Village - Fiche de lecture

Raoelisolonarivony - MISA M2

Janvier 2017

Première partie

A Survey of Procedural Methods for Terrain Modelling (2009) de Ruben M. Smelik et al.

Chapitre 1

Survey

1.1 Abstract

Les **méthodes procédurales** sont des alternatives négligées pour la création manuelle de contenu. Les points négatifs de ces méthodes concernent surtout l'*imprévisibilité et le manque de contrôle sur les résultats et l'absence de solutions intégrées*, bien que des publications récentes traitent de plus en plus la résolution de ces problèmes. Cette note recense les méthodes procédurales appliquées sur la **modélisation de terrain**, évaluant le réalisme de leur résultat, la performance et le contrôle que les utilisateurs peuvent avoir sur la procédure.

1.2 Introduction

Les mondes virtuels 3D sont de plus en plus complexes. Cependant, le processus, les outils et les techniques de modélisation de ces mondes sont laborieuses et répétitives, nécessitant des qualités de spécialiste en modélisation graphique 3D. Pour la modélisation procédurale, la philosophie est au lieu de créer les contenus à la main, *créer ses contenus automatiquement.* Cette approche a été appliquée avec succès pour générer, par exemple, *les textures, les modèles géométriques, les animations et même les sons.* Un sujet majeur dans la modélisation procédurale est la **génération automatique de modèles de terrain**, qui a commencé par les *phénomènes naturels comme les élévation de terrain et la croissance des plantes* dans les années 80 et 90 puis a étendue son champ aux environnements urbains au début du nouveau millénaire.

En dépit de résultats encourageants, la modélisation procédurale n'est pas souvent appliquée à la modelisation entière de terrain (mainstream terrain modelling). Plusieurs facteurs limitent cette transition de la modélisation manuelle vers la modélisation automatisée. Parmi eux, les articles de recherche autant que les outils commerciaux se concentrent sur un seul aspect de la modélisation de terrain (mt), (par exemple, la génération de profils d'élévation d'intérêt) et gère les aspects restants à

un degré limité ou pas du tout. L'intégration et l'ajustement des méthodes procédurales (mp) pour qu'elles puissent automatiquement générer un mt complet et consistent demeurent jusqu'à maintenant irrésolus. Un autre problème connue des méthodes procédurales est le manque de contrôle qu'elles procurent. L'aléa inhérent au résultat obtenu force souvent les utilisateurs à modéliser par "trial and error" (generate and test). De récentes publications traite ce problème avec des solutions spécifiques, mais elle n'a pas assez reçu encore d'attention suffisante.

Cette note présente un court recensement des méthodes procédurales appliquées à la mt. Nous discutons sur les propriétés importantes des méthodes, comme le réalisme du résultat, la performance de l'algorithme et les facilités qu'il offre aux utilisateurs sur le contrôle du processus de génération. L'objectif de cette note est double : donner aux lecteurs intéressés par le domaine de la modélisation de terrain procédurale une vue d'ensemble des recherches faites jusqu'à maintenant, vérifier les possibilités que les problèmes connus liés aux modèles de terrain sont en cours de traitement et ce qui reste à faire.

Dans (Smellik et al., 2008, 2009), nous avons identifié plusieurs conditions à remplir qu'un framework de modélisation procédurale doit remplir pour être qualitativement acceptable et représenter une alternative productive au workflow de modélisation existant. Nous décrivons le design conceptuel d'un tel framework. Il intègre les modélisations procédurales et gère les dépendances entre les particularités des terrains dans le but de générer un modèle de terrain complet et consistent, qui correspond à un schéma grossier dessiné du terrain fait par l'utilisateur. Il concerne la distinction de plusieurs couches dans le modèle du terrain, chacune contenant des caractéristiques naturelles (terre, eau, couches de végétation) et faites par l'homme (route et couches urbaines). Nous présentons le design, l'implémentation et les résultats de deux de ces couches. Dans cette note, la distinction des couches du terrain est aussi pratique pour structurer le travail que nous recensons. Les sections suivantes discutent des méthodes procédurales pour les données d'élévation, les corps d'eau, la végétation, les réseaux routiers et les environnements urbains, suivi d'une concludion de l'état de l'art.

1.3 Height-maps

Height-maps, i.e. des grilles à deux dimensions contenant des valeurs d'élévation, sont souvent utilisés comme la base d'un modèle de terrain. Il y a beaucoup d'algorihtmes procédurales pour créer les height-maps.

Parmi les plus anciens algorithmes figurent les méthodes basées sur la subdivision. Un height-map grossier est itérativement subdivisé, chaque itération utilise un aléa controllé pour ajouter les détails. Miller (1986) décrit plusieurs variantes de cette méthode de déplacement de point de milieu (midpoint), dans laquelle un nouveau point d'élévation est assigné à la moyenne de ces coins dans un triangle ou une forme de diamant additionnée d'une correction aléatoire. L'intervalle de la correction diminue à

chaque itération dépendant d'un paramètre qui contrôle la rugosité du height-map résultant.

La génération height-map est de nos jours souvent basée sur les générateurs de bruits fractals (Fournier et al., 1982 ; Voss, 1985), comme le bruit de Perlin (Perlin 1985), qui génère des bruits en échantillonant et en interpolant des points de la grille de vecteurs aléatoires. Redimensionner et ajouter plusieurs niveaux de bruit pour chaque point du height-map conduit à des structures naturelles, pseudo montagneux. Un livre recommandé pour les bruits fractals et la génération de height-map et celui d'Ebert et al. (2003).

Ces height-maps peuvent être transformés sur la base de filtres communs d'image (ex. lissage) ou sur les simulations de phénomène physique, comme l'érosion. L'érosion thermique réduit les changements aigus pour l'élévation, en distribuant itérativement le matériau depuis les points les plus hauts vers les plus bas, jusqu'à ce qu'un angle de talus, i.e. un angle maximale de stabilité pour le matériau comme la pierre ou le sable, soit atteint. L'érosion causé par les pluies (érosion fluviale) peut être simulée en utilisant, par exemple, l'automate cellulaire, où la quantité d'eau et de matériel dissout qui s'écoule en dehors des autres cellules est calculée sur la base de la descente (pente) locale de la surface du terrain. Musgrave traite tous les types d'érosion (Musgrave et al., 1989 ; Musgrave, 1993) et Olsen (2004) discutent de plusieurs optimisations de calcul avec une qualité réduite mais acceptable. Benes et Forsbach (2001) introduisent une structure de terrain convenable pour des algorithmes d'érosion plus réaliste. Leur modèle de terrain consiste en des tranches de matériaux empilées horizontalement, chacune ayant des valeurs d'élévation et des propriétés des matériaux, ex. la densité. C'est une compromis entre la structure height-map limitée mais efficace et le terrain tout en voxel. Le modèle permet également d'avoir des couches d'air, par ce moyen il supporte des structure caverneux (des grottes).

Bien que ces algorithmes d'érosions ajoutent beaucoup de réalisme aux terrains montagneux, ils sont réputés lents, ayant besoin de faire des centaines, voire des milliers d'itérations. Une récente recherche s'est penchée sur des algorithmes d'érosion interractifs, souvent en portant l'algorithmes sur le GPU. Des exemples prométeuses comprennent (Anh et al. 2007) et (Stava et al., 2008).

La génération de height-map basée sur le bruit basique produit des résultats bien aléatoires ; les utilisateurs contrôlent le résultat seulement à un niveau global, souvent utilisant des paramètres non intuitifs. Plusieurs chercheurs se sont penchés sur ce problème. Stachniak and Stuerwlinger (2005) propose une méthode qui intègre les contraintes (exprimées comme des masques d'images) dans le processus de génération de terrain. Il emploie un algorithme de recherche qui cherche un ensemble d'opérations de déformation acceptable à appliquer à un terrain aléatoire dans le but d'obtenir un terrain qui est conforme à ces contraintes. Schneider et al. (2006) introduisent un environnement d'édition dans lequel l'utilisateur peut changer le terrain en modifiant interactivement les fonctions de base du générateur de bruits (en remplaçant le bruit de Perlin par un ensemble d'images en niveau de gris dessinées par l'utilisateur). Zhou et al. (2007) décrivent une technique qui génère le terrain basé

sur un exemple en entrée height-map et une ligne dessinée par l'utilisateur qui définie l'occurence de caractéristiques de ligne courbée en niveau de gris, comme une crête de montagne. Les caractéristiques sont extraites de l'exemple height-map et doivent correspondre aux courbes dessinées et être recousu sur le height-map résultant. De Carpentier et Bidarra (2009) introduisent les pinceaux procéduraux (procedural brushes) : les utilisateurs peignent le terrain en height-map directement en 3D en appliquant simplement les pinceaux élévante de terrain (terrain raising brushes) mais également des pinceaux basés sur le GPU qui génère plusieurs types de bruit en temps réel (voir Fig. 1a). Saunders (2006) propose un méthode très différente, laquelle synthétise un height-map en se basant sur des Modèles d'élévation digitale (ou DEM : Digital Elevation Models) du terrain réel. Un utilisateur de son système Terrainosaurus dessine un carte 2D de régions polygonales, chacune d'elle est marquée pour avoir un certain profile d'élévation. Pour le réalisme, les bords direct (straight boundaries) de la région sont perturbés puis rastériser dans une grille. Un height-map est instantié en utilisant un algorithme génétique qui sélectionne les données DEM qui correspondent au profil d'élévation requis. Kamal et Uddin (2007) présente un algorithme de déplacement en milieu de point sous contrainte (constrained mid-point displacement algorithm) qui crée une seule montagne correspondante à des propriétés comme l'élévation et l'étalement de la base. Belhadj (2007) introduit un système plus général où un ensemble de valeurs élévations connues contraignent le processus de déplacement en milieu de point (mid-point deplacement). Des applications possibles sont l'interpolation grossière et incomplète de DEM ou des lignes dessinées par l'utilisateur.

Une limitation inhérente aux height-map est qu'ils ne supportent pas les rochers surplombants et les grottes (cavités). Gamito et Musgrave (2001) proposent un système de gauchissement de terrain qui résulte à un surplomb régulier artificiel. Une méthode récente (Peytavie et al., 2009) procure une structure plus élaborée avec différentes couches de matériaux qui supportent les rochers, les arches, les surplombs et les grottes. Leurs modèles de terrain sont visuellement plausibles et naturels.

Comme illustration de l'état-de-l'art parmi les outils de support, (Voir Fig. 1d) pour un rendu d'un height-map généré par L3DT (Torpy, 2009), un des outils commerciaux pour la création de height-maps.

1.4 Rivières, Océans et Lacs

Pour générer des rivières, plusieurs auteurs ont proposés des algorihtmes qui s'exécutent soit pendant soit après la génération height-map. Kelley et al (1988) prennent un réseau de rivière comme la base d'un height-map. Ils commencent par une unique rivière bien droite et le subdivisent récursivement, produisant un réseau de courants. Cela forme un squelette pour le height-map, qui est complété en utilisant une fonction d'interpolation de données éparpillées. Le type de climat et le matériau du sol influencent la forme du réseau du courant.

Prusinkiewicz et Hammel (1993) combinent la génération d'une rivière courbée avec un schéma de subdivision de height-map. Du triangle de départ d'une rivière, un bord est marqué comme l'entrée et l'autre bord comme la sortie d'une rivière. Dans l'étape de subdivision, le triangle est divisé en triangles plus petites, et la course de la rivière à partir de l'entrée jusqu'à la sortie peut prendre maintenant plusieurs formes alternatives. L'élévation des triangles contenant la rivière est prise comme la somme des déplacements négatives de la rivière sur tous les niveaux de récursion (produisant le lit de la rivière), les autres triangles sont traités en utilisant le déplacement de milieu de point standard. Après huit ou plus récursions, la course de la rivière résultante parait raisonnablement naturelle. Un point négatif majeur de l'approche est que la rivière est placée à un niveau d'élévation constant et bas, et alors les cisailements s'enfoncent à travers un terrain montagneux de manière non naturelle.

Une approche plus avancée décrite par Belhadj et Audibert (2005) crée un height-map avec les crêtes de montagne et les réseaux de rivières. Commençant avec une carte vide, ils placent les paires de particules de crête sur une élévation particulièrement haute et les déplacent dans les directions opposées pour plusieurs itérations. Une courbe de Gauss est dessinée sur le height-map le long des positions de la particule pour chaque itération. Ensuite, ils placent les particules de la rivière suivant le sommet de la crête de la montagne et les laissent s'écouler vers le bas suivant des règles simples de la physique (comparable à l'érosion hydraulique). Le points restants entre les crêtes et les rivières sont complétés avec une technique de déplacement de point milieu inversé. Pour ce type de terrain spécifique, i.e. les crêtes de montagnes et les vallées irriguées par un réseau de rivière dense, la méthode est rapide et efficace.

A l'exception des rivières, les corps d'eau procédurale, comme les océans et les lacs et leurs connections, réseaux de courants, deltas et chutes d'eau, ont retenu peu d'attention jusqu'à maintenant. La formation des lacs n'est pas considéré du tout. Les océans sont communément générés en prenant un niveau fixe d'eau (ex. $0m$) ou en commençant par un algorithme d'inondation à partir des points à faible élévation. Teoh (2008) a également déclaré que la recherche sur ce domaine est incomplète : plusieurs rivières et caractéristiques côtières n'ont pas été traitées. Il propose des algorithmes simples et rapides pour les rivières sinueux, les deltas et la formation de la plage, qui requiert plus de travail pour augmenter le réalisme.

1.5 Les modèles de plantes et la distribution de la végétation

A propos de la végétation, des auteurs ont développé plusieurs procédures de génération d'arbres et de modèles de plantes et des méthodes de placement automatique de la végétation dans un modèle de terrain. Les premières sont utilisées pour obtenir rapidement un ensemble de modèles de plantes similaires mais variables d'une même espèce ; les secondes délestent les modeleurs de terrain de la tâche laborieuse de placer manuellement tous ces modèles de végétation individuels dans une forêt large.

Les modèles de plantes procéduraux croissent, en commençant à la racine, ajoutant de plus en plus de petites branches et terminant par les feuilles. Ils sont fondés sur la grammaire formelle (grammar rewriting). Prusinkiewicz et Lindenmayer (1990) discutent du Lindenmayer-system, ou *L-system*, un système de réécriture souvent utilisé. Ils expliquent comment les règles de production peuvent être appliquées en 3D, et présentent plusieurs exemples d'arbres générés ensemble avec leur grammaire.

Lintermann et Deussen (1999) proposent un système plus intuitif pour modéliser procéduralement les plantes, en plaçant les composantes de la plante (ex. une feuille) dans un graphe. Les composantes connexes peuvent être structurées dans un sous-graphe (ex. une brindille). Le système parcours le graphe, générant et plaçant les instances des composantes dans un graphe intermédiaire qui est utilisé pour la génération géométrique. La figure 1e) montre un arbre créé avec leur logiciel de modélisation de plante commercial XFrog.

Deussen et al. (1998) décrivent un modèle de simulation d'écosystème pour remplir une surface de végétation. L'entrée pour le modèle de simulation est le height-map et une carte de l'eau, plusieurs propriétés d'espèces de plantes, comme le taux de croissance, et, optionnelement une distribution initiale de plantes. Basé sur cela et en prenant en compte les règles sur la compétition du sol, des rayons de lumière et de l'eau, une distribution de plantes à l'intérieur d'une surface est itérativement déterminée (voir Fig. 1b), s'exécutant pendant plusieurs minutes.

Une autre procédure de placement de la végétation par Hammes (2001) est fondée sur les écosystèmes. Il utilise des données d'élévation, des élévations relatives, des pente, direction de la pente et des bruits multi fractals pour sélectionner un des écosystèmes définis. Les textures des végétations du sol sont générés à la volée, dépendant du niveau de détails et de l'écosystème. L'écosystème détermine également le nombre de plantes par espèce, lesquelles sont placées aléatoirement.

La modélisation procédurale de la végétation produit des résultats fiables et sont déjà bien appliquées dans les jeux vidéos modernes, par exemple en utilisant le package commercial SpeedTree.

1.6 Les réseaux routiers

La génération de réseaux routiers pour les villes peut être faite par des méthodes variées, parmi lesquelles nous traitons les approches basées sur les pattern, L-systems, les simulations d'agents et les champs tensoriels. La plus simple technique est de générer une grille de carré dense (comme dans Greuter et al. (2003)). Le bruit de déplacement peut être ajouté aux points de la grille pour créer un réseau moin répétitif, mais le réalisme de cette technique est toujours limité.

Une méthode plus élaborée pour créer des routes est par le biais des modèles (ou "templates"), comme proposés par Sun et al. (2002). Ils observent plusieurs patterns fréquent dans les réseaux routiers réels et tentent de les reconstruire. Pour chaque pattern, il y a un template correspondant : un template basé sur la population (implémenté comme un diagramme de Voronoi des centres de population), un raster et un template radial, ou une template mixte. Les principales artères de la carte de la route sont les autoroutes, qui sont générées en premier en utilisant ces templates de pattern. Des règles simples sont appliquées pour vérifier leur validité. Quand des zones d'impasse sont rencontrées (ex. les océans), elles sont annulées ou déviées. Ensuite, les routes principales sont souvent courbées pour éviter les gradients d'élévation large. Les régions qu'elles englobent sont complétées avec un raster de rues.

Parish et Müller (2001) utilisent un L-system étendu pour faire croître leur réseau routier. Le L-system est conduit par des objectifs précis (goal-driven) et les objectifs en question sont la densité moyenne de la population (les rues tentent de connecter les centres de population et les patterns spécifiques de routes). Des exemples de tels patterns sont le raster et le pattern radial. Leur L-system est étendu avec des règles qui ont une tendance à connecter les routes nouvellement proposées avec les intersections existantes et des règles qui vérifient la validité de la route en respectant le terrain impassable et les contraintes d'élévation. Les rues sont aussi insérées dans les zones restantes comme des grilles simples.

Kelly et McCabe (2007) introduisent l'éditeur de ville interactive CityGen, dans lequel un utilisateur défini les routes principales en plaçant des noeuds dans le terrain 3D. Les régions inclues par ces routes peuvent être complétées avec une de ces trois patterns : les grilles style Manhattan, les routes de croissance industrielle avec des routes sans issues et des routes organiques comme par exemple les suburbaines nord-américaines.

Glass et al. (2006) décrivent plusieurs expérimentations de réplication de structures de route rencontrées dans les colonies informelles d'Afrique du Sud en utilisant la combinaison d'un diagramme de Voronoi pour les routes majeures avec des L-systeme ou des subdivisions régulières avec ou sans bruit de déplacement pour les routes mineures. Ils ont raisonnablement réussi à recréer les patterns observés.

A la différence des approches basées sur les grammaires ou les patterns ci-dessus, Lechner et al. (2003) prennent une approche basée sur l'agent, dans laquelle ils divisent la ville en zones incluant non seulement les zones résidentielles, commerciales ou industrielles, mais également des zones spéciales comme les bâtiments gouvernementaux, les squares, et les institutions. Ils placent deux agents, appelé l'*extender* et le *connector*, sur une position de semence dans la carte du terrain. L'extender recherche les zones non connectées dans la ville. Quand il trouve une telle zone qui est localisée non loin d'un réseau routier existant, il cherche le chemin le plus convenable pour connecter cette zone au réseau routier. Dans Lechner et al. (2006), les auteurs étendent cette méthode avec, entre autres choses, des agents qui développent les petites rues. Cette méthode donne des résultats plausibles, mais son désavantage

est son temps d'éxectution long.

Chen et al. (2008) proposent une modélisation interactive de réseaux routier en utilisant les champs tensoriels. Ils définissent comment créer des patterns communs de routes (grille, radial, suivant les bords) en utilisant des champs tensoriels. Un réseau routier est généré à partir d'un champ tensoriel, en traçant des lignes aérodynamiques à partir des points de semence dans les directions majeures des vecteurs propres jusqu'à ce qu'une condition d'arrêt soit remplie. Ensuite, suivant la courbe tracée, de nouveaux points de semence sont placés pour tracer les lignes aérodynamiques dans la direction perpendiculaire (vecteur propre mineur). Des utilisateurs peuvent placer des bases de champs tensoriels, comme un pattern radial, lisser le champ, ou utiliser un pinceau pour contraindre localement le champ vers une direction spécifique. Du bruit peut être appliqué pour rendre le réseau routier moins régulier et en conséquence plus plausible.

Dans les méthodes discutées, l'influence de la carte du terrain sous-jacente et du profil d'élévation est à degré variable prise en compte. La plupart des méthodes prennent seulement des mesures basiques pour éviter des routes montantes avec des pentes trop raides et des routes qui traversent des corps d'eau. Kelly and McCabe (2007) planifient le chemin précis de leurs routes principales entre les noeuds mis en place par l'utilisateur pour avoir autant de possibilités dans l'élévation que possible. Seulement, pour les terrains durs cette mesure ne sera pas adéquate et le terrain doit être modifié pour l'accomodation de la route. Bruneton and Neyret (2008) proposent une méthode simple et efficace pour mixer les profiles des routes dans les height-map en utilisant les shaders.

1.7 Les environnements urbains

Kelly and McCabe (2006) donnent un point de vue élaboré de plusieurs approches pour la génération d'environnement urbains. Watson et al. (2008) donne un point de vue pratique de l'état de l'art.

L'approche la plus commune pour générer des villes procéduralement est de commencer par un réseau de routes dense et identifier les régions polygonales inclues dans les rues. La subdivision de ces régions produisent des lots, pour lesquels différentes méthodes de subdivisions existent, voir par ex. Parish and Müller (2001) ou Kelly and McCabe (2007). Pour remplir ces lots avec des immeubles, soit la forme du lot est utilisée directement comme empreinte d'un immeuble, ou l'empreinte d'un immeuble est adapté pour le lot. En faisant simplement l'extrusion de l'empreinte jusqu'à une hauteur aléatoire, on peut générer une ville avec des gratte-ciels ou des immeubles de bureau. Pour obtenir des formes d'immeubles intéressantes, plusieurs approches ont été mises en oeuvre.

Greuter et al. (2003) génère des immeubles de bureau en combinant plusieurs formes primitives dans un plan d'étage et en faisant l'extrusion de cette forme pour différentes hauteurs. Parish and

Müller (2001) commençent par un plan d'étage rectangulaire et appliquent un L-system pour affiner l'immeuble. Ces approches sont surtout utiles pour des modèles simples d'immeubles de bureau.

Wonka et al. (2003) introduisent le concept de *split grammar*, une grammaire formelle indépendante du contexte (context-free formal grammar) dont le design est fait pour produire des modèles d'immeubles. Le split grammar ressemble à un L-system, mais il est bas" sur des formes comme éléments primitifs à la place des symboles. Dans leur système, un style spécific d'immeuble peut être acquise en prennant un attribut au début du symbole, lequel est propagé pendant la réécriture. Dans un modèle d'immeuble, le style peut être différent pour chaque étage (ex. un immeuble d'appartement avec des boutiques au le rez-de-chaussé). Leur approche repose surtout sur la génération cohérente et plausible des façades pour des immeubles au forme relativement simple. Larive and Gaildrat (2006) utilise un grammaire de la même sorte, appelé *wall grammar*. Avec cette grammaire, ils sont capables de générer des murs d'immeuble avec des détails géométriques additionnels, comme les balcons.

Müller et al. (2006) appliquent une autre type de grammaire, appelée *shape grammar*. La propriété principale d'un shape grammar est qu'il utilise des règles dépendant du contexte (context-sensitive), tandis qu'une split grammar utilisent des règles indépendantes du contexte (context-free), qui dans ce cas permet la possibilité de modéliser les toits et les formes arrondies (rotated shapes). Ils commencent avec une réunion de plusieurs formes volumétriques lesquelles définissent les bords des immeubles. Cette forme est ensuite divisée en étages et les façades produites sont subdivisées en murs, fenêtres, et portes par le moyen d'un système de grammaire. Dans une étape finale, le toit est construit sur le sommet de l'immeuble. Fig 1f montre un réseau routier et Fig 1g la ville correspondante générée par leur produit commercial, CityEngine (Procedural, inc. 2009). En plus des immeubles d'affaire bien connus, la grammaire peut aussi modéliser des immeubles résidentiels, ex. les maisons suburbaines ou les anciennes villas romaines.

Bien que les shape grammars dans Müller et al. (2006) peuvent générer des modèles d'immeubles visuellement convaiquants, Finkenzeller and Bender (2008) notent que l'information sémantique à propos du rôle de chaque forme dans l'immeuble final est manquante. Ils proposent de capturer cette information sémantique dans un graph typé. Leur workflow comprend trois étapes. En commençant par des traits de contour brute, un graphe de style d'immeuble peut être appliqué à ce modèle. Cela produit une représentation intermédiaire de graphe sémantique de l'immeuble, laquelle peut être modifiée ou regénérée avec un style différent. Dans la dernière étape, la géométrie est crée sur la base du modèle intermédiaire, et les textures sont appliquées, produisant l'immeuble 3D final. Finkenzeller (2008) décrit avec plus de détails la génération des façades et des toits de leur système (voir fig. 1c).

Young et al. (2004) décrivent une méthode pour créer des maisons au style vernaculaire du sud-est de la Chine en utilisant un shape grammar étendue. La grammaire est hierarchique et commencent au niveau de la ville (tandis que dans d'autres méthodes un shape grammar est appliqué à l'empreinte

d'un immeuble individuel). La grammaire produit ensuite les rues, les blocs d'habitation, les routes, et même les maisons de productions avec des composantes comme les portails, les fenêtres, les murs et les toîts. A travers des règles de contrôles (définissant par exemple, les contraintes de ratio de composants) la validité des immeubles peut être testée. En appliquant ce système de grammaire, une ville ancienne typique du sud-est de la Chine peut être généré avec des résultats plausibles, puisque les style d'immeuble de ces villes est très rigidement structuré.

Müller et al. (2007) utilisent une approche très différente pour la construction des façades des immeubles. Leur méthode prend un image unique de la façade de l'immeuble réel comme entrée et est capable de reconstruire un modèle détaillé de la façade en 3D, en utilisant une combinaison d'image et de génération de shape grammar.

Bien que les méthodes ci-dessys donne rapidement et visuellement des résultats plaisants, les villes qu'elles génèrent manque de structure réaliste. Une nouvelle recherche incorpore les théories et les modèles d'aménagement de territoire urbain existantes dans le processus de génération. Groenewegen et al. (2009) présentent une méthode qui génère une distribution de différents types de districts selon les modèles d'aménagement urbain dans l'Europe de l'Ouest et l'Amérique du Nord. Ils prennent en compte un grand éventail de facteurs significatifs, incluant le plan historique de la ville et de l'attraction que certains types de terrain (côteaus, océans, rivières) peuvent avoir par ex. les districs industriels ou résidentiels de grande classe. Weber et al. (2009) utilisent des modèles comparables (comparaison un peu simplifiée) pour une simulation d'une ville dans le temps. Leur système est rapide (environ 1s pour un an de simulation) et interactif, ce qui veut dire que l'utilisateur peut guider la simulation en changeant les routes ou en mettant une peinture pour le sol utilisé sur le terrain.

1.8 Conclusions

Les méthodes procédurales pour la modélisation de terrain deviennent de plus en plus attractif pour l'industrie et le domaine académique. Nous avons classifié ces méthodes dans cinq domaines principales, et discuté d'une grande variété d'approches de recherche et des résultats obtenus grâce à elles : l'élévation de terrain, les éléments d'eau, la végétation, les réseaux routiers et les environnements urbains.

Depuis les tous débuts, où l'intérêt se focalisait surtout sur la génération de height-map, jusqu'à maintenant, avec un déplacement vers de plus en plus d'environnements urbains réalistes, il y a un considérable corps de résultats disponibles. Beaucoup de méthodes procédurales basiques déploient des blocs communs de construction comme le bruit, la réécriture formelle ou les simples systèmes de simulation, pour chacune un large nombre de variantes, souvent spécifique à un domaine, sont de nos jours proposées, en particulier pour la route et les catégories urbaines. Parmi toutes les catégories discutées, les zones relatives à l'eau sont clairement les plus sous-développées.

Concernant les recherches à venir, plusieurs intéressantes tendances ont été identifiées. Parmi elles, trois directions prometeuses peuvent être résumées comme suit. Premièrement, la performance et l'intéractivité des méthodes procédurales continuent de progresser, souvent au moyen de programmation parallèle sur le GPU. Deuxièmement, les réseaux routiers et les zones urbaines continueront certainement de s'améliorer dans le sens de la variété et des niveaux de détails, mais le bond du réalisme sera fourni probablement par le déployement de plus de sémantiques autant dans le processus de génération procédurale que dans les modèles générés (Tutenel et al., 2008). Et enfin, la clé du déploiement répandu des méthodes procédurales par les non-experts (ex. les designers de jeux, les artistes, les designers de scénarios) sera l'intégration des méthodes procédurales dans un framework, offrant entre autre chose, plus de contrôles intuitifs, d'outils pour générer des modèles complets de terrain et des mécanismes non intrusives pour le maintien de la consistence parmi les caractéristiques générés (Smelik et al., 2009)

Deuxième partie

Génération automatique d'environnements virtuels urbains (2016) de Tiavina NIVOLALA

Résumé

La génération automatique d'environnements virtuels urbains trouve une application dans la planification urbaine.

Deux approches de génération :
— la génération direct à partir d'un modèle statique
— la génération procédurale
La méthode de Tiavina est une combinaison de ces deux approches. En deux étapes :
i) créer un *modèle de base de données hiérarchisée* contenant les *éléments fondamentaux d'un environnement urbain*
ii) appliquer des techniques procédurales pour générer automatiquement un environnement urbain à partir du *style architectural* correspondant à des éléments de la base nouvellement créée

Introduction générale

A partir des villes existantes, on a simulé les villes virtuelles. Les villes virtuelles sont utilisées et représentent des plateformes pour l'intégration d'information.

La visualisation et l'exploration des paysages urbains pour assister à :
— la planification urbaine
— le contrôle de trafic
— la gestion de désastres
— la création de mondes virtuels
— la réalité augmentée (superposition au monde réel)
Les villes ont des structures complexes d'où le besoin d'avoir un *modèle de données normalisé* pour obtenir une **structuration de données cohérente et interopérable ?**.

L'augmentation de la qualité des jeux vidéo à base de polygones en conjonction avec l'augmentation des performances des cartes graphiques demandent et permettent la **visualisation de grande quantité de données**. Il n'est pas commode de créer manuelle de grandes quantités de modèles, d'où le besoin de techniques procédurales. Sinon construite à la main, la construction de la ville virtuelle prendrait d'énorme délai.

La génération des objets avec une approche procédurale est aggrémentée de caractère aléatoire contrôlable. Le souci est qu'elle produit des résultats imprévisibles en forme et en apparence.

Chapitre 1 :

— Aperçu des différents standards et modèles déjà crées pour le stockage et la représentation des villes

— Quelques techniques de visualisation basés sur des modèles déjà existants

— Méthodes de visualisation purement procédurales.

Chapitre 2 : son approche pour la création du système

— La conception et l'alimentation de la base

— La génération des villes à partir de cette base

— Méthodes de visualisation purement procédurales.

Chapitre 3 :

— Outils utilisés pour la création du système

— Interface du système

Chapitre 2

Etat de l'art

2.1 Introduction

Generation de villes :

— depuis modèle existant → normes de représentation → représentation du modèle → visualisation (lecture du modèle)
— depuis photo (aériennes ou terrestres)
— de façon procédurale (sans modèle)

En deux parties :
— les standards dans la modélisation des villes
— les techniques de visualisation de ces dernières

2.2 Représentation des bâtiments et des villes

Plusieurs sources (modèles, procédures) ⇒ besoin d'adoption d'une forme de représentation.

Objectifs :
× décrire (représentation)
× gérer
× stocker

2.2.1 IFC

Format neutre ? pour :
× décrire + échanger des infos de construction de bâtiment
× norme pour openIBM ?

× objectif : fournir une base universelle pour échanger des données concernant les bâtiments
× orienté BdD, sur des définitions de classes d'objets (éléments, processus, formes) pour les projets de construction (ex. d'entités : *IfcWall*, de repr. de géométrie : *IfcExtrudedAreaSolid*
× IFC structure qui renfèrme des espaces : *IfcSpatialStructureElement* ⇒ résultent en sites/buildings/étages (⟵ dérive) *IfcProduct* ⇒ a un lieu, une représentation géométrique (modèles solides)

2.2.2 CityGML

Vue d'ensemble

Standard d'échanges et de représentation de données 3D Pour les modèles virtuels de villes et de paysages

× infos : géométrie/apparence/sémantique/topologie des objets
× Basé sur XML, ISO 19100 ? ?, Version 3.1.1 de Geography Markup Language (GML 3.1.1)
× Norme de l'Open Geospatial Consortium (OGC)

Modularisation de l'info géospatiale urbaine

Modules

Ville complexe ≈ Couverture de plusieurs thématiques des objets urbains CityGML rassemble :
× Modèle de base : concepts et composantes basiques
× Module d'ext. thématique : dépend du module de base

Les onzes modules d'extension thématiques :
× Building
× Apparences ? ?
× CityFurniture ? ?
× CityObjectGroup ?
× GenericCityObject ?
× Landuse
× Relief
× Transportation
× Vegetation
× WaterBody
× TexturedSurface