AFIG 2008 / Toulouse
IRIT Presse

Génération de Variétés d'Objets par Fragments

Nicolas Maréchal[1], Eric Galin[2], Eric Guérin[1], Samir Akkouche[1]

[1]LIRIS - CNRS - UCBL, France
[nicolas.marechal | eric.guerin | samir.akkouche]@liris.cnrs.fr

[2]LIRIS - CNRS - Université Lumière Lyon 2, France
eric.galin@liris.cnrs.fr

Abstract

Cet article présente une méthode permettant de générer une grande variété d'objets de même nature à partir d'un modèle initial. Le processus de génération de variétés consiste à découper l'objet initial en fragments dont la géométrie et la texture sont modifiées avant d'être stockés dans un atlas. Les fragments contenus dans l'atlas sont alors recombinés par instanciation pour générer un très grand nombre de modèles différents pour un faible surcoût mémoire. Notre approche a été intégrée à la plateforme commune Twilight 2 des sociétés Eden Games et Widescreen Games et validée sur différents modèles.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism

Keywords: génération procédurale, modélisation géométrique, variétés, fragments

1. Introduction

La puissance actuelle des machines permet d'augmenter considérablement la qualité des images de synthèse. Néanmoins, les paysages naturels manquent toujours de réalisme. L'augmentation de ce réalisme passe par le besoin de modéliser un grand nombre de détails et de variétés d'objets, brisant ainsi les répétitions engendrées par une forte instanciation. Le travers de cette méthode est la quantité accrue de données à stocker. L'ajout de détails [CH04] et d'imperfections [ZDW*05] sur la texture des objets contribue également fortement au réalisme d'une scène naturelle. Les techniques de modélisation procédurale de terrains [MKM89], de plantes [PHM93] ou d'immeubles [MWH*06] permettent de générer de la variété pour ces objets mais possèdent certaines limites :

- il est nécessaire de définir des règles de génération nombreuses et complexes pour pouvoir contrôler le résultat ;
- ces techniques sont spécifiques à une seule catégorie d'objets ;
- il est généralement impossible de se conformer à un style artistique défini par un designer, ce qui est fondamental dans les jeux vidéo et le cinéma.

Pour résoudre ce type de problèmes, nous proposons une méthode générique qui s'applique à une grande variété d'objets. Notre méthode se décompose en deux processus et permet, à partir d'un seul modèle initial texturé, de générer des variétés de géométries et de texture de ce modèle. Le processus de génération de variétés de géométrie consiste à découper l'objet en fragments qui sont ensuite modifiés pour créer un atlas avant d'être assemblés pour former des variétés d'objets. Le processus de génération de variétés de texture consiste à appliquer des modifications à la texture de l'objet qui sont ensuite compactées dans des atlas de fragments de texture avant d'être combinées pour générer des variétés d'objets différents par la texture. Les principales contributions de cet article sont :

- une méthode générique permettant de générer des variétés d'objets de même nature à partir d'un seul objet initial texturé de topologie quelconque (Figures 2, 11) pour un très faible surcoût mémoire ;

Figure 1: *Variété d'arbres dans une forêt de Central Park, tous les arbres ont été générés à partir d'un modèle d'arbre d'Alone In The Dark™.*

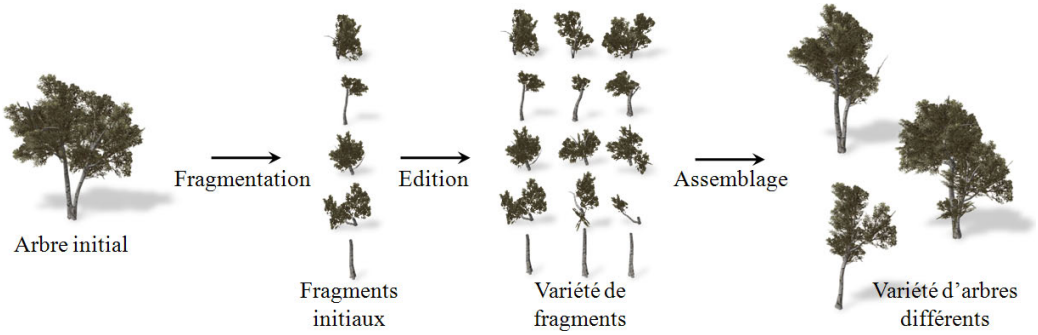

Figure 2: *Génération de variétés d'objets par modification de la géométrie.*

- une technique de modification de fragments permettant de garantir automatiquement la continuité entre les fragments lors de l'assemblage ;
- une méthode de compactage des modifications de la texture permettant d'utiliser les coordonnées de texture existantes et de garantir les raccords de texture initiaux.

En outre, notre méthode est générale et applicable à n'importe quel contexte même celui des jeux vidéo où les contraintes sont celles d'un faible espace mémoire, d'une faible tesselation des modèles et d'un affichage temps réel.

L'organisation de cet article est la suivante. En section 2, nous présentons les travaux existant sur la variété de géométrie. Dans la section 3, nous présentons le processus de génération de variétés de géométrie par fragments. En section 4, nous présentons le processus de génération de variétés de texture par fragments. Nous présentons nos résultats en section 5 puis nous concluons dans la section 6.

2. Etat de l'art

Il existe plusieurs approches permettant de générer des variétés d'objets.

Génération procédurales : les techniques de génération à base de grammaires formelles telles que les L-systèmes [MP96] [PHM93] [SFS05] sont fréquemment utilisées pour générer des plantes et simuler leur croissance. Pour les arbres, Weber et Penn [WP95] proposent une approche purement géométrique en partant du principe que la structure d'un arbre ne possède pas plus de quatre niveaux de récursivité. À chaque niveau de récursivité, ils spécifient un ensemble de paramètres choisis dans un intervalle de possibilités. Plus récemment, des grammaires orientées sur les formes géométriques [WWSR03] [MWH*06] [PM01] ont été définies pour générer des bâtiments et des villes.

Ces méthodes ont certaines limites : il est difficile d'obtenir un objet ayant une forme souhaitée car les paramètres permettant de contrôler la forme d'un objet sont extrêmement nombreux et complexes. Bien que ces méthodes permettent de générer énormément de modèles différents et fournissent des résultats très réalistes, il devient impossible de créer et de rendre de très grandes scènes composées de centaines ou de milliers d'objets. Ces scènes doivent être simplifiées tout en préservant leur apparence globale [DHL*98] [CHPR07].

Synthèse à partir d'exemples : Merrell [Mer07] propose une méthode procédurale inspirée des techniques de génération de textures par l'exemple [WL00] [EF01] permettant de générer de larges modèles géométriques à partir d'exemples découpés en morceau en utilisant une grille 3D puis assemblés. Ceci permet de générer de nombreux objets et environnements différents. Sa méthode est très adaptée aux objets architecturaux ayant des éléments qui se répètent et structurés sous forme de grille 3D.

Cependant, son algorithme possède des limites qu'il détaille. Les objets exemples tels que les arbres ou les terrains doivent être modélisés avec attention et spécifiquement pour son algorithme. Ceci est incompatible avec une de nos motivations qui est que l'artiste ne soit pas sous contrainte pour modéliser les objets exemples et conserve ses habitudes de travail.

Méthodes d'interpolation : la génération d'objets peut également se faire en utilisant les techniques de morphing [SCFRC01] [LDSS99] [ACOL00]. Leur but est de créer une transition entre un objet initial et un objet final.

Cependant, il est très difficile de mettre en correspondance deux objets topologiquement différents et complexes tels que des arbres.

3. Génération de variétés de géométries par fragments

Le processus de génération de variétés de géométries par fragments (Figure 2) se décompose en trois étapes : la décomposition du modèle, la création des variétés de fragments et l'assemblage de ces fragments. Étant donné un objet initial O maillé et texturé, notre système le décompose en frag-

ments notés F_i tel que :

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

où n est le nombre de fragments composant l'objet.

Les fragments F_i sont modifiés pour créer des ensembles de variétés de fragments qui sont stockés dans un atlas. Nous notons F_i^j la $j^{\text{ème}}$ modification de F_i et $\mathcal{F}_i$ l'ensemble de ces modèles.

L'atlas $\mathcal{F}$ stockant les variétés de fragments $\mathcal{F}_i$ nous permet de générer un ensemble d'objets $\mathcal{V}$ ayant des formes différentes en assemblant les fragments F_i^j. Un objet $\tilde{O} \in \mathcal{V}$ est défini par :

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

Le nombre d'objets que nous pouvons générer par cette approche est donné par :

$$\text{Card}(\mathcal{V}) = \prod_{i=1}^{n} \text{Card}(\mathcal{F}_i)$$

où $\text{Card}(\mathcal{F}_i)$ est le nombre de fragments de l'ensemble $\mathcal{F}_i$.

Le processus de génération de variétés de géométries permet de générer un très grand nombre de variétés d'objets pour un faible nombre de fragments.

3.1. Fragmentation

Cette étape décompose un objet initial maillé et texturé O en fragments F_i. Un fragment est un morceau de maillage de l'objet initial obtenu par découpage. Notre technique permet de découper O selon n'importe quelle méthode de découpe. Dans notre implémentation, l'utilisateur a le choix entre deux méthodes :

– dans le cas général, l'utilisateur définit un chemin quelconque sur la surface de l'objet. Cette méthode ajoute des nouveaux sommets le long des arêtes dont les coordonnées de texture sont obtenues par interpolation linéaire des coordonnées de texture des sommets des faces découpées (Figure 3) ;

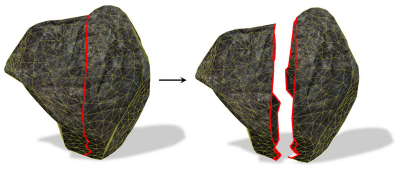

Figure 3: *Chemin de découpe ajoutant des sommets.*

– lorsque le nombre de sommets est limité par la contrainte du temps réel, il est préférable de ne pas en rajouter et de définir un chemin en sélectionnant des sommets existants. Nous appliquons cette découpe pour les maillages d'objets des jeux vidéo (Figure 4).

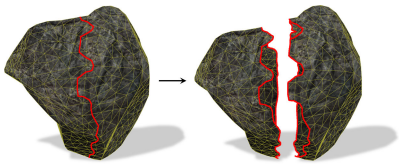

Figure 4: *Chemin de découpe utilisant des sommets existant.*

Nous appelons R_{ij} le contour commun définissant la région de connexion entre deux fragments F_i et F_j obtenus par découpage. La figure 5 illustre le découpage d'un bouleau en deux fragments F_i et F_j ainsi que leur région de connexion commune R_{ij}.

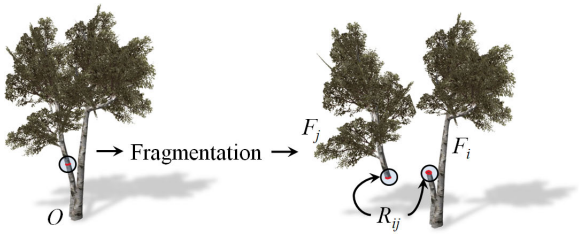

Figure 5: *Bouleau découpé en deux fragments.*

Lors de la fragmentation de l'objet O, nous construisons un graphe G associé à O et noté :

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

où $\mathcal{N}$ désigne les nœuds de G et $\mathcal{A}$ les arcs entre deux nœuds de G. Les nœuds contiennent les fragments F_i issus du découpage et il existe un arc A_{ij} entre deux fragments F_i et F_j lorsqu'ils possèdent une région de connexion commune R_{ij}. La figure 6 illustre le graphe associé au bouleau que nous avons découpé en cinq fragments.

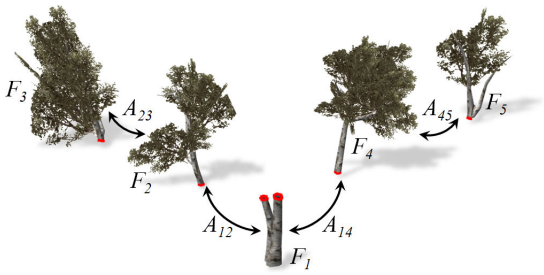

Figure 6: *Graphe associé au bouleau constitué de cinq fragments.*

3.2. Création de l'atlas de fragments

Nous définissons l'atlas de fragments modifiés en éditant les différents fragments F_i pour créer des variétés de fragments notés $\mathcal{F}_i = \{F_i^j\}$. Lors de la modification d'un modèle, tous les fragments de $\mathcal{F}_i$ doivent rester connectables à tous les fragments de $\mathcal{F}_j$ lorsque ces ensembles sont reliés par un arc dans le graphe G et inversement.

Les nœuds du graphe G sont redéfinis de manière à ce que les nœuds $\mathcal{N}$ contiennent les modèles $\mathcal{F}_i$ et G est mis à jour de façon à garantir la continuité de la texture et de la géométrie entre les fragments $\mathcal{F}_i$ et $\mathcal{F}_j$ lorsque la région de connexion R_{ij} d'un fragment F_i et/ou d'un fragment F_j est affectée par une modification.

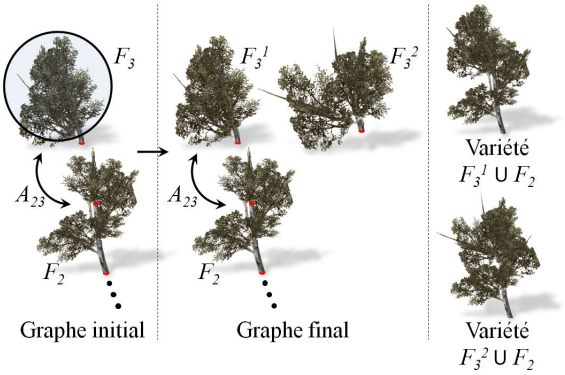

Figure 7: *Mise à jour du graphe du bouleau pour une modification n'affectant pas de régions de connexion.*

En fonction de l'endroit où est appliquée la modification sur le maillage de O, le graphe est mis à jour comme suit :

Modification locale d'un fragment : lorsque la modification du fragment F_i n'affecte pas les sommets des régions de connexion de celui-ci, un nouveau fragment est ajouté au nœud $\mathcal{F}_i$ de G (Figure 7).

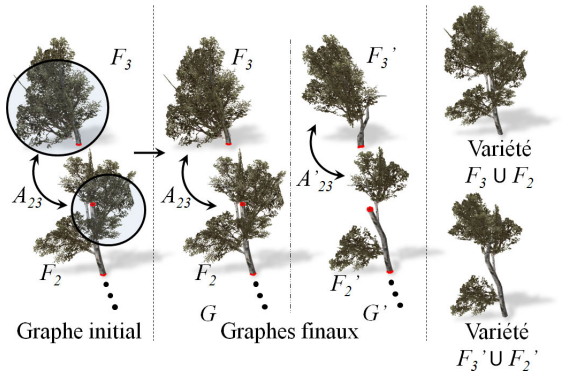

Figure 8: *Mise à jour du graphe du bouleau pour une modification quelconque affectant une région de connexion.*

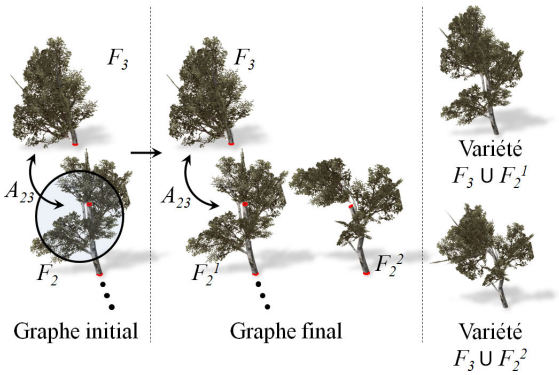

Figure 9: *Mise à jour du graphe du bouleau pour une transformation solide d'une région de connexion.*

Modification de la région de connexion : lorsque la modification affecte des sommets de la région de connexion R_{ij} de deux fragments F_i et F_j, le graphe G est dupliqué pour créer un nouveau graphe G' où les fragments de $\mathcal{F}_i$ différents du fragment F_i et ceux de $\mathcal{F}_j$ différents de F_j sont supprimés. Ceci crée un nouvel ensemble de possibilités en divisant l'atlas (Figure 8).

Lorsque la modification est une transformation solide du type translation, rotation ou mise à l'échelle appliquée à l'ensemble des sommets de la région de connexion R_{ij} du fragment F_i, nous stockons les transformations effectuées dans un repère associé à R_{ij} afin d'éviter de diviser l'atlas et ainsi avoir une plus grande combinatoire. Ce repère nous permet de connecter correctement les fragments $\mathcal{F}_j$ au fragment F_i lors de l'assemblage (Figure 9).

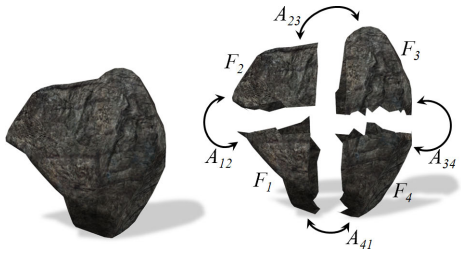

Figure 10: *Graphe cyclique issue de la fragmentation d'un rocher.*

Lors de certaines fragmentations, le graphe associé à l'objet peut posséder des parties cycliques (Figure 10). Une modification, par transformation solide, d'une région de connexion d'un fragment appartenant à un cycle rend impossible l'assemblage et provoque des fissures. Par conséquent, nous interdisons ce type de modification dans notre implémentation garantissant ainsi la connectabilité de tous les fragments.

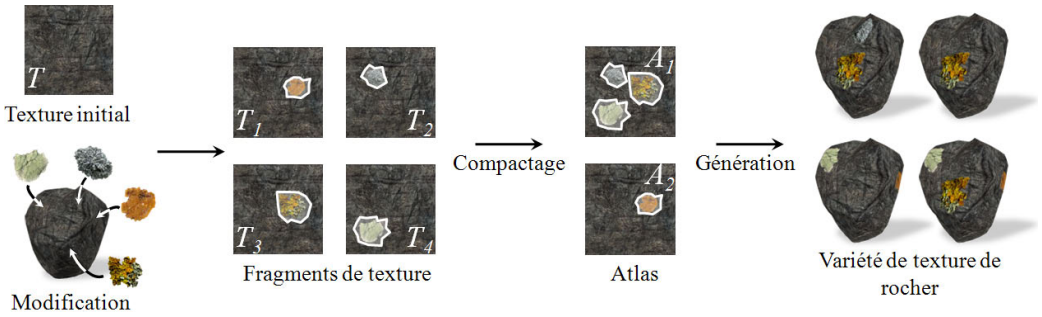

Figure 11: *Génération de variétés d'objets par modification de la texture.*

3.3. Assemblage

Cette étape consiste à générer un nouvel objet $\tilde{O}$ en sélectionnant un fragment F_i^j dans chaque $\mathcal{F}_i$. Les fragments sont assemblés en faisant correspondre les sommets des régions de connexion garantissant ainsi la continuité de la géométrie et de la texture entre fragments.

Durant cette étape d'assemblage, il est nécessaire de recalculer la position de certains fragments lorsqu'une transformation solide a été appliquée à une région de connexion. Pour cela, nous appliquons la transformation du repère de la région de connexion à tous les fragments du sous-arbre pour garantir la continuité entre les fragments.

Dans notre implémentation, le choix des fragments F_i^j de $\tilde{O}$ peut être fait soit de manière manuelle par l'utilisateur, soit automatiquement en fonction d'heuristiques. Bien que le choix manuel permette un contrôle plus précis, cette méthode s'avère limitée lorsqu'un très grand nombre d'objets doit être généré comme dans le cas de synthèse de forêt (Figures 1, 17).

Nous avons implémenté une heuristique permettant d'attribuer une note globale $\in [0..1]$ à l'objet généré. Cette note globale est fonction de trois notes $\in [0..1]$ attribuées à chaque fragment par l'artiste. Ces notes représentent l'importance des modifications apportées à la géométrie, à la texture et au repère des régions de connexion. Chaque note doit être attribuée de telle sorte qu'elle reflète les modifications subies par le fragment F_i^j par rapport au fragment initial F_i.

4. Génération de variétés de texture par fragments

Étant donné un objet initial O, nous proposons une méthode permettant de modifier la texture initiale notée T de cet objet et de générer un très grand nombre de variétés de texture à partir d'un faible nombre de modifications.

Le processus de génération de variétés de texture par fragments (Figure 11) se décompose en trois étapes : ajout de détails par des modifications locales de la texture initiale T et la création de fragments de texture notées T_k, le compactage des fragments de texture T_k en atlas de fragments de textures et la génération des variétés d'objets par combinaison des éléments de texture modifiés.

4.1. Modifications de la texture

Cette étape consiste à appliquer des modifications successives sur la texture de l'objet. Dans notre système, nous modifions la texture en peignant directement sur les facettes de l'objet O. La région où la texture a été modifiée définit un fragment de texture noté $T_k = (P_k, M_k, C_k)$. P_k est un masque de pixel permettant de définir la région de modification effectuée. M_k est un masque de facettes de O représentant la surface de texture utilisée par les facettes de l'objet O qui ont été modifiées dans l'espace de texture. C_k est un cluster regroupant les facettes de l'objet affectées et garantissant l'affichage de la modification dans son intégralité lors de la génération de variétés (Figure 12).

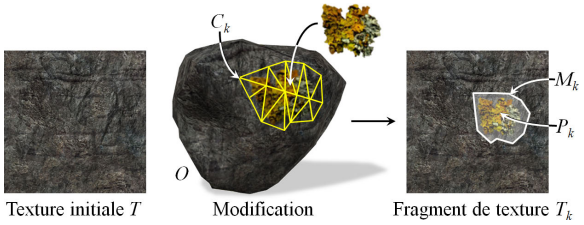

Figure 12: *Fragment de texture $T_k = (P_k, M_k, C_k)$ obtenus suite à une modification locale.*

Recouvrements et tiling : dans le cas général, une modification locale sur l'objet correspond à une modification locale de la texture mais certaines coordonnées de texture du maillage peuvent provoquer soit des recouvrements de cluster de faces dans l'espace de texture (Figure 13) soit des répétitions cycliques de la texture (tiling) sur l'objet (Figure 14). Par conséquent, des modifications locales de l'objets à différents endroits peuvent être en conflit car elles se situent à la même position dans l'espace de texture. Pour résoudre ce problème, nous répartissons les modifications sur plusieurs atlas de fragments de textures lors du compactage.

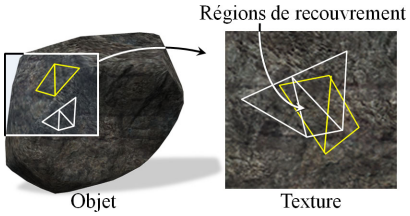

Figure 13: *Recouvrement de clusters.*

Le tiling peut également exister au niveau des facettes de l'objet et donc provoquer la répétition indésirable d'une modification au sein même de celles-ci (Figure 14). Pour résoudre ce problème, nous subdivisons ces facettes au niveau des répétitions de la texture. Ceci nous permet de supprimer le tiling et nous garantit qu'une modification de ces facettes ne produira pas de répétition indésirable de la modification.

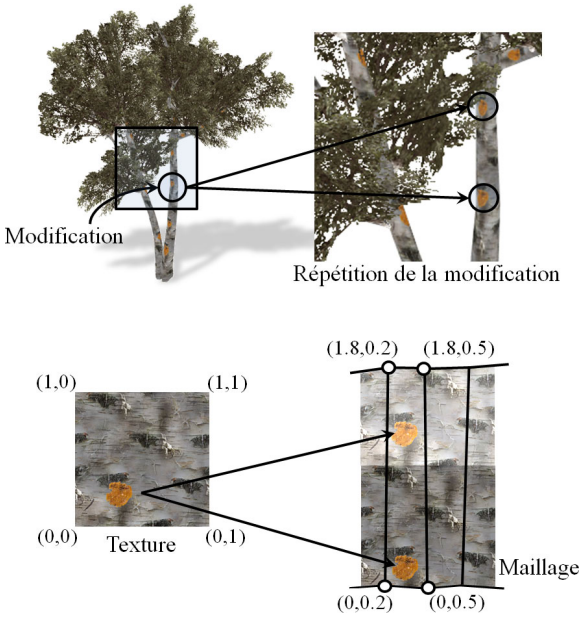

Figure 14: *Suppression d'une répétition indésirable produit par le tiling par subdivision des facettes.*

À l'issue de n étapes d'édition, nous disposons de n fragments de texture ce qui est coûteux en mémoire. Par conséquent, il est nécessaire de réduire le nombre de fragments de texture en les compactant dans des atlas de fragments de textures A_i.

4.2. Compactage

Cette étape permet de réunir les fragments de texture T_k dans des atlas de fragments de textures notée A_i (Figure 15). Chaque atlas de fragments de textures A_i est une copie de la texture initiale T dans laquelle nous regroupons les fragments de texture T_k qui ne sont pas en conflit.

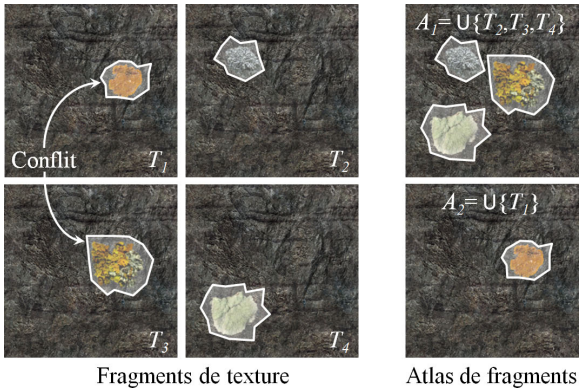

Figure 15: *Compactage des fragments de texture en atlas de fragments de textures.*

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 de fragments de textures si et seulement si $P_i \bigcap M_j = \emptyset$ et $P_j \bigcap M_i = \emptyset$. Ce type de compactage n'est pas optimal en terme de mémoire par rapport à un compactage minimisant le nombre de pixels non utilisés [LPRM02] [SWG*03] dans les atlas de fragments de textures. Cependant, il permet d'utiliser les coordonnées de texture du maillage de l'objet existantes et ainsi réduire la mémoire utilisée pour stocker l'objet en évitant d'ajouter de nouvelles coordonnées de texture pour chaque atlas de fragments de textures. De plus, l'utilisation de coordonnées de texture uniques garantit la préservation des raccords de textures initiaux.

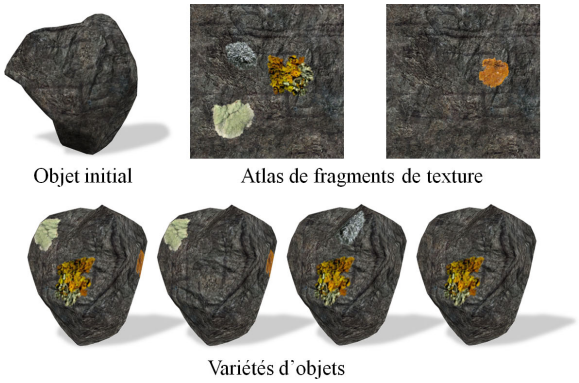

Figure 16: *Quatre variétés de rocher parmi les* $2^4 = 16$ *possibilités.*

Dans notre implémentation, nous énumérons tous les compactages possibles en testant toutes les intersections entre les masques de pixels et les masques de faces de tous

les fragments de texture et nous conservons l'ensemble des atlas de fragments de textures ayant le cardinal le plus faible.

Après compactage nous associons aux facettes des clusters C_k du fragments de texture T_k l'identifiant de l'atlas de fragments de textures A_i dans lequel a été compacté ce dernier. À ce stade, les masques de pixels et de faces ne sont plus nécessaires et peuvent donc être supprimés des fragments de texture.

4.3. Génération de variétés

Cette étape consiste à choisir parmi les fragments de texture stockés dans les atlas ceux qui seront utilisés pour faire apparaître un certains nombre de modifications sur l'objet et générer des variétés d'objets par combinaison (Figure 16).

Dans notre implémentation, le choix des fragments de texture peut être fait soit de manière manuelle par l'utilisateur soit automatiquement en fonction d'une probabilité associée à chaque fragment de texture et qui est définie par l'utilisateur.

Lorsqu'un fragment de texture T_k a été sélectionné, nous attribuons à chacune des facettes du cluster C_k l'identifiant de l'atlas A_i contenant le fragment T_k. Les facettes du cluster sont ensuite verrouillées excluant ainsi les autres fragments de texture dont les clusters utilisent ces facettes des choix possibles. Ceci permet d'éviter un probable remplacement partiel d'un fragment de texture par un autre.

À l'affichage de l'objet, nous utilisons les identifiants associés aux facettes pour charger les atlas contenant les fragments de texture qui ont été sélectionnés et pour plaquer les modifications de texture sur ces facettes en utilisant les coordonnées de texture du maillage existantes.

5. Résultats

Les algorithmes de notre méthode de génération de variétés ont été implémentés et intégrés dans la plateforme de conception *Twilight 2* commune aux sociétés Eden Games et Widescreen Games pour le projet GENAC 2. Les ressources graphiques utilisées pour illustrer nos résultats sont issues des jeux vidéo *Alone In The Dark*™ et *Test Drive Unlimited 2*™ conçus par la société Eden Games.

Les répétitions engendrées par une forte instanciation des objets brisent le réalisme des scènes. Les résultats démontrent l'efficacité de notre méthode pour briser ces répétitions et accroître le réalisme en générant un très grand nombre de variétés d'objets pour un faible surcoût mémoire et peu de modifications.

Le tableau 1, présente les caractéristiques des objets initiaux : nombre de sommets et de facettes, et taille mémoire (en ko) du modèle. Le tableau 2 présente le nombre de fragments générés à l'issus du processus de découpage, le nombre de variété par fragments, le nombre d'objets ainsi générés et le surcoût mémoire (en ko) engendré par les modifications de géométrie et la structure de données.

Modèle	Bouleau	Rocher	Colonne	Maison
Sommets	745	578	233	568
Faces	628	1152	312	923
Mémoire	328	146	98	186

Table 1: *Complexité des modèles utilisés et taille mémoire.*

Modèle	Bouleau	Rocher	Colonne	Maison
Fragments	6	4	5	7
Variétés	3	4	4	3
Objets	729	4096	1024	2187
Mémoire	+177 (54%)	+469 (321%)	+417 (425%)	+90 (48%)

Table 2: *Nombre d'objets générés par notre méthode et surcoût mémoire engendré.*

Central Park : pour le jeu *Alone In The Dark*™ se déroulant autour de Central Park, 17 essences d'arbres et 3 variétés par essences ont été réalisées par les graphistes portant le nombre total d'arbres à 51. Nous avons appliqué notre méthodes aux différents modèles d'arbres de Central Park pour générer automatiquement une plus grande variété d'arbres (Figures 17, 1). Les modifications apportées aux fragments ont été principalement des suppressions des branchages et de feuillages ce qui explique le faible surcoût mémoire obtenu (54%).

Il a fallu 3 heures à une personne non graphiste pour fragmenter et créer les 729 variétés d'arbres. À titre de comparaison, le temps nécessaire à un graphiste pour créer un arbre est de 8 heures. Notre méthode permet donc de réduire les temps et coût de production qui sont des facteurs cruciaux dans le secteur industriel.

Éboulement : la figure 18 montre un éboulis dans Central Park. Initialement, il n'existait aucun éboulis dans cette scène et notre système a permi d'en créer un à partir d'un seul rocher. En plus des modifications de géométrie permettant de générer 256 rochers, nous avons effectué 4 modifications de la texture qui ont été compactées en 2 atlas de fragments de textures et qui ont permis de générer 4096 rochers différents. Au cours des modifications de géométrie, le nombre de sommets est resté constant ce qui explique un surcoût mémoire (321%) plus important que pour le bouleau. De plus, les modifications apportées à la texture de l'objet ont engendré un surcoût mémoire de 2 textures (256x256) supplémentaires.

Figure 17: *Une partie de la forêt de Central Park peuplé de bouleaux générés par notre méthode à partir d'un modèle initial.*

Figure 19: *Un quartier résidentiel à Hawaii : toutes les maisons sont différentes et ont été générées à partir d'un modèle initial découpé en fragments.*

Quartier résidentiel à Hawaii : dans le jeu *Test Drive Unlimited 2*™ se déroulant sur l'île d'Hawaii, les variétés de maisons des quartiers résidentiels ont été modélisées manuellement par les graphistes qui ont ajouté des éléments tels que des abris ou des garages à un bloc initial. Dans la figure 19 nous montrons le résultat obtenu en utilisant notre système qui a permi d'automatiser cette tâche.

Great Hall du Metropolitan Museum of Art : la figure 20 montre l'entrée du Great Hall du Metropolitan Museum of Art de New York en ruine. Avant d'appliquer notre système aux colonnes, seules quelques unes d'entre elles étaient abimées. Les modifications géométriques apportées à la colonne ont été principalement des cassures nécessitant l'ajout de sommets ce qui explique que le surcoût mémoire soit plus important que pour le rocher (425%).

6. Conclusions

Dans cet article, nous avons présenté une méthode qui, à partir d'un modèle initial texturé, permet de générer des variétés de ce modèle pour un faible surcoût mémoire et peu de modifications. Cette méthode permet de se conformer à un style artistique donné et peut être appliquée à différents types d'objets. Les fragments étant instanciables, notre système est bien adapté aux contraintes de l'affichage temps réel et en particulier au monde des jeux vidéo. Nos résultats montrent que cette technique accroît le réalisme des scènes en brisant les répétitions engendrées par l'instanciation. De

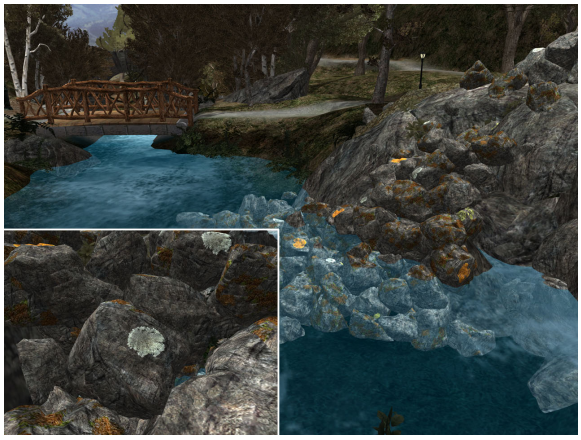

Figure 18: *Génération automatique des différents rochers d'un éboulement dans Central Park.*

Figure 20: *L'entrée du Great Hall du Metropolitan Museum of Art de New York en ruine.*

plus, elle permet de réduire les temps et coûts de production qui sont des facteurs cruciaux dans le secteur industriel.

Remerciements

Nos remerciements vont à F. Jay, P. Deltour, J.Y. Geffroy, G. le Proux de la Rivière et P. Bouvier et à toutes les autres personnes des sociétés Eden Games et Widescreen Games pour leur participation à ce projet. Nous remercions également E. Duranton pour l'implémentation de notre méthode sur la plateforme *Twilight 2* de la société Eden Games. Ce travail a bénéficié d'un financement dans le cadre du projet de la région Rhône-Alpes GENAC 2 (pôle de compétitivité Imaginove).

References

[ACOL00] ALEXA M., COHEN-OR D., LEVIN D. : As-rigid-as-possible shape interpolation. In *Proceedings of ACM SIGGRAPH* (2000), pp. 157–164.

[CH04] CARR N. A., HART J. C. : Painting detail. *ACM Transactions on Graphics (Proc. SIGGRAPH) 23*, 3 (2004), 845–852.

[CHPR07] COOK R., HALSTEAD J., PLANCK M., RYU D. : Stochastic simplification for aggregate detail. *ACM Transactions on Graphics (Proc. SIGGRAPH) 26*, 3 (2007), 79–88.

[DHL*98] DEUSSEN O., HANRAHAN P., LINTERMANN B., MĚCH R., PHARR M., PRUSINKIEWICZ P. : Realistic modeling and rendering of plant ecosystems. In *Proceedings of ACM SIGGRAPH* (1998), pp. 275–286.

[EF01] EFROS A. A., FREEMAN W. T. : Image quilting for texture synthesis and transfer. In *Proceedings of ACM SIGGRAPH* (2001), pp. 341–346.

[LDSS99] LEE A. W. F., DOBKIN D., SWELDENS W., SCHRÖDER P. : Multiresolution mesh morphing. In *Proceedings of ACM SIGGRAPH* (1999), pp. 343–350.

[LPRM02] LÉVY B., PETITJEAN S., RAY N., MAILLOT J. : Least squares conformal maps for automatic texture atlas generation. *ACM Transactions on Graphics (Proc. SIGGRAPH) 21*, 3 (2002), 362–371.

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

[MKM89] MUSGRAVE F. K., KOLB C. E., MACE R. S. : The synthesis and rendering of eroded fractal terrains. In *Proceedings of ACM SIGGRAPH* (1989), pp. 41–50.

[MP96] MĚCH R., PRUSINKIEWICZ P. : Visual models of plants interacting with their environment. In *Proceedings of ACM SIGGRAPH* (1996), pp. 397–410.

[MWH*06] MÜLLER P., WONKA P., HAEGLER S., ULMER A., GOOL L. V. : Procedural modeling of buildings. *ACM Transactions on Graphics (Proc. SIGGRAPH) 25*, 3 (2006), 614–623.

[PHM93] PRUSINKIEWICZ P., HAMMEL M. S., MJOLSNESS E. : Animation of plant development. In *Proceedings of ACM SIGGRAPH* (1993), pp. 351–360.

[PM01] PARISH Y. I. H., MÜLLER P. : Procedural modeling of cities. In *Proceedings of ACM SIGGRAPH* (2001), pp. 301–308.

[SCFRC01] SLOAN P.-P. J., CHARLES F. ROSE I., COHEN M. F. : Shape by example. In *Proceedings of the Symposium on Interactive 3D graphics* (2001), pp. 135–143.

[SFS05] STREIT L., FEDERL P., SOUSA M. C. : Modelling plant variation through growth. *Computer Graphics Forum (Proc. of Eurographics '05) 24*, 3 (2005), 497–506.

[SWG*03] SANDER P. V., WOOD Z. J., GORTLER S. J., SNYDER J., HOPPE H. : Multi-chart geometry images. In *Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry processing* (2003), pp. 146–155.

[WL00] WEI L.-Y., LEVOY M. : Fast texture synthesis using tree-structured vector quantization. In *Proceedings of ACM SIGGRAPH* (2000), pp. 479–488.

[WP95] WEBER J., PENN J. : Creation and rendering of realistic trees. In *Proceedings of ACM SIGGRAPH* (1995), pp. 119–128.

[WWSR03] WONKA P., WIMMER M., SILLION F., RIBARSKY W. : Instant architecture. *ACM Transactions on Graphics (Proc. SIGGRAPH) 22*, 3 (2003), 669–677.

[ZDW*05] ZHOU K., DU P., WANG L., SHI J., GUO B., SHUM H.-Y. : Decorating surfaces with bidirectional texture functions. *IEEE Transactions on Visualization and Computer Graphics 11*, 5 (2005), 519–528.