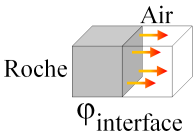


FIGURE 3.11 – Évolution du coefficient convectif de l'eau en convection naturelle. Les valeurs du coefficient sont beaucoup plus importantes que pour l'air.

3.4.1.4 Échanges thermiques par radiation



La plupart des matériaux présents dans la nature peuvent être considérés comme des corps gris. Un corps gris émet de l'énergie par rayonnement de façon diffuse et isotrope. Son émissivité est pour toutes les longueurs d'ondes dans un rapport constant avec celui d'un corps noir de même température (un corps noir étant un objet idéal qui absorbe toute l'énergie électromagnétique qu'il reçoit, sans en réfléchir ni en transmettre). L'émissivité d'un matériau est la capacité relative de sa surface à émettre la chaleur par radiation. C'est le ratio entre l'énergie émise par le matériau sous forme de rayonnement et celle qu'un corps noir émettrait à la même température. L'émissivité est donc la mesure de la capacité d'un matériau à émettre l'énergie absorbée sous forme de radiation. L'énergie totale rayonnée par un corps noir peut être calculée à l'aide de la loi de Stefan-Boltzmann qui a été découverte expérimentalement par Joseph Stefan (physicien, mathématicien et poète Slovène) et dont les fondations théoriques ont été posées dans le cadre de la thermodynamique par un de ses étudiants en doctorat, Ludwig Boltzmann en 1884. L'hypothèse du corps gris permet le calcul direct de la densité de flux de chaleur émise de la surface du matériau dont la température est connue en pondérant la loi de Stefan-Boltzmann par l'emissivité du matériau.

Soit ϵ_{roche} l'émissivité de la roche, σ la constante de Stefan-Boltzmann ($\text{Js}^{-1}\text{m}^{-2}\text{K}^{-4}$) et T_{roche} la température de la roche en surface. La loi de Stefan-Boltzmann établit que la puissance totale rayonnée par unité de surface dans le demi-espace s'exprime par l'équation :

$$\varphi_{\text{rad}} = \epsilon_{\text{roche}}\, \sigma\, T_{\text{roche}}^4$$

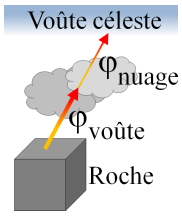
Soit T_{air} la température de l'air environnant. À la frontière entre l'air et un matériau tel que la roche à température T_{roche}, la densité de flux s'exprime par la relation suivante :

$$\varphi_{\text{interface}} = \epsilon_{\text{roche}}\, \epsilon_{\text{air}}\, \sigma\, (T_{\text{roche}}^4 - T_{\text{air}}^4)$$

L'air est considéré comme un corps noir. Par conséquent, son émissivité ϵ_{air} est égale à

1 et l'équation précédente se simplifie.

3.4.1.5 Échanges radiatifs avec la voûte céleste



Soit $\epsilon_{\text{voûte}}$ l'émissivité de la voûte céleste, ϵ_{roche} l'émissivité de la roche, σ la constante de Stefan-Boltzmann précédemment définie, $T_{\text{voûte}}$ la température de la voûte céleste et T_{roche} la température du voxel de roche. La densité de flux de chaleur entre la surface et la voûte céleste est définie comme suit :

$$\varphi_{\text{voûte}} = \epsilon_{\text{roche}}\,\epsilon_{\text{voûte}}\,\sigma\,(T^4_{\text{roche}} - T^4_{\text{voûte}})$$

On peut simplifier le calcul en considérant la voûte céleste comme un corps noir ayant une température $T_{\text{voûte}}$ qui prend en compte son emissivité. Par conséquent, l'équation précédente se simplifie en fixant l'émissivité de la voûte céleste $\epsilon_{\text{voûte}}$ à 1.

Afin de calculer l'échange radiatif entre la surface et la voûte céleste, nous devons calculer la température $T_{\text{voûte}}(t)$ de la voûte céleste. Celle-ci dépend de la température de l'air $T_{\text{air}}(t)$ donnée en Kelvin et de la température de point de rosée $T_{\text{dp}}(t)$ donnée en degrés Celsius pour un ciel sans nuage. Soit $h(t) \in [0, 24]$ l'heure du jour à partir de minuit. La température de la voûte céleste est donnée par la relation suivante (LILV08) :

$$\begin{aligned} T_{\text{voûte}}(t) &= T_{\text{air}}(t)\,\big(0.711 + 0.0056\,T_{\text{dp}}(t) \\ &+ 7.3 \times 10^{-5} T_{\text{dp}}(t)^2 + 0.013\cos(2\pi\,h(t)/24)\big)^{1/4} \end{aligned}$$

Cette équation simule la diminution de la température de la voûte céleste avec la diminution de la quantité de vapeur d'eau présente dans l'air en fonction de la température du point de rosée T_{dp}.

Lorsque des nuages sont présents dans l'atmosphère, la valeur de $\varphi_{\text{voûte}}$ diminue. Afin de modéliser ce phénomène, nous pondérons la densité de flux de chaleur par l'indice de clarté noté $I_{\text{c}}(t)$ dont la valeur est comprise entre 0.3 et 0.8 (DB91). Dans notre scénario climatique, $C(t) \in [0, 1]$ représente la couverture nuageuse. Par conséquent, nous définissons l'indice de clarté tel que $I_{\text{c}}(t) = 0.8 - 0.5\,C(t)$. De plus, nous calculons un facteur de visibilité du ciel F permettant de définir la proportion de ciel visible au point considéré de la surface. Ce facteur correspond au ratio entre le nombre de rayons atteignant le ciel r_c et le nombre de rayons émis depuis la surface r_t : $F = r_c/r_t$. La direction angulaire (α, β) des rayons est choisie de façon uniforme sur l'hémisphère (Figure 3.12). Soit $\mathbf{p}(p_x, p_y)$ un point obtenu par échantillonnage uniforme du disque unitaire D. La direction angulaire est définie de la façon suivante :

$$\begin{aligned} \alpha &= \cos^{-1} p_x \\ \beta &= 2\pi\,p_y \end{aligned} \tag{3.8}$$

La densité de flux de chaleur entre la surface et la voûte céleste en présence d'un ciel nuageux est définie comme suit :

$$\varphi_{\text{nuage}} = F\,I_{\text{c}}(t)\,\varphi_{\text{voûte}}$$

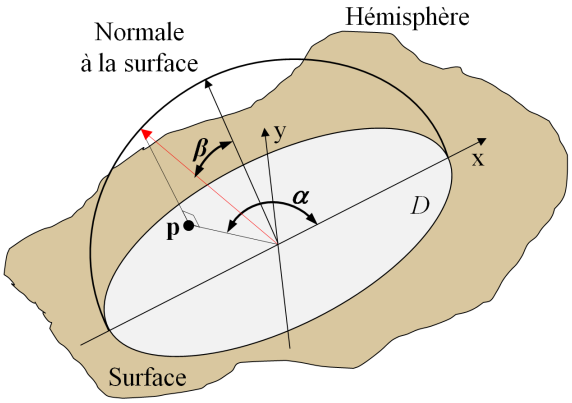
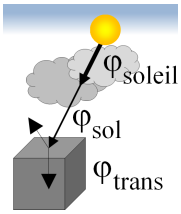


FIGURE 3.12 – Direction angulaire d'un rayon sur l'hémisphère.

3.4.1.6 Énergie solaire



L'énergie solaire représente la contribution directe des rayons du soleil sur la surface des objets. Pour évaluer cette contribution sur la surface des objets, nous devons calculer le rayonnement extraterrestre noté φ_{soleil} ($\mathrm{J\,s^{-1}m^{-2}}$). Il correspond à la densité de flux de chaleur reçue depuis le soleil en haute atmosphère.

Soit n le jour de l'année et θ l'angle d'incidence des rayons solaires avec la surface. Le rayonnement solaire avant filtrage atmosphérique est donné par (DB91) :

$$\varphi_{\text{soleil}} = 1367\ (1 + 0.033\cos(2\pi\, n/365))\ \cos\theta$$

$\cos\theta$ est calculé à partir de la latitude, la déclinaison, l'azimut, l'inclinaison de la surface et l'angle horaire du soleil (Annexe A).

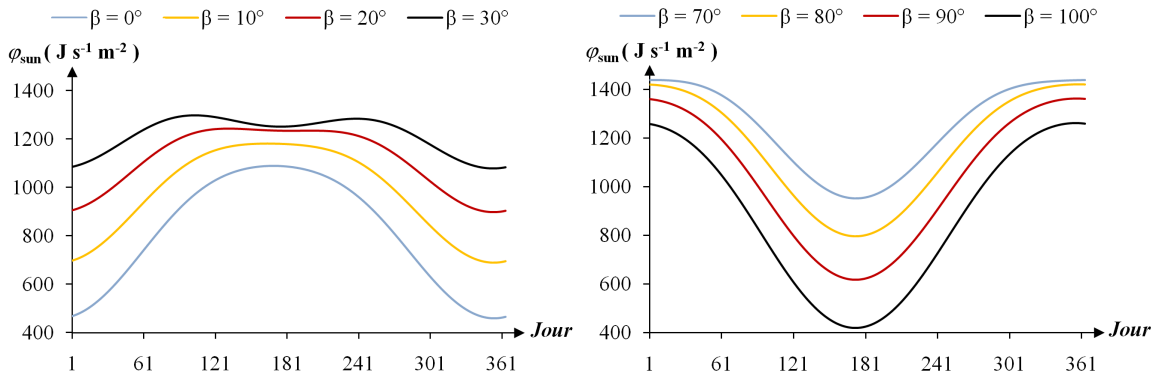


FIGURE 3.13 – Évolution de l'énergie solaire avant filtrage atmosphérique au cours de l'année pour différentes inclinaisons de surface.

La figure 3.13 illustre l'évolution, sur une année, de l'énergie solaire arrivant sur une surface ayant différentes inclinaisons β avant filtrage atmosphérique. La surface

est orientée plein sud (l'azimut est de 180°). Les valeurs du flux sont évaluées lorsque le soleil se trouve au zénith (l'angle horaire est de 0°). La latitude est de 51.41° Nord et correspond à la localisation géographique de Lake Louise (Alberta, Canada).

L'atmosphère réfléchi 33% de l'énergie dans l'espace. Elle est également filtrée par l'atmosphère et par les nuages. Par conséquent, nous pondérons la densité de flux de chaleur φ_{soleil} par l'indice de clarté $I_{\text{c}}(t)$ et par la constante 0.67 pour calculer l'énergie reçue sur la surface du sol :

$$\varphi_{\text{sol}} = 0.67\, I_{\text{c}}(t)\, \varphi_{\text{soleil}}$$

Une partie de cette énergie est réfléchie par la surface. L'énergie transmise au sol φ_{trans} est évaluée en utilisant l'albédo $a \in [0, 1]$:

$$\varphi_{\text{trans}} = (1 - a)\, \varphi_{\text{sol}}$$

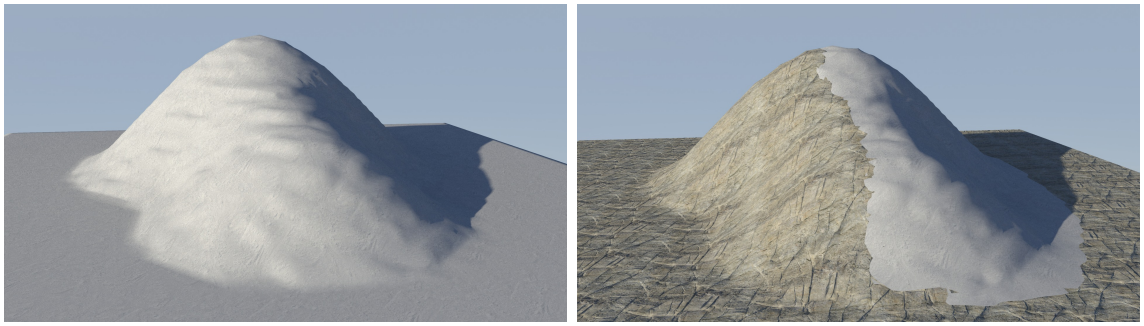


FIGURE 3.14 – Influence du rayonnement solaire sur la fonte du manteau neigeux.

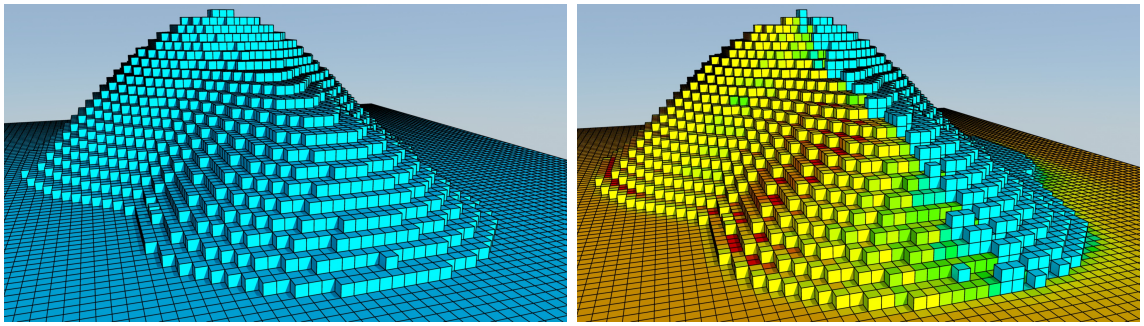


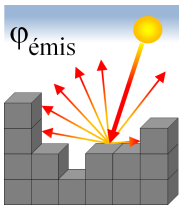
FIGURE 3.15 – Une coupe thermique montrant le côté exposé au rayonnement solaire plus chaud.

L'albédo est le ratio entre l'énergie solaire reflétée et l'énergie solaire incidente (la Table 3.2 rappelle les valeurs d'albédo pour les différents matériaux utilisés dans notre simulation). Dans notre implémentation, le calcul de l'énergie solaire s'effectue durant la période diurne lorsque $D(t) = 1$. Pour chaque voxel, un rayon est émis dans la direction du soleil. Si ce rayon n'intersecte pas le terrain, alors le voxel reçoit de l'énergie solaire. Le calcul de la position du soleil dans le ciel est détaillé en Annexe B.

La figure 3.14 illustre des collines recouvertes de neige et l'évolution du manteau neigeux lorsqu'il est exposé au rayonnement solaire. Dans cette simulation, la température de l'air a été fixée à 273.15 K afin de limiter son effet sur la neige. Comme prévu, la neige fond sur les faces des collines orientées au sud qui sont plus exposées au rayonnement solaire.

La figure 3.15 illustre la température des voxels au début et à la fin de la simulation. Les zones ombragées sont plus froides que les zones ensoleillées.

3.4.1.7 Échanges radiatifs entre surfaces



Les échanges radiatifs entre surfaces (radiosité) sont un phénomène visible qui joue un rôle très important en montagne : les rayons du soleil sont réfléchis par la face ensoleillée de la vallée, faisant fondre la neige accumulée sur le bas de la face opposée. La radiosité correspond à la quantité totale d'énergie émise et réfléchie par une surface. Les calculs de radiosité sont très fréquents en informatique graphique et sont utilisés pour l'illumination globale des scènes. Soit $\mathbf{p}$ un point sur la surface, E_o l'énergie ayant pour direction de sortie $\vec{\omega}_o$, E_e l'énergie émise par la surface selon la loi de Stefan-Boltzmann, E_i l'énergie incidente ayant pour direction $\vec{\omega}_i$, B une fonction représentant la façon dont l'énergie est reflétée par la surface (BRDF) et Ω l'hémisphère d'intégration. L'énergie en sortie $E_o(\mathbf{p}, \vec{\omega}_o)$ est définie comme suit (Figure 3.16) :

$$E_o(\mathbf{p}, \vec{\omega}_o) = E_e(\mathbf{p}, \vec{\omega}_o) + \int_\Omega B(\mathbf{p}, \vec{\omega}_i \rightarrow \vec{\omega}_o)\, E_i(\mathbf{p}, \vec{\omega}_i) \cos\theta_i d\vec{\omega}_i$$

La résolution de cette équation est très complexe et est généralement effectuée par itération à l'aide de la méthode d'intégration de Monte Carlo.

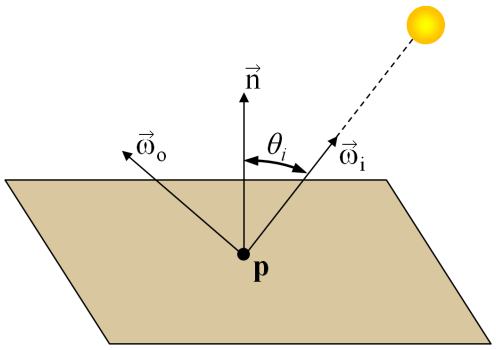


FIGURE 3.16 – Direction d'incidence et de sortie de l'énergie en un point $\mathbf{p}$ de la surface.

Au lieu de simuler la radiosité complète dans la scène, nous considérons seulement le premier rebond de chaleur lors de la réflexion sur la surface de la scène. Les autres rebonds successifs sont considérés comme négligeables.

Rappelons que φ_{sol} représente la densité de flux de chaleur provenant du soleil et atteignant la surface du sol. La quantité d'énergie reflétée par la surface du sol est définie comme suit :

$$\varphi_{\text{réfléchi}} = a\,\varphi_{\text{sol}}$$

De plus, chaque voxel en contact avec l'air émet de l'énergie radiative sous forme de rayonnement électromagnétique notée $\varphi_{\text{interface}}$. Par conséquent, la quantité totale d'énergie émise depuis la surface est :

$$\varphi_{\text{émis}} = \varphi_{\text{réfléchi}} + \varphi_{\text{interface}}$$

Dans notre implémentation, nous simulons la radiosité en émettant des rayons depuis la surface des côtés des voxels en contact avec l'air. La direction des rayons est choisie uniformément sur l'hémisphère de la même façon que pour calculer le facteur de visibilité du ciel. La réflexion de l'énergie étant considérée diffuse, la fonction B est simplement égale à $1/\pi$. De plus, l'énergie s'atténue proportionnellement avec l'inverse de la distance au carré. Soit n le nombre de rayons émis et θ_i l'angle d'incidence, l'énergie reçue en un point $\mathbf{p}$ provenant d'un point $\mathbf{i}$ est :

$$\varphi_{\text{reçue}} = \frac{\varphi_{\text{émis}}}{\pi \parallel \mathbf{p} - \mathbf{i} \parallel n} \cos\theta_i$$

Une partie de l'énergie reçue est reflétée par la surface réceptrice, l'énergie transmise φ_{trans} est :

$$\varphi_{\text{trans}} = (1 - a)\,\varphi_{\text{reçue}}$$

3.4.2 Changement d'état, infiltration et évaporation

Dans notre système, un matériau peut avoir trois états différents : solide, liquide ou gaz. Un changement d'état correspond à la transition d'un matériau d'un état à un autre. Une transition d'un état solide à un état liquide requiert que le matériau soit à température de fusion T_{g}. La température de fusion dépend du type de matériau considéré. Considérons une transition d'un état solide à un état liquide. Lorsque la température de fusion est atteinte, l'énergie injectée dans le matériau n'augmente plus sa température. Celle-ci est absorbée dans un puits thermique Q_{puits}. Lorsque l'énergie absorbée atteint la chaleur latente de fusion L_{f} pour le volume de matériau considéré, il devient complètement liquide. La chaleur latente de fusion L_{f} dépend également de la nature du matériau.

Lorsque le matériau a atteint la température de fusion et que l'énergie absorbée par le puits thermique Q_{puits} augmente, le matériau est considéré comme ayant deux états simultanément. Suivre exactement l'évolution du front de liquéfaction requiert la résolution du problème de Stefan qui est un problème complexe et demande beaucoup de temps de calcul. Par conséquent, nous approximons ce problème en utilisant un pourcentage de matériau permettant d'évaluer exactement la quantité de matière

liquide et solide sans avoir à connaître la position exacte du front. Celui-ci est évalué de la façon suivante :

$$r = 1 - \frac{Q_{\text{puits}}}{L_{\text{f}}\,\rho\,V}$$

Dans notre implémentation, l'eau, la neige et la glace sont les trois matériaux pouvant entrer en changement d'état.

La fonte de la neige ou de la glace en eau est un phénomène complexe impliquant la thermique ainsi que la dynamique des fluides. Lorsque la glace ou la neige fondent, elle sont converties en eau qui peut être partiellement ou totalement absorbée par le sol, changeant ainsi ses propriétés thermiques. L'eau peut également ruisseler sur la surface du sol et générer des flaques ou s'accumuler dans les rivières ou les lacs. L'eau ruisselante peut également geler et former des stalactites en fonction de sa vitesse, de la température du sol et de la température de l'air.

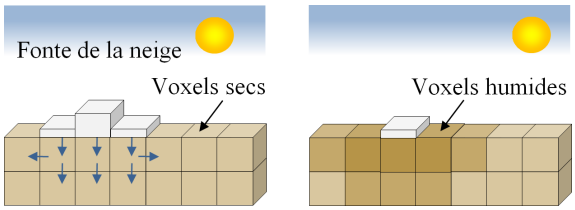


FIGURE 3.17 – L'eau provenant de la fonte de la neige est absorbée par le sol, les propriétés thermiques des voxels de sol sont modifiées.

Dans notre implémentation, nous effectuons un certain nombre d'approximations et ne simulons pas la dynamique de l'eau qui ruisselle. Celle provenant de la fonte de la neige et de la glace est entièrement absorbée par le sol ou par les grands volumes d'eau existants tels que les lacs et les rivières. Lorsqu'elle est absorbée par un grand volume d'eau, nous modifions la température des voxels d'eau correspondants. Nous simulons l'absorption de l'eau par les voxels de sol comme suit.

Les voxels de sol sont caractérisés par un coefficient d'humidité noté w qui définit la quantité d'eau maximale pouvant s'infiltrer par les pores et les fissures du sol. Nous approximons la porosité et l'action de la capillarité par diffusion de l'eau absorbée aux voxels de sols voisins (Figure 3.17).

L'évaporation de l'eau est un phénomène très complexe. De nombreux facteurs tels que le degré d'humidité dans l'air, les impuretés présentes dans l'eau, la force du vent, la pression atmosphérique, la température et la masse volumique du sol doivent être pris en compte. Cependant, l'évaporation peut s'apparenter à un changement d'état du fait de la transformation de l'eau en vapeur. Dans notre implémentation, nous approximons l'évaporation par un simple changement d'état et prenons donc seulement en compte les facteurs de température et de masse volumique du sol. En fonction de la quantité d'humidité q_h infiltrée dans le sol, nous calculons l'énergie E_{vap} nécessaire pour vaporiser cette humidité à l'aide de la chaleur latente de vaporisation de l'eau

($L_\mathrm{v} = 2\ 257.10^3 \mathrm{Jkg}^{-1}$) :

$$E_\mathrm{vap} = q_h\, L_\mathrm{v}\, V\, \rho$$

À chaque itération, nous calculons le nouveau degré d'humidité en fonction de l'énergie Q injectée dans le voxel. Cette énergie est entièrement utilisée pour évaporer l'eau et, par conséquent, la température du voxel reste inchangée. La nouvelle quantité d'humidité q_h est calculée de la façon suivante :

$$q_h = q_h - q_h \frac{Q}{E_\mathrm{vap}}$$

3.4.3 Convection naturelle

Tout le rayonnement solaire direct pénètre les masses d'eau par l'interface air-eau. Une quantité significative de la lumière est réfléchie par la surface de l'eau, tout particulièrement lorsque l'angle incident des rayons du soleil qui frappent la surface est faible. De toute la lumière qui pénètre les masses d'eau, une partie est dispersée vers le haut. L'absorption de l'énergie solaire produit une génération de chaleur.

Lorsque l'eau est réchauffée par l'absorption de l'énergie solaire, il se produit un changement brutal de température entre la zone photique et la zone aphotique. La zone photique est la zone aquatique comprise entre la surface et la profondeur maximale d'un lac ou d'un océan, exposée à une lumière suffisante pour que la photosynthèse se produise. À l'inverse dans la zone aphotique la photosynthèse est impossible.

La masse volumique de l'eau varie en fonction de sa température. Au dessus de 277 K, l'eau chaude est plus légère que l'eau froide. En été, le changement de densité de l'eau assure que l'eau chaude reste en surface et devient de plus en plus chaude, divisant ainsi la masse d'eau horizontalement par une thermocline. La thermocline correspond à la zone de transition thermique rapide entre les eaux superficielles et les eaux profondes. Cette stratification est maintenue tant que l'eau ne subit pas de mouvements dus au vent ou aux courants.

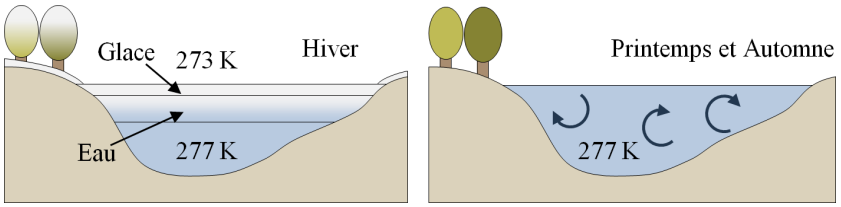


FIGURE 3.18 – La convection naturelle se produit au printemps et en automne tandis qu'en hiver l'eau est stratifiée.

En dessous de 277 K, la courbe de la masse volumique de l'eau est inversée : l'eau froide devient plus légère que l'eau chaude (Figure 3.19). Par conséquent, en automne et au printemps, l'eau chaude et l'eau froide entrent dans un mouvement de convection

homogénéisant sa température. En hiver, lorsque la température de l'air est inférieure à la température de l'eau, les mouvements de convection s'estompent (Figure 3.18).

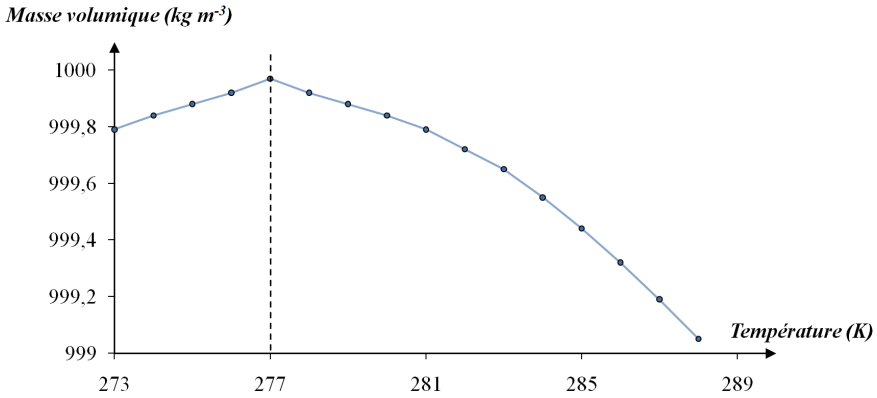


FIGURE 3.19 – Évolution de la masse volumique de l'eau en fonction de la température.

Dans notre implémentation, nous approximons ce phénomène en moyennant la température de tous les voxels d'eau connectés les uns aux autres, lorsque la température de la masse d'eau est supérieure à 277 K. En dessous de 277 K, nous arrêtons l'homogénéisation et calculons les échanges thermiques par conduction, faisant ainsi apparaître un gradient thermique dans l'eau. Cependant, en dessous de 277 K, les mouvements de convection subsistent et diminuent progressivement jusqu'à leur arrêt complet. Pour prendre en compte ce phénomène, nous continuons de calculer les échanges thermiques par convection tant que la température de la masse d'eau est supérieure à 275 K. La valeur du coefficient convectif est interpolé entre 277 K et 275 K pour prendre en compte le ralentissement des mouvements convectifs.

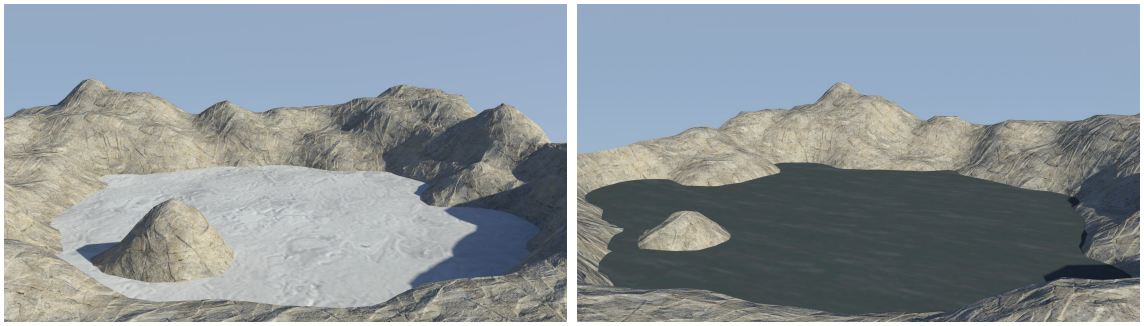


FIGURE 3.20 – Après deux jours et demi, la surface du plus petit lac est complètement gelée (gauche) tandis que le plus grand lac (droite) n'a toujours pas gelé à cause de l'inertie thermique.

La figure 3.20 illustre l'impact de l'inertie thermique engendré par la convection naturelle dans la simulation des échanges thermiques. Nous avons modélisé deux petits lacs de volume d'eau différents (respectivement $11m^3$ et $26m^3$) et simulé l'évolution

de la température et le gel de l'eau en utilisant le même scénario climatique. Plus précisément, les paramètres de la simulation ont été définis comme suit :

- les températures initiales du sol et de l'eau étaient respectivement de 279 K et 281 K ;
- la simulation du rayonnement solaire a été désactivée afin que le lac ayant une plus grande surface d'eau exposée ne reçoive pas plus d'énergie solaire que le plus petit lac ;
- la température de l'air T_{air} décroît linéairement de 277 K à 269 K sur une période de trois jours.

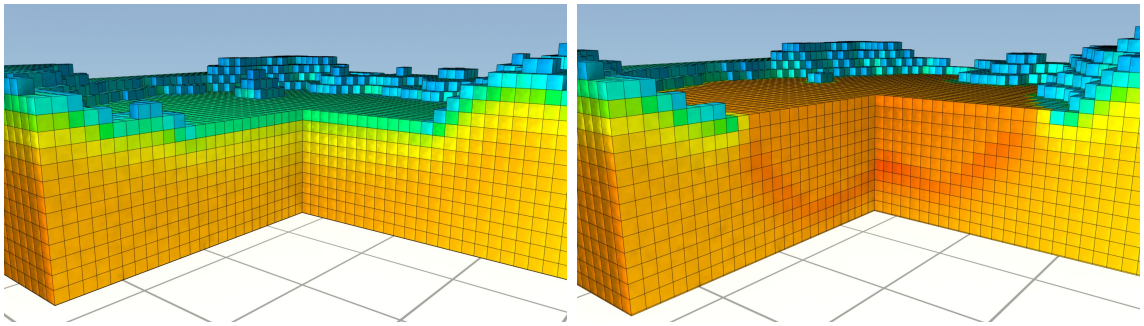


FIGURE 3.21 – Un gradient thermique apparaît dans le plus petit lac (gauche) lorsque la température passe en dessous de 277 K, tandis que le lac plus profond (droite) est toujours sous l'effet de la convection naturelle.

Après deux jours et demi, la surface du plus petit lac est complètement gelée, tandis que le plus grand lac n'a pas commencé à geler. La figure 3.21 illustrant la température des voxels du plus petit lac montre l'apparition d'un gradient thermique. Ceci signifie que la température de l'eau dans le lac est inférieure à 277 K. En revanche, la température du plus grand lac n'a pas atteint le seuil de 277 K. Ceci explique la température homogène de la masse d'eau qui subit des mouvements de convection naturelle.

L'air est également sujet au mouvement de convection. Dans notre implémentation, nous considérons la température de l'air comme constante et nous ne considérons pas l'influence du vent. Cependant, l'altitude joue un rôle important sur la température. Tous les 150 mètres, la température diminue de 1 K. Nous avons modélisé ce phénomène en utilisant une fonction permettant de convertir l'altitude des voxels en altitude réelle à l'aide des bornes minimum et maximum d'altitudes réelles. La dénivellation nous permet de déduire directement la diminution de la température. Nous appliquons cette diminution de température à celle de l'air et celle du point de rosée. Le scénario climatique définit ces températures à l'altitude minimale de la scène.

3.4.4 Conditions aux bords du domaine

En régime transitoire, lorsque la profondeur dans le sol augmente, l'amplitude des oscillations de température est de plus en plus amortie. Par conséquent, nous devons évaluer à quelle profondeur la température du sol est quasi-constante.

Une bonne approximation consiste à analyser les variations de la température dans le sol comme étant une fonction de la variation périodique de la température de l'air (CJ59). Considérons un plan infini séparant le sol et l'air. Considérons également que la variation de la température de l'air est définie par la fonction périodique $\delta T_{\text{air}}(t) = T_0 \cos(\omega\, t)$. Soit $\gamma = \lambda / \rho\, C_p$ la diffusivité thermique du sol. La variation de la température du sol s'écrit :

$$\delta T_{\text{ground}}(z,t) = T_0\, e^{-\sqrt{\frac{\omega}{2\gamma}}z} \cos\left(\omega\, t - \sqrt{\frac{\omega}{2\gamma}}z\right)$$

Étant donnée une variation maximum de température T_ϵ faible, la profondeur à laquelle la température du sol peut être considérée comme constante est définie par l'équation suivante :

$$z = -\ln\left(\frac{T_\epsilon}{T_0}\right)\sqrt{\frac{2\gamma}{\omega}}$$

La figure 3.22 illustre l'amortissement et le déphasage de la température dans un sol rocheux avec l'augmentation de la profondeur. L'amplitude des oscillations est de 20K et la période est de un an. Nous remarquons qu'au delà de 12 mètres, la température peut être considérée quasi-constante.

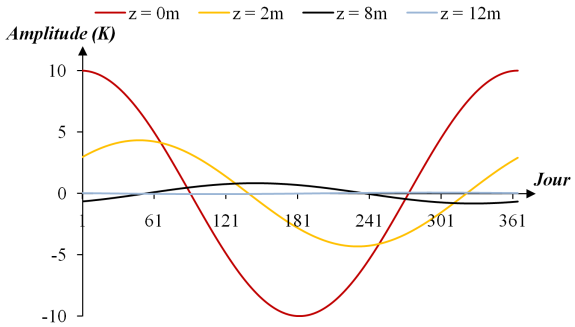


FIGURE 3.22 – Amortissement et déphasage de la température avec l'accroissement de la profondeur dans un sol rocheux.

Par conséquent, dans notre implémentation, nous ne simulons pas les échanges thermiques au-delà de cette profondeur. Nous appliquons donc des conditions de Neumann en fixant le flux à zéro en bas du domaine. Nous appliquons également des conditions de Neumann sur les quatre bords verticaux du domaine. La température de l'air imposée en haut du domaine correspond à une condition de Dirichlet.

3.4.5 Détails d'implémentation

Pour chaque voxel $\mathcal{V}_{ijk}$ dans la scène, nous stockons un identifiant correspondant au type de matériau (qui nous permet de récupérer les constantes thermiques du matériau), la température T et l'énergie totale δQ^n représentant la somme de tous les flux thermiques à chaque itération n de la simulation. Les voxels deux-matériaux, tels que les voxels eau-glace et air-neige, stockent également la quantité relative de matériaux sous forme de pourcentage du volume du voxel qui est nécessaire pour le calcul des changements d'état.

Le gradient spatial de température entre deux voxels voisins est calculé simplement par $\delta T/\delta x$ avec δx la résolution spatiale du voxel et δT la différence de température. La simulation globale est réalisée en utilisant le schéma d'intégration d'Euler explicite. Soit δt le pas de temps d'intégration. À chaque pas n et pour chaque voxel $\mathcal{V}_{ijk}$, nous calculons la somme de tous les flux de chaleur δQ^n et calculons la nouvelle température de la façon suivante :

$$T^{n+1} = T^n + \frac{\delta Q^n}{\rho\, C_p\, V}\delta t$$

Les simulations de paysages présentées dans la section 3.6 ont été conduites en utilisant une résolution spatiale de 1 m, tandis que les exemples techniques illustrant l'impact du rayonnement solaire et l'inertie thermique de l'eau, ont été réalisés avec une résolution spatiale de 10 cm. Pour des raisons de lisibilité, la grille de voxels des images illustrant leurs températures est à une résolution plus faible que la grille réelle.

Le pas de temps pour les simulations a été choisi en évaluant la raideur de l'équation. La raideur de l'équation de la conduction de la chaleur est très faible et permet l'utilisation d'un pas de temps très grand. Cependant, la boucle de simulation globale prend en compte beaucoup d'autres flux de chaleur qui ont un impact bien plus important. Nous avons expérimenté que l'énergie solaire, les échanges radiatifs avec la voûte céleste et les échanges convectifs sont les contributions majeures et donnent les valeurs de densité de flux de chaleur les plus importantes. Nos expérimentations numériques ont montré qu'un pas de temps d'intégration de 600 s est une borne supérieure de stabilité pour une résolution spatiale de 1 m. De plus, ce pas de temps nous permet de capturer les événements météorologiques de notre scénario climatique avec une précision suffisante.

	Temps (h)	Pas de temps (s)	#voxels (M)	#itérations
Pic rocheux	5	200	3.2	3888
Lake Louise	6	400	6.5	1943

TABLEAU 3.3 – Temps de calcul et nombre de voxels de nos simulations de paysages.

La résolution des équations sur une grille de voxels est très coûteuse en temps de calcul. Par conséquent, nous avons parallélisé notre implémentation pour profiter au

mieux des nouvelles technologies multi-cœur. Notre grille de voxels est découpée en sous-grilles qui sont ensuite stockées dans une file d'attente. La résolution des équations par les différents threads s'effectue sur chacune des sous-grilles. Lorsqu'un thread termine son exécution, celui-ci est réexécuté sur une autre sous-grille qui n'a pas encore été analysée et ce jusqu'à ce que la file d'attente soit vide. La table 3.3 présente les temps de calcul des simulations de paysages, la taille des grilles de voxels, les pas de temps d'intégration ainsi que le nombre d'itérations effectuées.

La figure 3.23 illustre l'interface de notre logiciel de simulation développé à l'aide de la librairie QT et d'OpenGL (gauche) ainsi que l'interface OpenGL d'édition du scénario climatique (droite).

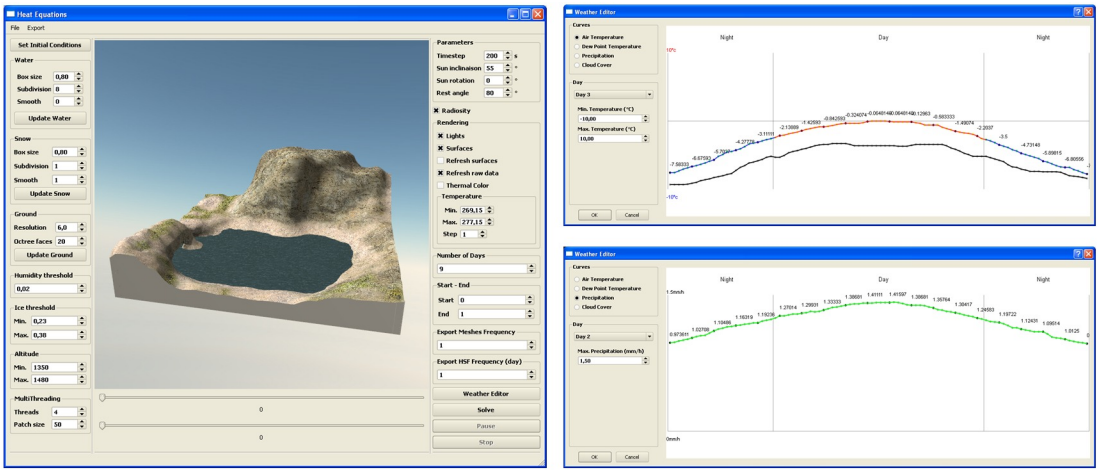


FIGURE 3.23 – Interfaces de simulation et d'édition du scénario climatique.

3.5 Instanciation

Le rendu réaliste de la neige et de la glace est un problème de recherche très complexe et ne sera pas abordé. Cependant, lors des rendus finaux, nous avons utilisé des BSSRDF conjointement au moteur de rendu Mental Ray®.

Dans cette section, nous présentons la méthode qui nous permet de générer une représentation sous forme de maillage texturé de l'eau, de la glace et de la neige (Figure 3.24). Notre approche consiste à générer trois maillages différents à partir du modèle voxel qui seront notés $\mathcal{M}_{\text{Eau}}$, $\mathcal{M}_{\text{Neige}}$ et $\mathcal{M}_{\text{Glace}}$ respectivement. Les transitions lisses entre l'eau et la glace sont obtenues par mélange des textures correspondantes en utilisant la quantité relative d'eau et de glace comme fonction de pondération.

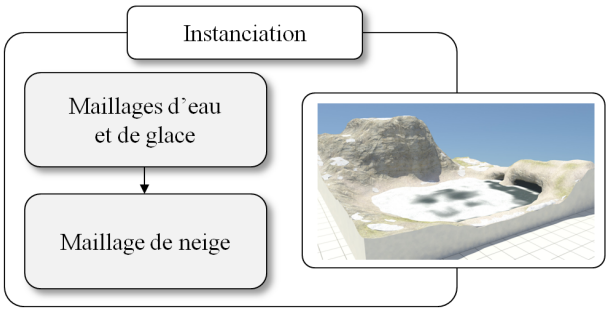


FIGURE 3.24 – Aperçu du processus d'instanciation.

3.5.1 Maillage de neige

Le maillage $\mathcal{M}_{\text{Neige}}$ représentant la couche de neige est généré comme étant un champ de hauteurs texturé au dessus du terrain (HAH02, FG09). Ce champ de hauteurs est obtenu par élévation des sommets $\mathbf{v}$ du maillage représentant la surface du terrain d'une valeur correspondant à la quantité de neige contenue dans les voxels situés au dessus de $\mathbf{v}$.

3.5.2 Maillages d'eau et de glace

La génération des maillages d'eau et de glace est plus complexe. La complexité réside dans la caractérisation de différents voxels d'eau, de glace et d'eau-glace impliqués. Notre méthode s'effectue en deux étapes. Nous générons tout d'abord un maillage générique $\mathcal{M}$ en calculant l'intersection de la surface correspondant aux voxels d'eau, de glace, et d'eau-glace avec le terrain. Ce maillage est ensuite décomposé en un maillage représentant l'eau $\mathcal{M}_{\text{Eau}}$ et un maillage représentant la glace $\mathcal{M}_{\text{Glace}}$ en analysant les caractéristiques des matériaux à chaque sommet.

Le maillage de glace est défini comme une combinaison de deux champs de hauteurs représentant le haut et le bas de la surface de la couche de glace.

Afin de caractériser la quantité relative d'eau et de glace à chaque sommet du maillage, nous calculons une fonction de potentiel de glace et une fonction de potentiel d'eau notées respectivement f_{Eau} et f_{Glace}. Ces fonctions de potentiel sont définies en utilisant le même type de fonctions de convolution que celles décrites par Peytavie et coll. (PGMG09) afin de lisser les valeurs discrètes de quantité d'eau et de glace stockées dans la représentation en voxels. Pour chaque sommet du maillage $\mathcal{M}$, nous calculons l'épaisseur de la couche de glace $\delta(\mathbf{v})$ en calculant l'intersection entre la surface implicite $f_{Glace}(\mathbf{x}) = 0$ et la droite verticale passant par le sommet $\mathbf{v}$.

Si $\delta(\mathbf{v}) = 0$, alors le sommet $\mathbf{v}$ appartient au maillage d'eau et est inchangé. Sinon, nous créons le sommet haut $\mathbf{p}$ et le sommet bas $\mathbf{q}$ de la surface de glace en déplaçant $\mathbf{v}$ verticalement. Rappelons que la masse volumique de la glace et celle de l'eau sont

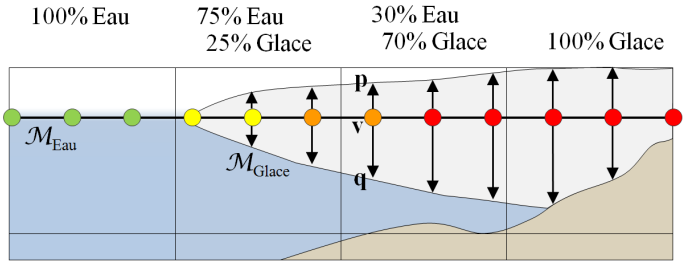


FIGURE 3.25 – Instanciation de l'eau et de la glace.

respectivement 915 et 999 kg m^{-3}. Par conséquent, 8.5% de la glace émerge de l'eau tandis que 91.5% est immergée. Nous approximons ce phénomène en définissant le sommet haut et le sommet bas respectivement par $\mathbf{p} = \mathbf{v} + 0.085\,\delta(\mathbf{v})$ et $\mathbf{q} = \mathbf{v} - 0.915\,\delta(\mathbf{v})$.

3.5.3 Texture

Le maillage d'eau est texturé en utilisant une texture d'eau. En revanche, la texture du maillage de glace est obtenue par mélange de deux textures d'eau et de glace. Le coefficient de mélange est défini comme étant une fonction de l'épaisseur de la couche de glace. Le degré d'humidité dans le sol est également utilisé pour définir le coefficient de mélange entre une texture de sol d'aspect sec et une d'aspect humide.

3.6 Résultats

Notre méthode a été implémentée en C++ sur une machine équipée d'un processeur Intel® Core™ 2 Quad 2.4GHz avec 3GB de RAM. L'implémentation a été parallélisée afin de profiter au mieux de la technologie multi-cœur. Les résultats démontrent que notre méthode permet de capturer des phénomènes naturels complexes et de générer des paysages hivernaux réalistes évoluant en fonction du climat.

3.6.1 Simulation de paysages

Dans la simulation suivante, nous avons élaboré un scénario climatique, sur une période de neuf jours, sur un paysage représentant un pic rocheux. La première moitié du scénario est une période de froid et de chute de neige, tandis que la seconde moitié est un réchauffement. Le temps de calcul de cette simulation est de 5 heures avec un pas de temps d'intégration de 200 s. La scène se compose de 3.2 millions de voxels. Une vue simplifiée du scénario climatique utilisé est donnée dans la figure 3.26.

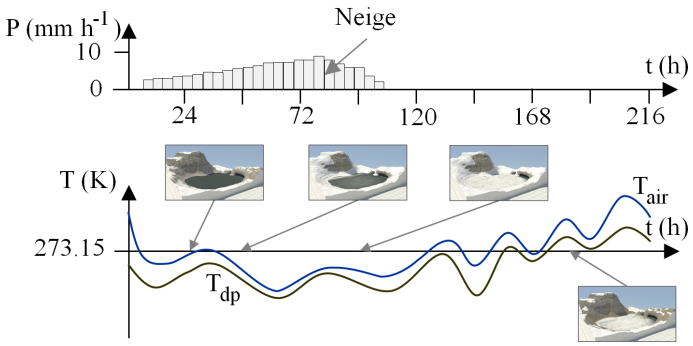


FIGURE 3.26 – Aperçu du scénario climatique.

Deux heures après le début de la chute de neige, la neige commence à s'accumuler sur le sol. La figure 3.27 illustre l'accumulation de la neige sur le sol (gauche), la propagation de la glace lors du gel du lac en commençant par les zones ombragées (milieu) et en finissant par le gel complet du lac.

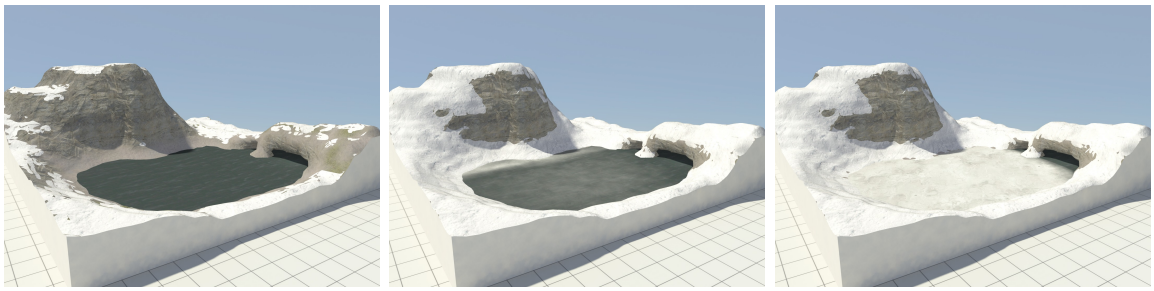


FIGURE 3.27 – Accumulation de la neige sur le sol et gel du lac avec une chute de température sous 273.15K.

La figure 3.28 (gauche) est un gros plan mettant en évidence le début de gel de l'eau dans l'ombre du pic rocheux. L'abri sous roche offre une protection à l'eau située juste en dessous et bloque les échanges radiatifs avec la voûte céleste ce qui empêche le gel de l'eau (Figure 3.28, droite).

La figure 3.29 illustre la fonte de la neige et de la glace. L'eau issue de la fonte de la neige s'infiltre dans le sol créant des zones humides (gauche et milieu). Avec le temps, l'eau s'évapore redonnant un aspect sec au sol (droite).

La figure 3.30 (gauche) illustre le changement d'apparence du sol dû à l'infiltration et à la diffusion de l'eau. Après évaporation, le sol retrouve son aspect sec (Figure 3.30, droite). La simulation de la diffusion de l'eau dans le sol permet de faire apparaître une bande d'humidité autour des plaques de neige qui subsistent.

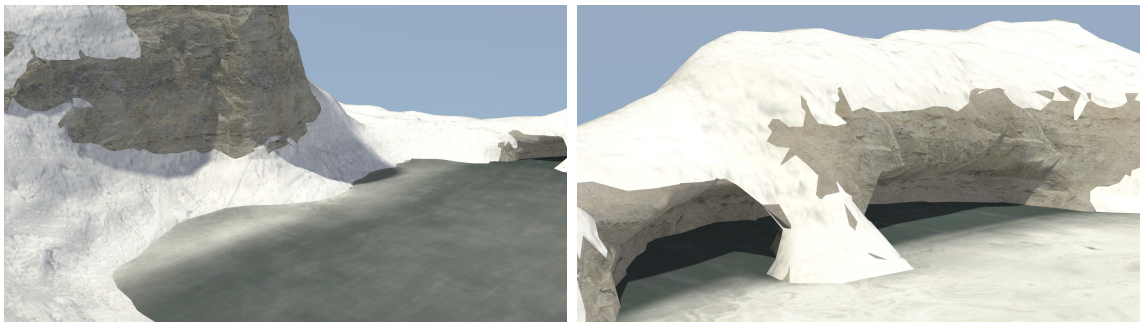


FIGURE 3.28 – Début de gel dans l'ombre du pic rocheux (gauche) et blocage des échanges radiatifs avec la voûte céleste empêchant l'eau de geler (droite).

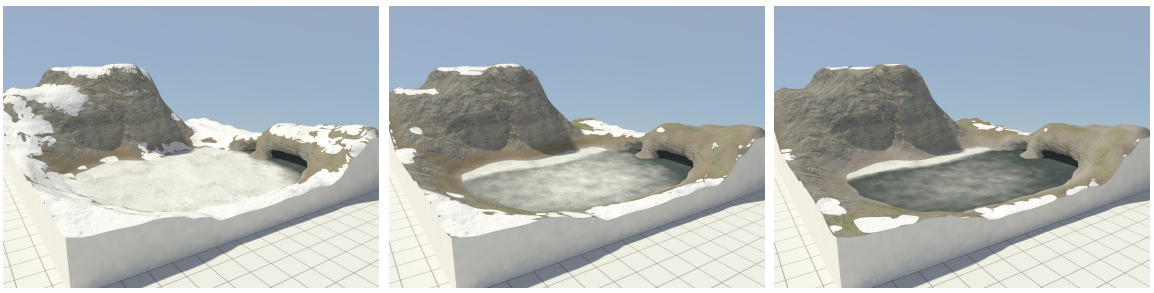


FIGURE 3.29 – Fonte de la neige et de la glace ; l'eau s'infiltre dans le sol puis s'évapore.

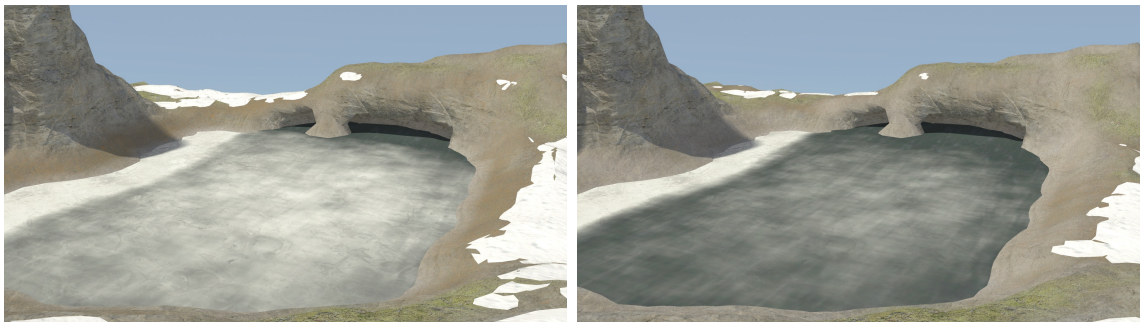


FIGURE 3.30 – Mise en évidence de l'aspect humide du sol après infiltration de l'eau issue de la fonte de la neige (gauche). Après évaporation, le sol retrouve un aspect sec (droite).

3.6.2 Paysages issus de Modèles Numériques de Terrain

Nous avons modélisé les alentours de Lake Louise (Alberta, Canada) ainsi que le lac à partir d'un modèle numérique de terrain issu de GéoBase [1]. La simulation des échanges thermiques a été réalisée à l'aide d'un scénario climatique de neuf jours reproduisant les conditions météorologiques de la région. Le pas de temps d'intégration est de 400 s et le nombre de voxels est de 6.5 millions. La figure 3.31 illustre le début du gel du lac dans la région qui est le plus souvent dans l'ombre des montagnes environnantes.

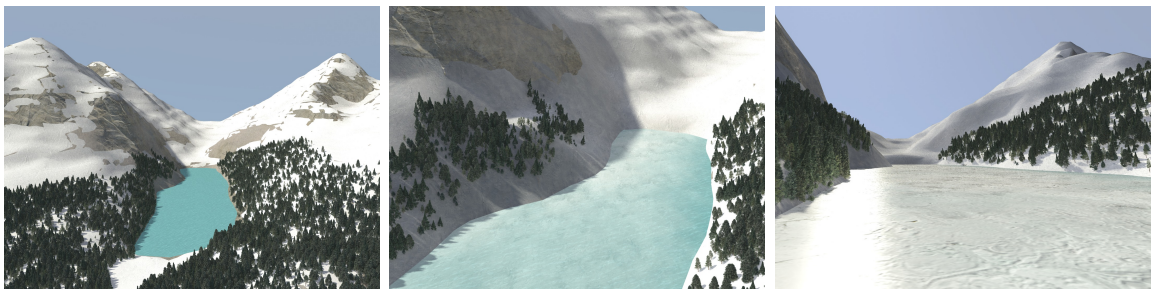


FIGURE 3.31 – Le gel du Lac Louise en hiver.

La figure 3.32 illustre le dégel du lac et la fonte de la neige. La partie droite du lac est la première à dégeler car elle est la plus exposée aux rayons du soleil (droite et milieu). De la glace persiste dans la région située à l'ombre des montagnes voisines (gauche). L'influence de l'altitude sur les températures de l'air et de point de rosée laisse apparaître très nettement la frontière entre les neiges éternelles et éphémères.

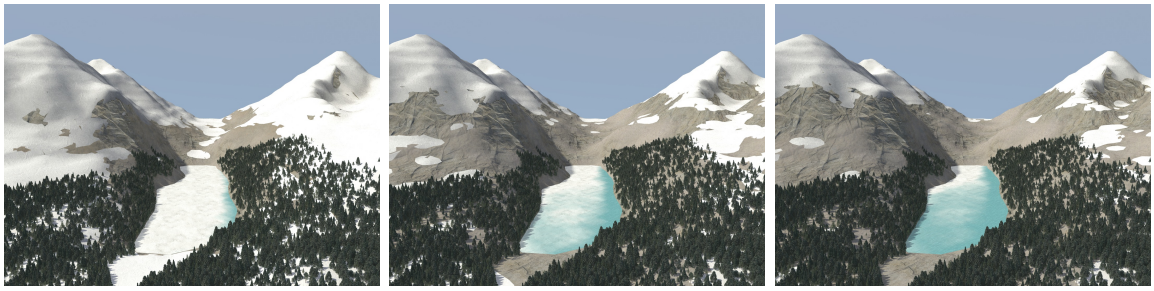


FIGURE 3.32 – Le dégel du Lac Louise et l'apparition des neiges éternelles.

3.7 Conclusion

Dans ce chapitre, nous avons proposé une simulation complète des échanges thermiques présents dans la nature pour créer des paysages hivernaux. Les résultats obtenus illustrent l'importance de prendre en compte les échanges thermiques ainsi que les

1. www.geobase.ca

conditions climatiques pour suivre l'évolution du manteau neigeux et des couches de glace. Notre méthode est un premier pas vers une simulation globale de l'ensemble des phénomènes qui existent en hiver. Cependant, la simulation thermique et le scénario climatique peuvent être utilisés dans d'autres contextes que celui des paysages hivernaux. Par exemple, elle permettrait de simuler l'impact des changements climatiques sur les écosystèmes ainsi que sur le vieillissement des objets.

Notre méthode peut être améliorée en de nombreux points pour obtenir une simulation globale de paysages hivernaux. La présence d'arbres agit comme un filtre sur les échanges thermiques, leur prise en compte est donc importante. L'instanciation rapide et efficace de la neige sur ce type d'objets est un problème très difficile et nécessite une étude particulière. Des ponts de neige pouvant se former entre les branchages et les épines des sapins, une approche utilisant des surfaces implicites serait certainement la plus adaptée.

Simuler le vent produirait une distribution de neige plus réaliste. Le vent provoque la formation de congères, érode la neige, change ses propriétés thermiques et visuelles et peut faire sublimer la neige. L'évaporation de l'eau serait également beaucoup plus précise. De plus, sous l'effet de son propre poids, la neige se compacte et les changements de température au sein du manteau ont tendance à la convertir en glace. Des croûtes de glace peuvent se former en surface. Des méthodes de rendu sont également à développer pour prendre en compte ces changements d'apparence.

La simulation des courants dans des masses d'eau permettrait une simulation plus précise de la convection naturelle. D'autre part, en simulant l'écoulement de l'eau sur le sol, il serait possible de créer des rivières de glace, des glaciers, des stalactites et des cascades de glace. Pour finir, la simulation de la formation des fissures sur la couche de glace améliorerait de façon significative son aspect visuel.

Conclusion et perspectives

Dans cette thèse, nous nous sommes intéressés aux problèmes du réalisme pour la création de scènes de synthèse. Le réalisme provient de la diversité de formes et d'apparences des objets et des paysages ainsi que de l'accumulation de détails. Nous avons donc abordé le problème du réalisme en proposant des méthodes permettant de générer de la variété de contenu graphique statique, adaptative et dynamique. Les résultats obtenus démontrent l'importance d'introduire de la diversité dans les scènes pour accroître leurs réalismes. Chacune des approches que nous avons utilisé possède ses avantages et inconvénients.

Notre méthode à base d'exemples permet de générer rapidement des variétés de contenu statique à partir d'un objet initial. Le faible surcoût mémoire engendré rend notre approche particulièrement bien adaptée aux contraintes des consoles de jeux vidéo. De plus, l'interface proposée est intuitive pour un artiste et il peut imposer son style graphique en toute liberté. La réduction du temps nécessaire pour modéliser des milliers d'objets est un gain important pour les sociétés du loisir numérique et leur permet de réduire leur coût de production. Finalement, les résultats montrent que l'introduction de variété brise les répétitions provoquées par l'instanciation massive des objets et accroît considérablement le réalisme des scènes. Néanmoins, il est nécessaire d'avoir des modèles initiaux pour pouvoir utiliser notre méthode. D'autre part, les formes des objets générés dépendent de la forme des objets initiaux. Une interface de modélisation des fragments permettrait de créer librement de nouvelles formes sans être limité par la forme des objets initialement fournis. Cette interface doit être développée en fonction des besoins des artistes. Il est donc important d'avoir leur retour d'expérience lors de sa conception. Finalement, le contenu statique généré en utilisant cette approche n'est pas très adapté à la génération d'objets manufacturés.

L'utilisation de modèles paramétrés est beaucoup plus adaptée à ce type d'objets. L'approche procédurale utilisée permet de générer rapidement des variétés d'objets en modifiant quelques paramètres. De plus, il est possible d'adapter la géométrie en fonction de contraintes extérieures tel que le relief d'un paysage. Les résultats obtenus en utilisant les modèles paramétrés pour la génération de routes démontrent la puissance de ce type d'approche. Dès lors que la trajectoire de la route ou que le relief du pay-

sage sont modifiés, les modèles s'adaptent automatiquement aux nouvelles contraintes. L'artiste n'a donc plus besoin de re-modéliser l'intégralité des ouvrages. Le gain de temps est considérable. Toutefois, l'inconvénient majeur de ce type d'approche est que les artistes n'ont pas les notions de programmation nécessaires pour créer de nouveaux modèles. Leur formation devrait être adaptée pour qu'ils puissent intégrer les concepts indispensables. Ceci leur permettrait de ne plus avoir à créer les objets un par un et de modéliser directement des variétés de contenu adaptatif. À plus court terme, une solution envisagée serait de créer une interface intuitive permettant de mettre en place rapidement des modèles paramétrés quelconques. Cependant, dans certains cas, il ne sera pas possible de créer facilement le modèle souhaité. Le programmer directement serait beaucoup plus simple.

La diversité des objets et des paysages est également le résultat de phénomènes complexes altérant et modifiant leur apparence visuelle au cours du temps. Pour modéliser ces changements d'apparence, l'approche généralement utilisée est la simulation. À l'échelle d'une année, les paysages subissent des changements engendrés par l'évolution des conditions climatiques. Nous nous sommes intéressés à ce type de changement en proposant une méthode permettant de simuler l'évolution de paysages hivernaux en fonction des conditions climatiques. L'approche que nous avons utilisé consiste à simuler les échanges thermiques, les changements d'état et la convection naturelle. Notre objectif était de suivre l'évolution du manteau neigeux et de la couche de glace se formant en surface des masses d'eau. Le contenu graphique que nous générons est dynamique. Afin de contrôler notre simulation, nous avons mis en place un scénario climatique décrivant de façon complète le climat. Notre méthode est un premier pas vers une simulation de paysages plus complète. En effet, cette approche peut être utilisée pour simuler l'impact des changements météorologiques sur les écosystèmes et le vieillissement des objets.

De nombreuses améliorations peuvent être apportées à notre méthode. Tout d'abord, la simulation du vent permettrait de simuler la formation de congères et l'érosion de la neige. Nous pourrions également améliorer l'évaporation de l'eau. En simulant l'écoulement de l'eau, nous pourrions prendre en compte son ruissellement sur le sol lors de la fonte de la neige et ainsi simuler la formation de stalactites et de cascades de glace. Ceci permettrait également de créer les torrents de montagnes. Simuler l'apparition des fissures améliorerait très nettement l'aspect visuel de la couche de glace. Au cours du temps, la neige et la glace changent d'apparence. Sous l'effet de la chaleur, elles peuvent fondre et geler plusieurs fois. Une méthode de rendu doit être mise en œuvre pour prendre en compte ce changement d'aspect. Il serait intéressant de prendre en compte la présence des arbres dans la simulation. Les arbres peuvent être considérés comme des filtres modifiant les échanges thermiques. De plus l'instanciation de la neige sur ce type d'objets est difficile. Déplacer verticalement la surface ne suffirait pas. L'utilisation de surfaces implicites permettrait de résoudre le problème de formation des ponts de neige entre les branches et les aiguilles des sapins. D'autre part, modéliser les traces d'animaux et de skis permettrait d'avoir de la neige beaucoup plus réaliste. Finalement, simplifier la simulation en deux dimensions ou par des fonctions procédurales accélérerait considérablement les temps de calcul. De plus, nous pourrions traiter

des paysages très grands.

Pour conclure cette thèse, l'idéal serait de concevoir un système générique permettant de créer rapidement des variétés d'objets et de paysages évoluant au cours du temps. Par exemple, le système devra être capable de générer tout autant des arbres que des paysages aux fils des saisons et en fonction des conditions climatiques. Les variétés seront obtenues en modifiant quelques paramètres tels que l'âge ou le jour dans l'année. Le climat devra être pris en compte pour pouvoir faire apparaître de la neige ou des flaques d'eau. Le vieillissement pourra être simulé pour altérer les objets en ajoutant des impacts, des fractures, de la mousse ou encore de la rouille.

Annexes

Annexe A

Calcul de l'angle d'incidence des rayons solaires

L'angle d'incidence θ du rayonnement solaire atteignant une surface inclinée est l'angle entre la direction du rayonnement sur la surface et la normale à cette surface. Il dépend de nombreux autres angles (Figure A.1) :

- la **latitude** ϕ qui est la localisation géographique angulaire Nord ou Sud d'un point sur la terre par rapport au plan équatorial. Au Nord l'angle est positif ; $\phi \in [-\pi/2, \pi/2]$.
- la **longitude** φ représentant la localisation géographique angulaire Est-Ouest d'un point sur la terre (Est négatif, Ouest positif). Le méridien de Greenwich étant la référence ($\varphi = 0$).
- la **déclinaison** δ correspondant à l'angle entre le vecteur ayant pour origine le centre de la terre et pour direction le soleil et le plan équatorial ; $\delta \in [-0.4092, 0.4092]$.
- l'**inclinaison de la surface** β par rapport au plan horizontal ; $\beta \in [0, \pi]$. Si $\beta > \pi/2$, la surface est inclinée vers le bas.
- l'**angle d'azimut de la surface** γ qui correspond à l'angle entre la projection de la normale sur le plan horizontal et le méridien local (axe Sud-Nord) mesuré dans le sens des aiguilles d'une montre à partir du Nord.
- l'**angle horaire** ω est l'angle entre le vecteur ayant pour origine le centre de la terre et pour direction le soleil projeté sur le plan équatorial et le méridien local. Cet angle est négatif avant le midi solaire (soleil au zénith) et positif après.

Soit n le jour de l'année (au 1$^{\text{er}}$ janvier, $n = 1$), la déclinaison est calculée à l'aide de l'équation de Cooper :

$$\delta = 0.4092 \sin\left(2\pi \frac{284 + n}{365}\right)$$

L'angle horaire ω est calculé en fonction du temps solaire T_s lui même calculé à partir du temps solaire moyen T_m et de l'équation du temps e. Le temps solaire T_s

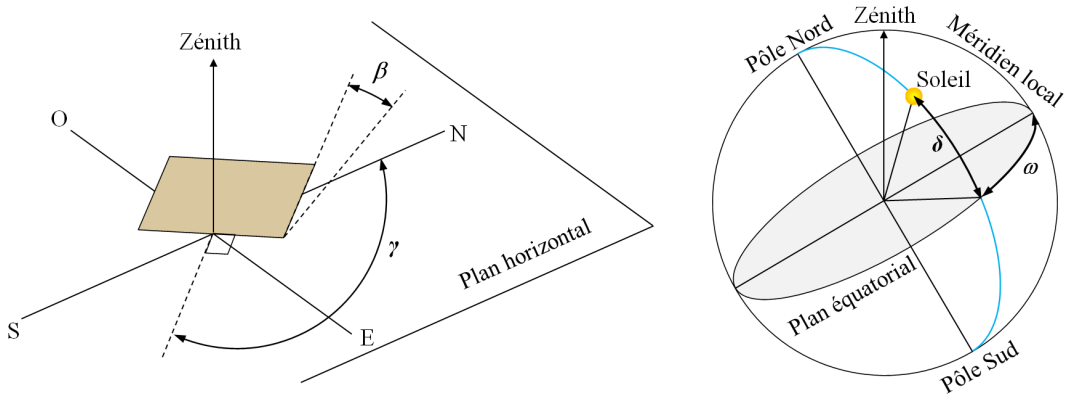


FIGURE A.1 – Angles permettant de calculer l'angle d'incidence du rayonnement solaire.

est la durée entre deux retours successifs du soleil au méridien local (soit au zénith). Le temps solaire moyen T_m est basé sur un soleil moyen fictif se déplaçant à vitesse constante tout au long de l'année. Quel que soit le jour de l'année, la durée d'un jour solaire moyen est de 24 heures. L'équation du temps e (en minutes) représente la différence entre le temps solaire moyen T_m et le temps solaire T_s. Soit T_u l'heure en temps universel, φ la longitude. Le temps solaire moyen s'exprime de la façon suivante :

$$T_m = T_u + \frac{\varphi}{0.2617}$$

L'équation du temps (Figure A.2) est définie par la relation suivante :

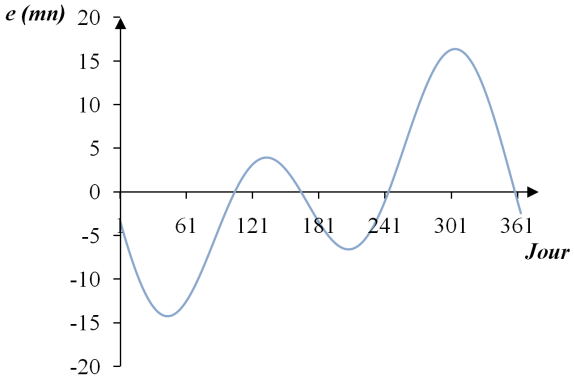


FIGURE A.2 – Évolution de l'équation du temps au cours d'une année.

$$\begin{aligned} e &= 229.2\,(0.000075 + 0.001868\,\cos\eta - 0.032077\,\sin\eta \\ &- 0.014615\,\cos 2\eta - 0.04089\,\sin 2\eta) \end{aligned}$$

avec

$$\eta = \frac{2\pi\, n}{365}$$

Le temps solaire T_s est alors calculé en utilisant la relation suivante :

$$T_s = T_m + \frac{e}{60}$$

L'angle horaire ω est finalement déduit en utilisant l'équation suivante :

$$\omega = \frac{2\pi\,(T_s - 12)}{24}$$

Le cosinus de l'angle d'incidence θ peut finalement être calculé par :

$$\begin{aligned}\cos\theta &= \sin\delta\sin\phi\cos\beta + \sin\delta\cos\phi\sin\beta\cos\gamma + \cos\delta\cos\phi\cos\beta\cos\omega \\ &- \cos\delta\sin\phi\sin\beta\cos\gamma\cos\omega - \cos\delta\sin\beta\sin\gamma\sin\omega\end{aligned}$$

Si θ est supérieur à 90° alors la surface n'est pas exposée au rayonnement solaire.

Annexe B

Calcul de la position du soleil

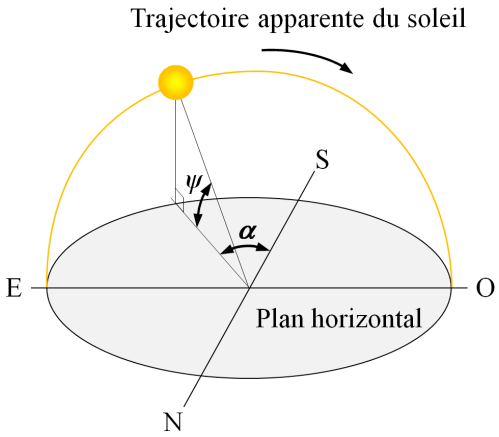


FIGURE B.1 – Angles définissant la position du soleil dans le ciel.

La position du soleil dans le ciel est exprimée à l'aide de deux angles : l'altitude solaire ψ et l'azimut solaire α (Figure B.1). L'altitude solaire est l'angle entre le plan horizontal et le vecteur ayant pour origine le centre de la scène et pour direction le soleil. Il est calculé en fonction de la latitude ϕ, de la déclinaison δ et de l'angle horaire ω par l'équation suivante :

$$\psi = \sin^{-1}(\sin\phi\sin\delta + \cos\phi\cos\delta\cos\omega)$$

L'azimut solaire est l'angle entre le méridien local et la projection du vecteur ayant pour origine le centre de la scène et pour direction le soleil (négatif à l'Est et positif à l'Ouest). Il est calculé à l'aide de la déclinaison δ, de l'angle horaire ω et de l'altitude solaire ψ. Selon le signe de l'angle horaire ω, l'azimut solaire est calculé comme suit : si $\omega < 0$ alors

$$\alpha = -\pi + \cos^{-1}\left(\frac{\sin\psi\sin\phi - \sin\delta}{\cos\psi\cos\phi}\right)$$

sinon

$$\alpha = \pi - \cos^{-1}\left(\frac{\sin\psi\sin\phi - \sin\delta}{\cos\psi\cos\phi}\right)$$

La figure B.2 illustre l'évolution de l'altitude solaire en fonction de l'azimut solaire à Lake Louise ($\phi = 51.41$, $\varphi = 116.18$). Les courbes représentent le mouvement apparent du soleil dans le ciel pour différentes dates de l'année.

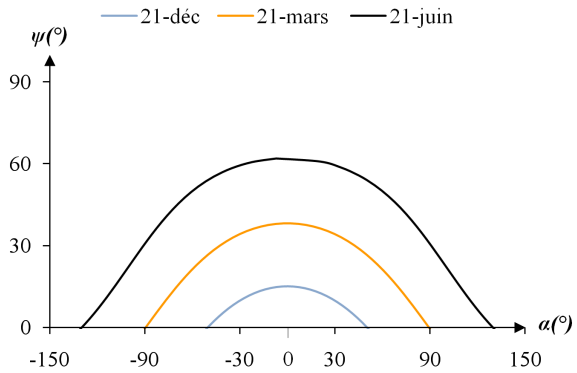


FIGURE B.2 – Mouvement apparent du soleil dans le ciel.

Annexe C

Calcul de l'heure du lever et du coucher du soleil

Les heures du lever et du coucher du soleil en temps solaire sont calculées à l'aide de la latitude ϕ et de la déclinaison δ. L'heure du lever R et l'heure du coucher S du soleil sont définies respectivement par les relations suivantes :

$$R = 12 - \frac{\cos^{-1}(-\tan\phi\tan\delta)}{0.2617}$$

$$S = 12 + \frac{\cos^{-1}(-\tan\phi\tan\delta)}{0.2617}$$

La figure C.1 illustre l'évolution de l'heure du lever et du coucher du soleil en temps solaire au cours de l'année à Lake Louise.

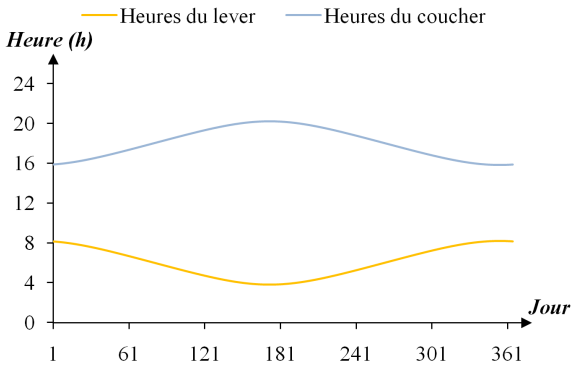


FIGURE C.1 – Heures du lever et du coucher du soleil au cours de l'année (Lake Louise).

Bibliographie

[ACOL00] M. Alexa, D. Cohen-Or, and D. Levin. As-rigid-as-possible shape interpolation. In *Proceedings of ACM SIGGRAPH*, pages 157–164, 2000.

[AD05] M. Alsweis and O. Deussen. Modeling and visualization of symmetric and asymmetric plant competition. In *Eurographics Workshop on Natural Phenomena*, pages 83–88, 2005.

[AKM+06] M. Attene, S. Katz, M. Mortara, G. Patane, M. Spagnuolo, and A. Tal. Mesh segmentation - a comparative study. In *Proceedings of the IEEE International Conference on Shape Modeling and Applications*, page 7, 2006.

[ARB07] D. G. Aliaga, P. A. Rosen, and D. R. Bekins. Style grammars for interactive visualization of architecture. *IEEE Transactions on Visualization and Computer Graphics*, 13(4) :786–797, 2007.

[AVB08] D. G. Aliaga, C. A. Vanegas, and B. Beneš. Interactive example-based urban layout synthesis. *ACM Transactions on Graphics*, 27(5) :160 :1–10, 2008.

[BA05a] F. Belhadj and P. Audibert. Modeling landscapes with ridges and rivers. In *Proceedings of the ACM symposium on Virtual reality software and technology*, pages 151–154, 2005.

[BA05b] F. Belhadj and P. Audibert. Modeling landscapes with ridges and rivers : bottom up approach. In *Proceedings of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, pages 447–450, 2005.

[BAv09] B. Beneš, N. Andrysco, and O. Šťava. Interactive modeling of virtual ecosystems. In *Eurographics Workshop on Natural Phenomena*, pages 9–16, 2009.

[BCS03] B. Beneš, J. A. Cordóba, and J. M. Soto. Modeling virtual gardens by autonomous procedural agents. In *Proceedings of Theory and Practice of Computer Graphics*, pages 58–66, 2003.

[Bel07] F. Belhadj. Terrain modeling : a constrained fractal model. In *Proceedings of the 5th international conference on Computer graphics, virtual reality, visualisation and interaction in Africa*, pages 197–204, 2007.

[BPF+03] F. Boudon, P. Prusinkiewicz, P. Federl, C. Godin, and R. Karwowski. Interactive design of bonsai tree models. *Computer Graphics Forum*, 22(3) :591–599, 2003.

[BTHB06] B. Beneš, V. Těšínský, J. Hornyš, and S. K. Bhatia. Hydraulic erosion. *Computer Animation and Virtual Worlds*, 17(2) :99–108, 2006.

[CC75] S. W. Churchill and H. H. S. Chu. Correlating equations for laminar and turbulent free convection from a vertical plate. *Int. J. Heat Mass Transfer*, 18 :1323–1329, 1975.

[CD05] R. Cook and T. DeRose. Wavelet noise. *ACM Transaction on Graphics*, 24(3) :803–811, 2005.

[CEW+08] G. Chen, G. Esch, P. Wonka, P. Müller, and E. Zhang. Interactive procedural street modeling. *ACM Transaction on Graphics*, 27(3) :103 :1–10, 2008.

[Chu76] S. W. Churchill. A comprehensive correlating equation for forced convection from flat plates. *AIChE J.*, 22 :264–268, 1976.

[CHZ00] J. M. Cohen, J. F. Hughes, and R. C. Zeleznik. Harold : a world made of drawings. In *Proceedings of the 1st international symposium on Non-photorealistic animation and rendering*, pages 83–90, 2000.

[CJ59] H. S. Carslaw and J. C. Jaeger. *Conduction of heat in solids,* 2nd *Edition.* Oxford University Press, 1959.

[CKX+08] X. Chen, S. B. Kang, Y. Xu, J. Dorsey, and H. Shum. Sketching reality : Realistic interpretation of architectural designs. *ACM Transactions on Graphics*, 27(2) :11 :1–15, 2008.

[CLDD09] M. Cabral, S. Lefebvre, C. Dachsbacher, and G. Drettakis. Structure-preserving reshape for textured architectural scenes. *Computer Graphics Forum*, 28(2) :469–480, 2009.

[CNX+08] X. Chen, B. Neubert, Y.-Q. Xu, O. Deussen, and S. B. Kang. Sketch-based tree modeling using markov random field. *ACM Transactions on Graphics*, 27(5) :109 :1–9, 2008.

[CXW+05] Y. Chen, L. Xia, T.-T. Wong, X. Tong, H. Bao, B. Guo, and H.-Y. Shum. Visual simulation of weathering by $\gamma - ton$ tracing. *ACM Transactions on Graphics*, 24(3) :1127–1133, 2005.

[DB91] J. A. Duffie and W. A. Beckman. *Solar Engineering of Thermal Processes,* 2^{nd} *Edition*. John Wiley & Sons, 1991.

[DEJ+99] J. Dorsey, A. Edelman, H. W. Jensen, J. Legakis, and H. K. Pedersen. Modeling and rendering of weathered stone. In *Proceedings of ACM SIGGRAPH*, pages 225–234, 1999.

[DGA04] B. Desbenoit, E. Galin, and S. Akkouche. Simulating and modeling lichen growth. *Computer Graphics Forum*, 23(3) :341–350, 2004.

[DHL+98] O. Deussen, P. Hanrahan, B. Lintermann, R. Měch, M. Pharr, and P. Prusinkiewicz. Realistic modeling and rendering of plant ecosystems. In *Proceedings of ACM SIGGRAPH*, pages 275–286, 1998.

[DL97] O. Deussen and B. Lintermann. A modelling method and user interface for creating plants. In *Proceedings of Graphics Interface*, pages 189–197, 1997.

[DPH96] J. Dorsey, H. K. Pedersen, and P. Hanrahan. Flow and changes in appearance. In *Proceedings of ACM SIGGRAPH*, pages 411–420, 1996.

[EF01] A. A. Efros and W. T. Freeman. Image quilting for texture synthesis and transfer. In *Proceedings of ACM SIGGRAPH*, pages 341–346, 2001.

[EMI01] C. Erikson, D. Manocha, and W. V. Baxter III. Hlods for faster display of large static and dynamic environments. In *Proceedings of the symposium on Interactive 3D graphics*, pages 111–120, 2001.

[FB07] D. Foldes and B. Beneš. Occlusion-based snow accumulation simulation. In *The Fourth Workshop on Virtual Reality Interactions and Physical Simulation*, pages 35–41, 2007.

[Fea00] P. Fearing. Computer modeling of fallen snow. In *Proceedings of ACM SIGGRAPH*, pages 37–46, 2000.

[FFC82] A. Fournier, D. Fussell, and L. Carpenter. Computer rendering of stochastic models. *Communication of the ACM*, 25(6) :371–384, 1982.

[FG09] N. Festenberg and S. Gumhold. A geometric algorithm for snow distribution in virtual scenes. In *Eurographics Workshop on Natural Phenomena*, pages 17–25, 2009.

[FKS+04] T. Funkhouser, M. Kazhdan, P. Shilane, P. Min, W. Kiefer, A. Tal, S. Rusinkiewicz, and D. Dobkin. Modeling by example. *ACM Transactions on Graphics*, 23(3) :652–663, 2004.

[FO02] B. E. Feldman and J. F. O'Brien. Modeling the accumulation of wind-driven snow. In *Proceedings of ACM SIGGRAPH*, pages 218–218, 2002.

[FP04] P. Federl and P. Prusinkiewicz. Finite element model of fracture formation on growing surfaces. *Lecture Notes in Computer Science*, pages 3037 :138–145, 2004.

[GM01] M. Gamito and F. K. Musgrave. Procedural landscapes with overhangs. In *10th Portuguese Computer Graphics Meeting*, pages 33–42, 2001.

[GMe09] J. Gain, P. Marais, and W. Straßer. Terrain sketching. In *Proceedings of the symposium on Interactive 3D graphics and games*, pages 31–38, 2009.

[GPMG10] E. Galin, A. Peytavie, N. Maréchal, and E. Guérin. Procedural generation of roads. *Computer Graphics Forum*, 29(2) :429–438, 2010.

[GTR+06] J. Gu, C.-I Tu, R. Ramamoorthi, P. Belhumeur, W. Matusik, and S. Nayar. Time-varying surface appearance : acquisition, modeling and rendering. *ACM Transactions on Graphics*, 25(3) :762–771, 2006.

[HAH02] H. Håglund, M. Anderssonand, and A. Hast. Snow accumulation in real-time. In *Proceedings of SIGRAD*, 2002.

[HTK98] K. Hirota, Y. Tanoue, and T. Kaneko. Generation of crack patterns with a physical model. *The Visual Computer*, 14(3) :126–137, 1998.

[HTK00] K. Hirota, Y. Tanoue, and T. Kaneko. Simulation of three-dimensional cracks. *The Visual Computer*, 16(7) :371–378, 2000.

[IOI06] T. Ijiri, S. Owada, and T. Igarashi. Seamless integration of initial sketching and subsequent detail editing in flower modeling. *Computer Graphics Forum*, 25(3) :617–624, 2006.

[IOOI05] T. Ijiri, S. Owada, M. Okabe, and T. Igarashi. Floral diagrams and inflorescences : interactive flower modeling using botanical structural constraints. *ACM Transaction on Graphics*, 24(3) :720–726, 2005.

[KAL06] T. Kim, D. Adalsteinsson, and M. Lin. Modeling ice dynamics as a thin-film stefan problem. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 167–176, 2006.

[KBKv09] P. Krištof, B. Beneš, J. Křivánek, and O. Šťava. Hydraulic erosion using smoothed particle hydrodynamics. *Computer Graphics Forum*, 28(2) :219–228, 2009.

[KHL04] T. Kim, M. Henson, and M. Lin. A hybrid algorithm for modeling ice formation. In *Proceedings of the ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 305–314, 2004.

[KL03] T. Kim and M. Lin. Visual simulation of ice crystal growth. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 86–97, 2003.

[LDSS99] A. W. F. Lee, D. Dobkin, W. Sweldens, and P. Schröder. Multiresolution mesh morphing. In *Proceedings of ACM SIGGRAPH*, pages 343–350, 1999.

[LGG+07] J. Lu, A. S. Georghiades, A. Glaser, H. Wu, L.-Y. Wei, B. Guo, J. Dorsey, and H. Rushmeier. Context-aware textures. *ACM Transactions on Graphics*, 26(1) :3 :1–22, 2007.

[LILV08] J. H. Lienhard *IV* and J. H. Lienhard *V*. *A Heat Transfer Textbook,* 3rd *Edition*. Phlogiston Press, Cambridge, Ma, 2008.

[Lin68] A. Lindenmayer. Mathematical models for cellular interactions in development, I & II. *Journal of Theoretical Biology*, pages 280–315, 1968.

[LP02] B. Lane and P. Prusinkiewicz. Generating spatial distributions for multilevel models of plant communities. In *Proceedings of Graphics Interface*, pages 69–80, 2002.

[LPRM02] B. Lévy, S. Petitjean, N. Ray, and J. Maillot. Least squares conformal maps for automatic texture atlas generation. *ACM Transactions on Graphics*, 21(3) :362–371, 2002.

[LWW08] M. Lipp, P. Wonka, and M. Wimmer. Interactive visual editing of grammars for procedural architecture. *ACM Transactions on Graphics*, 27(3) :102 :1–10, 2008.

[Man82] B. B. Mandelbrot. *The Fractal Geometry of Nature*. W. H. Freeman & Co Ltd, 1982.

[MBF04] N. Molino, Z. Bao, and R. Fedkiw. A virtual node algorithm for changing mesh topology during simulation. *ACM Transactions on Graphics*, 23(3) :385–392, 2004.

[MC00] K. Muraoka and N. Chiba. Visual simulation of snowfall, snow cover and snowmelt. In *Proceedings of the Seventh International Conference on Parallel and Distributed Systems*, pages 187–194, 2000.

[MDG01] S. Mérillou, J.-M. Dischler, and D. Ghazanfarpour. Corrosion : simulating and rendering. In *Proceedings of Graphics interface*, pages 167–174, 2001.

[MDH07] X. Mei, P. Decaudin, and B.-G. Hu. Fast hydraulic erosion simulation and visualization on GPU. In *Proceedings of Pacific Graphics*, pages 47–56, 2007.

[Mer07] P. Merrell. Example-based model synthesis. In *Proceedings of the Symposium on Interactive 3D graphics and games*, pages 105–112, 2007.

[MG08] S. Mérillou and D. Ghazanfarpour. A survey of aging and weathering phenomena in computer graphics. *Computers & Graphics*, 32(2) :159 – 174, 2008.

[Mil86] G. S. P. Miller. The definition and rendering of terrain maps. In *Proceedings of ACM SIGGRAPH*, pages 39–48, 1986.

[MKM89] F. K. Musgrave, C. E. Kolb, and R. S. Mace. The synthesis and rendering of eroded fractal terrains. In *Proceedings of ACM SIGGRAPH*, pages 41–50, 1989.

[MM08] P. Merrell and D. Manocha. Continuous model synthesis. *ACM Transactions on Graphics*, 27(5) :158 :1–7, 2008.

[MMDJ01] M. Müller, L. McMillan, J. Dorsey, and R. Jagnow. Real-time simulation of deformation and fracture of stiff materials. In *Proceedings of the Eurographic workshop on Computer animation and simulation*, pages 113–124, 2001.

[MP96] R. Měch and P. Prusinkiewicz. Visual models of plants interacting with their environment. In *Proceedings of ACM SIGGRAPH*, pages 397–410, 1996.

[MWH+06] P. Müller, P. Wonka, S. Haegler, A. Ulmer, and L. Van Gool. Procedural modeling of buildings. *ACM Transactions on Graphics*, 25(3) :614–623, 2006.

[MZWG07] P. Müller, G. Zeng, P. Wonka, and L. Van Gool. Image-based procedural modeling of facades. *ACM Transactions on Graphics*, 26(3), 2007.

[NFD07] B. Neubert, T. Franken, and O. Deussen. Approximate image-based tree-modeling using particle flows. *ACM Transactions on Graphics*, 26(3) :88 :1–8, 2007.

[NTB+91] A. Norton, G. Turk, B. Bacon, J. Gerth, and P. Sweeney. Animation of fracture by physical modeling. *The Visual Computer*, 7(4) :210–219, 1991.

[OBH02] J. F. O'Brien, A. W. Bargteil, and J. K. Hodgins. Graphical modeling and animation of ductile fracture. *ACM Transactions on Graphics*, 21(3) :291–294, 2002.

[OH99] J. F. O'Brien and J. K. Hodgins. Graphical modeling and animation of brittle fracture. In *Proceedings of ACM SIGGRAPH*, pages 137–146, 1999.

[OOI05] M. Okabe, S. Owada, and T. Igarashi. Interactive design of botanical trees using freehand sketches and example-based editing. *Computer Graphics Forum*, 24(3) :487–496, 2005.

[OS04] H. Ohlsson and S. Siepel. Real-time rendering of accumulated snow. In *Proceedings of SIGRAD*, 2004.

[Per85] K. Perlin. An image synthesizer. In *Proceedings of ACM SIGGRAPH*, pages 287–296, 1985.

[PGMG09] A. Peytavie, E. Galin, S. Merillou, and J. Grosjean. Arches : a framework for modeling complex terrains. *Computer Graphics Forum (Proc. of Eurographics '09)*, 28(2) :457–467, 2009.

[PH93] P. Prusinkiewicz and M. Hammel. A fractal model of mountains with rivers. In *Proceedings of Graphics Interface*, pages 174–180, 1993.

[PHL+09] W. Palubicki, K. Horel, S. Longay, A. Runions, B. Lane, R. Měch, and P. Prusinkiewicz. Self-organizing tree models for image synthesis. *ACM Transactions on Graphics*, 28(3) :58 :1–10, 2009.

[PHM93] P. Prusinkiewicz, M. S. Hammel, and E. Mjolsness. Animation of plant development. In *Proceedings of ACM SIGGRAPH*, pages 351–360, 1993.

[PJM94] P. Prusinkiewicz, M. James, and R. Měch. Synthetic topiary. In *Proceedings of ACM SIGGRAPH*, pages 351–358, 1994.

[PKA+05] M. Pauly, R. Keiser, B. Adams, P. Dutré, M. Gross, and L. J. Guibas. Meshless animation of fracturing solids. *ACM Transactions on Graphics*, 24(3) :957–964, 2005.

[PL90] P. Prusinkiewicz and A. Lindenmayer. *The Algorithmic Beauty of Plants*. Springer-Verlag, 1990.

[PM01] Y. I. H. Parish and P. Müller. Procedural modeling of cities. In *Proceedings of ACM SIGGRAPH*, pages 301–308, 2001.

[QTZ+06] L. Quan, P. Tan, G. Zeng, L. Yuan, J. Wang, and S. B. Kang. Image-based plant modeling. *ACM Transactions on Graphics*, 25(3) :599–604, 2006.

[REF+88] P. De Reffye, C. Edelin, J. Françon, M. Jaeger, and C. Puech. Plant models faithful to botanical structure and development. In *Proceedings of ACM SIGGRAPH*, pages 151–158, 1988.

[Ric53] B. R. Rich. An investigation of heat transfer from an inclined flat plate in free convection. *Trans. ASME*, 75 :489–499, 1953.

[RLP07] A. Runions, B. Lane, and P. Prusinkiewicz. Modeling trees with a space colonization algorithm. In *Eurographics Workshop on Natural Phenomena*, pages 63–70, 2007.

[RMMD04] A. Reche-Martinez, I. Martin, and G. Drettakis. Volumetric reconstruction and interactive rendering of trees from photographs. *ACM Transactions on Graphics*, 23(3) :720–727, 2004.

[SBSCO06] A. Sharf, M. Blumenkrants, A. Shamir, and D. Cohen-Or. SnapPaste : an interactive technique for easy mesh composition. *The Visual Computer*, 22(9) :835–844, 2006.

[SCFRC01] P.-P. J. Sloan, III C. F. Rose, and M. F. Cohen. Shape by example. In *Proceedings of the Symposium on Interactive 3D graphics*, pages 135–143, 2001.

[SCOL+04] O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, and H.-P. Seidel. Laplacian surface editing. In *Proceedings of the Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 175–184, 2004.

[SEN08] I. Saltvik, A. C. Elster, and H. R. Nagel. Parallel methods for real-time visualization of snow. In *Applied Parallel Computing. State of the Art in Scientific Computing*, volume 4699, pages 218–227, 2008.

[SFS05] L. Streit, P. Federl, and M. C. Sousa. Modelling plant variation through growth. *Computer Graphics Forum*, 24(3) :497–506, 2005.

[SLSS06] L. Streit, P. Lapides, M. C. Sousa, and E. Sharlin. Modeling plant variations through 3d interactive sketches. In *3rd Eurographics Workshop on Sketch-Based Interfaces and Modeling*, 2006.

[SOH99] R. Sumner, J. O'Brien, and J. Hodgins. Animating sand, mud, and snow. *Computer Graphics Forum*, 18(1) :17–26, 1999.

[SRDT01] I. Shlyakhter, M. Rozenoer, J. Dorsey, and S. Teller. Reconstructing 3d tree models from instrumented photographs. *IEEE Computer Graphics and Applications*, 21(3) :53–61, 2001.

[Sti80] G. Stiny. Introduction to shape and shape grammars. *Environment and Planning B*, 7(3) :343–351, 1980.

[SWB01] J. Smith, A. Witkin, and D. Baraff. Fast and controllable simulation of the shattering of brittle objects. *Computer Graphics Forum*, 20(2) :81–90, 2001.

[SWG+03] P. V. Sander, Z. J. Wood, S. J. Gortler, J. Snyder, and H. Hoppe. Multi-chart geometry images. In *Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry processing*, pages 146–155, 2003.

[TF88] D. Terzopoulos and K. Fleischer. Deformable models. *The Visual Computer*, 4(6) :306–331, 1988.

[TPBF87] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer. Elastically deformable models. *Computer Graphics*, 21 :205–214, 1987.

[TZW+07] P. Tan, G. Zeng, J. Wang, S. B. Kang, and L. Quan. Image-based tree modeling. *ACM Transactions on Graphics*, 26(3) :87 :1–8, 2007.

[VAW+10] C. A. Vanegas, D. G. Aliaga, P. Wonka, P. Müller, P. Waddell, and B. Watson. Modelling the Appearance and Behaviour of Urban Spaces. *Computer Graphics Forum*, 29(1) :25–42, 2010.

[vBBK08] O. Šťava, B. Beneš, M. Brisbin, and J. Křivánek. Interactive terrain modeling using hydraulic erosion. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 201–210, 2008.

[WBCG09] J. Wither, F. Boudon, M.-P. Cani, and C. Godin. Structure from silhouettes : a new paradigm for fast sketch-based design of trees. *Computer Graphics Forum*, 28(2) :541–550, 2009.

[WI04] N. Watanabe and T. Igarashi. A sketching interface for terrain modeling. In *ACM SIGGRAPH Posters*, page 73, 2004.

[WL00] L.-Y. Wei and M. Levoy. Fast texture synthesis using tree-structured vector quantization. In *Proceedings of ACM SIGGRAPH*, pages 479–488, 2000.

[WM05] D. Walton and D. Meek. A controlled clothoid spline. *Computers & Graphics*, 29(3) :353–363, 2005.

[WMWG09] B. Weber, P. Müller, P. Wonka, and M. Gross. Interactive geometric simulation of 4d cities. *Computer Graphics Forum*, 28(2) :481–492, 2009.

[WP95] J. Weber and J. Penn. Creation and rendering of realistic trees. In *Proceedings of ACM SIGGRAPH*, pages 119–128, 1995.

[WWSR03] P. Wonka, M. Wimmer, F. Sillion, and W. Ribarsky. Instant architecture. *ACM Transactions on Graphics*, 22(4) :669–677, 2003.

[WWXP06] C. Wang, Z. Wang, T. Xia, and Q. Peng. Real-time snowing simulation. *The Visual Computer*, 22(5) :315–323, 2006.

[WY04] Q. Wu and Y. Yu. Feature matching and deformation for texture synthesis. *ACM Transactions on Graphics*, 23(3) :364–367, 2004.

[YZX+04] Y. Yu, K. Zhou, D. Xu, X. Shi, H. Bao, B. Guo, and H.-Y. Shum. Mesh editing with poisson-based gradient field manipulation. *ACM Transactions on Graphics*, 23(3) :644–651, 2004.

[ZSTR07] H. Zhou, J. Sun, G. Turk, and J. M. Rehg. Terrain synthesis from digital elevation models. *IEEE Transactions on Visualization and Computer*, 13(4) :834–848, 2007.

Publications

Revues internationales avec comité de lecture

N. Maréchal, E. Guérin, E. Galin, S. Mérillou, and N. Mérillou. Heat transfer simulation for modeling realistic winter sceneries. *Computer Graphics Forum*, 29(2) :449-458, 2010.

E. Galin, A. Peytavie, N. Maréchal, and E. Guérin. Procedural generation of roads. *Computer Graphics Forum*, 29(2) :429-438, 2010.

Conférences internationales avec comité de lecture

N. Maréchal, E. Galin, E. Guérin, and S. Akkouche. Component-based model synthesis for low polygonal models. In *Graphics Interface*, 2010.

Conférences nationales sans comité de lecture

N. Maréchal, E. Galin, E. Guérin, and S. Akkouche. Génération de variétés d'objets par fragments. In *AFIG'08 : 21^{ième} journées de l'Association Française d'Informatique Graphique*, 63-71, 2008.

A. Peytavie, N. Maréchal, E. Guérin, and E. Galin. Génération procédurale de routes. In *AFIG'09 : 22^{ième} journées de l'Association Française d'Informatique Graphique*, 123-132, 2009.

Génération de contenu graphique

Résumé : L'objectif de cette thèse est la recherche de nouvelles techniques de génération de contenu numérique pour des applications de jeu vidéo. Le manque de variété de terrains, d'objets et de détails affecte fortement le réalisme des paysages de synthèse. Dans ce contexte, un des principaux goulots d'étranglements est la modélisation des ressources graphiques permettant de créer les scènes. Afin de simplifier et d'accélérer cette tâche, nous présentons des méthodes permettant de générer automatiquement du contenu graphique pour créer de grands paysages à la fois complexes et originaux.

Notre première approche permet de créer et d'éditer rapidement des variétés d'objets à partir d'un modèle initial fourni par un graphiste, sous la contrainte d'une représentation avec très peu de triangles. Nous présentons également une méthode de génération procédurale des variétés d'objets. Ensemble, ces méthodes permettent de créer aussi bien des variétés de formes naturelles que des ouvrages d'arts tels que des routes, des ponts et des tunnels capables de s'adapter automatiquement au relief d'un paysage.

Nous proposons une autre méthode, s'appuyant sur une simulation physique et thermique, pour créer des paysages hivernaux évoluant au cours du temps en fonction des conditions climatiques. Cette approche permet de suivre l'évolution du manteau neigeux ainsi que l'épaisseur de la glace qui se forme en surface d'un lac.

Graphical content generation

Abstract : The goal of this thesis is the search for new techniques for generating digital content for video game applications. The lack of variety of objects and details strongly affects the realism of synthetic landscapes. In this context, a major bottleneck is the creation of graphical assets to create the scenes. To simplify and accelerate this task, we present methods to automatically generate graphical content with a view to creating large, convincing and realistic landscapes.

We present a first category of method to create and edit quickly a vast variety of objects from an initial model provided by a graphic designer, under the constraint of low polygonal modeling. We also present a procedural method for generating of varieties of objects. Together, these methods can create many varieties of natural forms as well as roads, bridges and tunnels that can automatically adapt to the landform features of an input terrain.

We propose another method, based on a physical simulation and heat transfer simulation to create winter landscapes that evolve over time depending on the weather conditions. This approach allows us to to follow the evolution of the snowpack and the thickness of the ice that forms on the surface of a lake.