REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mohamed Khider – BISKRA
Faculté des Sciences Exactes, des Sciences de la Nature et de la Vie
Département d'informatique

N° d'ordre : *IVA 4* /M2/2019

Mémoire

présenté pour obtenir le diplôme de master académique en

Informatique

Parcours : Images et vie artificielle

Modélisation procédural de monde virtuel

Par :

MOHAMMEDI ABDELRAOUFE

Soutenu le 07 juillet 2019, devant le jury composé de :

Mr.NourEddine DJEDI	**Professeur**	**Président**
Mr.Belaiche Hamza	**M A A**	**Rapporteur**
Chighoub Fouzia	**M A A**	**Examinateur**

Remerciement

En préambule à ce mémoire nous remerciant ALLAH qui nous aide et nous donne la patience et le courage durant ces langues années d'étude. Nous tenant à remercier sincèrement Monsieur, Belaiche Hamza, qui , entant que Directeur de mémoire, s'est toujours montré à l'écoute et très disponible tout au long de la réalisation de ce mémoire, ainsi pour l'inspiration, l'aide et le temps qu'il a bien voulu nous consacrer et sans qui ce mémoire n'aurait jamais vu le jour.

Je remercie également les membres du jury:

NourEddine DJEDI Professeur

Chighoub Fouzia M A A

D'avoir accepté l'évaluation de ce travail. J'exprime ma profonde reconnaissance à tous ceux qui m'ont aidé à l'élaboration de ce mémoire

Abdelraoufe mohammdi

Dédicace

Je dédie ce travail à mes parents et à ma famille et en particulier à ma mère qui, après Dieu Tout-puissant, est la cause de mon succès et je n'oublie pas mes amis qui m'ont aidée dans ce travail humble et mes amis de l'étude et tous ceux qui me connaissent de près ou de loin.. j'ai l'intention de faire de ce modeste travail une œuvre de charité pour ma mère, mon père, ainsi que le candidat du professeur "Ouannes Nesrine" et moi-même.

Abdelraoufe mohammdi

Résume

Un monde virtuel est un monde créé artificiellement par un logiciel. La représentation de ce monde et de ses habitants peut être en deux ou en trois dimensions (2D/3D). Il existe plusieurs techniques pour générer un monde virtuel, mais dans notre étude nous avons concentré sur modélisation procédurale qui correspondent à nos besoins ; un temps de calcule réduit (temps réel) et des résultat inattendues (aléatoires).

Notre étude concerne la modélisation procédurale des mondes virtuel. Un monde contient beaucoup d'éléments qui ne peuvent pas être couverts pendant cette étude, nous avons concentrer sur les terrains et les plantes qui ce sont les deux éléments les plus importants du monde virtuel et qui jouent un rôle fondamental dans la création d'une scène naturelle.

Dans ce mémoire, nous avons fait une étude globale sur le domaine de la modélisation procédurale. Nous avons choisi de créer un terrain par la technique procédurale la plus courante basée sur l'algorithme Perlin et de générer les plantes en utilisant le L-système, pour obtenir notre monde Virtual.

Table des matières

Table des matières

Table des figures

Table des figures

Liste des tableaux

Introduction générale

Il y a déjà 44 ans que les premiers dessins par ordinateur ont été produits notamment pour le système de défense et de contrôle aérien SAGE et au M.I.T. avec l'ordinateur TX1. Pendant plus de 39 ans, certains scientifiques ont utilisé la capacité des ordinateurs pour produire des diagrammes et des graphes dans leurs rapports, thèses et articles. Pendant toute cette période, le public, même le public averti tel que la communauté universitaire, ignorait littéralement les possibilités graphiques de l'ordinateur. Il faut tout de même convenir que le matériel était encore cher, pas toujours très maniable et les résultats pas toujours très spectaculaires [32].

Aujourd'hui, la situation a radicalement change, tous les micro-ordinateurs personnels ont des capacités graphiques. Les millions de téléspectateurs des pays occidentaux sont envahis par les génériques et logos produits par ordinateur. On est d'ailleurs parfois ébahi par le réalisme et la perfection de certaines images générées par ordinateur. Que ce soit dans le domaine technique, médical ou simplement artistique, ces images forcent notre admiration et nous questionnent. Dans le domaine de la synthèse d'images réalistes, en moins de dix ans, on a assiste a des progrès spectaculaires, autant du point de vue matériel que du point de vue logiciel. Il faut d'ailleurs remarquer que leur évolution est intimement liée. Il serait difficilement concevable de produire des images réalistes sans disposer d'un terminal graphique de haute résolution avec au moins quelques centaines de couleurs affichables simultanément [32].

Plus récemment, c'est le domaine du multimédia qui s'est développe grâce aux progrès fulgurant dans les télécommunications. Avec Internet et World Wide Web, on peut consulter ou échanger des images créées a l'autre bout du monde. Par la réalité virtuelle, on peut plonger aussi dans des mondes virtuels et de nouveau, grâce aux télécommunications rapides comme les réseaux ATM, il est même possible de partager les mondes virtuels entre des participants du monde entier [32].

Le progrès spectaculaires de matériel ne suffit pas il faut aussi un progrès logiciel.

La conception d'une application graphique a affichage 3D suit plusieurs étapes :

- détermination des scènes devant être représentées,
- modélisation géométrique de ces scènes,
- placement des textures et des lumières,
- programmation.

Pour avoir ce progrès logiciel, il faut apporter des optimisations dans les algorithmes utilisés dans chaque étape de conception. L'optimisation généralement réduit l'efficacité, alors il faut chercher des compromis.

Un monde virtuel est une entité créé artificiellement par un logiciel, la représentation de ce monde et de ses habitants peut être en deux dimensions (2D) ou en trois dimensions (3D) contenir le terrain et la végétation et les masses d'eau, les réseaux de routes, la configuration urbaine, les bâtiments et les intérieurs de bâtiment …ect**.**

Dans notre étude nous allons concentrer sur les terrains et les plantes qui ce sont les deux éléments les plus importants du monde virtuel et qui jouent un rôle fondamental dans la création d'une scène naturelle.

La modélisation procédurale est un terme générique pour un certain nombre de techniques, utilisé pour la génération de modèle. Cela peut aller des jeux vidéo (infographie, terrain de jeu, personnages impliqués dans le jeu, etc.), dans l'art génératif (ou art procédural), ou bien encore dans l'ingénierie (organisation, production de formes optimales, comme un profil d'aile, etc.).

Parmi les plus connues, on peut citer les fractales (mouvement Brownien fractionnaire) en général pour simuler la nature, les L-Systèmes, une variante des fractales s'appliquant

notamment aux plantes, le diagramme de Voronoï permettant une répartition réaliste, telle des bulles à la surface d'un liquide, etc.. Plus récemment, l'apprentissage profond, et ses variantes spécialisées est une technique permettant d'apprendre les processus de création d'artistes et de les imiter.

Objectif : Notre travail concerne l'étude du génération des terrains et les plantes qui sont considérés comme des objets naturels, cette problématique intéresse de plus en plus les laboratoires de recherche en synthèse d'images. Néanmoins, la qualité visuelle qu'ils offrent est très souvent associée a une complexité dans la prise en charge des aspects modélisation.

Notre objectif principal est de choisir les algorithmes adéquats pour générer un monde virtuel en utilisant les méthodes procédurales, permettant de réaliser la simulation la plus complète possible.

Axes du mémoire : Pour assurer les objectifs de notre travail, nous organisons ce mémoire

en cinq chapitres :

- Chapitre 1 " **Modélisation procédurale**" Dans ce chapitre, nous allons examiner les catégories et les méthodes procédurale

- Chapitre 2 " **Modélisation procédurale de terrain** " : Dans ce chapitre, nous allons examiner quelques algorithmes pour la génération des terrains, ces algorithmes permettent de produire un modèle numérique pour notre terrain et nous allons exposer une technique d'habillage de terrain, qui consiste a colorer le terrain en espérant obtenir un résultat suffisamment réaliste.

- Chapitre 3 " **Modélisation procédurale de plante** " Dans ce chapitre, nous allons examiner quelques algorithmes pour la génération des plantes.

- Chapitre 4 " **Conception** " : Dans ce chapitre nous allons présenter la conception de l'application, la conception globale et la conception détaillée de l'application.

- Chapitre 5 ” **Implémentation (mise en œuvre)** ” : Dans ce chapitre nous allons découvrons les différents outils que nous avons utilisés et la mise en œuvre de notre approche, et présenté les résulta.

Chapitre 1 : Modélisation procédurale

1. Introduction :

Les mondes virtuels d'aujourd'hui défient la capacité de la création humaine. Essayer de reproduire des scènes naturelles, avec des modèles volumineux et complexes, implique de reproduire leur complexité et leurs détails inhérents. La génération procédurale aide en permettant aux artistes de créer et de généraliser des objets pour des scènes très détaillées.
La génération procédurale est une catégorie des techniques de modélisation (les deux autres étant : la simulation informatique et la modélisation par esquisses ou par édition) permettant la création d'un monde virtuel cohérent, Dans ce chapitre nous allons d 'écrire les méthodes de procédural.

2. Définition d'un monde virtuel :

C'est un monde créé artificiellement par un logiciel informatique et pouvant héberger une communauté d'utilisateurs présents sous forme d'avatars ayant la capacité de s'y déplacer et d'y interagir. La représentation de ce monde et de ses habitants est en deux ou en trois dimensions**.**Un monde virtuel peut contenir le terrain et la végétation et les masses d'eau, les réseaux de routes, la configuration urbaine, les bâtiments et les intérieurs de bâtiment …ect

Dans notre étude nous allons concentrer sur **les terrains** et **les plants**

3. Structure générale de système infographiste

Nous pouvons classer les techniques existantes en trois grandes catégories (Figure 1.1) : les méthodes d'édition et d'esquisse, les techniques de simulation et la génération procédurale. Certaines méthodes sont parfois à l'intersection de différence catégories et combinent plusieurs avantages pour s'adapter aux contraintes de modélisation fixée par le problème posé.

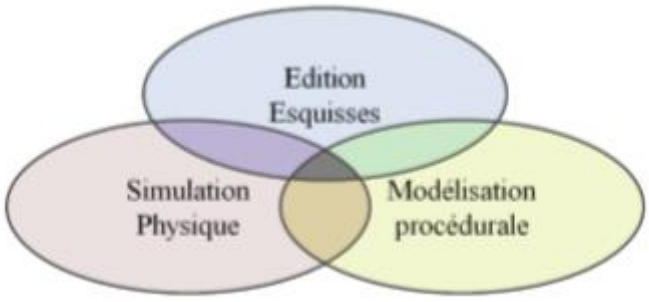

Figure 1.1 : Techniques de modélisation des éléments.

Les méthodes à base d'édition ou d'esquisses permettent de contrôler précisément la forme des éléments. Elles consistent à en modier itérativement la surface an de contraindre l'objet vers une forme souhaitée. Ce type de modélisation demande une forte intervention de l'utilisateur et s'avère longue et coûteuse lors de la création d'un grand nombre d'objets. Malgré ces inconvénients, cette méthode est la plus utilisée dans les industries du lm et des jeux vidéo. La génération procédurale

permet de modéliser rapidement et automatiquement un nombre important d'éléments en dénissant un ensemble de procédures et avec une intervention de l'utilisateur réduite. Ces procédures sont contrôlées par des paramètres soit dénis en entrée par un utilisateur, soit calculés aléatoirement dans un intervalle également spécié par l'utilisateur.

enfin , les méthodes de simulation physique permettent de modéliser un élément évoluant au cours du temps. Ces méthodes reproduisent des phénomènes physiques, chimique, biologique appliqués à un objet pour obtenir un résultat réaliste. Le résultat nal ne peut pas être contrôlé car il dépend uniquement des équations physiques. La création d'un élément est long et coûteux car elle nécessite de résoudre de nombreuses équations physiques. La plupart des méthodes existantes sont aux frontières de ces catégories an d'en combiner les diérents avantages.[1]

4. Définition de la modélisation procédurale (PM)

La modélisation procédurale traite de la génération de contenu (semi) automatique par le biais d'un programme ou d'une procédure. Entre autres avantages, sa compression des données et son potentiel de produire une variété de détails l'intervention humaine au rendu .La modélisation procédurale attrayante pour un environnement virtuel et devenu de plus en plus utilisée dans les films, les jeux et les simulations. Nous examinons les méthodes procédurales qui sont utiles pour générer des caractéristiques de mondes virtuels, notamment des terrains, de la végétation, des rivières, des routes, des bâtiments et des villes entières. [2] La modélisation procédurale est une technique d'infographie conçue pour réduire la puissance de calcul nécessaire au rendu d'une scène géométriquement complexe. Les techniques de rendu traditionnelles extraient chaque élément de scène d'une base de données au moment de l'exécution et l'insèrent dans la scène à restituer. Cela signifie que chaque élément doit être présenté en entier pour montrer la scène. La modélisation procédurale vise à résoudre ce problème uniquement.

5. Catégorisation des méthodes procédurales :

En Cette partie comporte un aperçu des méthodes existantes de gestion des particules pour la génération d'éléments spécifiques du monde virtuel, tels que le terrain, la végétation, les masses d'eau , les réseaux de routes, la configuration urbaine , les bâtiments et les intérieurs de bâtiment

Le tableau 1.1 présente une classification des méthodes de gestion des particules discutées dans la présente enquête, classées en fonction de leurs principales caractéristiques d'application et de la famille ou de la catégorie de techniques sous-jacentes. Nous avons opté pour l'identification de six de ces catégories, représentées par les colonnes du tableau 1.1, bien que toutes les méthodes de MP (Modélisation procédurale) ne s'intègrent pas parfaitement dans ce schéma, car elles peuvent, dans certains cas, combiner des techniques de plusieurs familles.

	principales caractéristiques d'application et de la famille	**Des éléments spécifiques du monde virtuel appartenant à cette famille**
Stochastique	Certaines approches procédurales sont basées sur des méthodes puréli stochastiques; la plupart d'entre eux utilisent des générateurs de bruit pour créer des motifs ou des structures fractales. Un représentant typique de ces techniques est le travail de Mus grave et al. Utilisant fBm pour créer des terrains ou des textures.	Le terrain- les masses d'eau - les réseaux de routes- la configuration urbaine -les intérieurs de bâtiment
Intelligence artificielle	Les techniques d'intelligence artificielle, telles que les algorithmes de détermination de trajectoire ou de planification, complètent divers algorithmes procéduraux.	le terrain- la végétation- les masses d'eau- les réseaux de routes- la configuration urbaine - les bâtiments
Simulations	Les simulations sont à la limite de méthodes purement procédurales et de certains autres domaines tels que la physique, la modélisation urbaine ou la biologie.	le terrain la végétation les masses d'eau les réseaux de routes la configuration urbaine
Grammars	Les méthodes basées sur la grammaire peuvent être considérées comme des approches procédurales pures et leurs applications peuvent être trouvées dans presque tous les domaines d'application. Les deux approches les plus importantes sont les systèmes L, et les grammaires de forme	le terrain la végétation les masses d'eau les réseaux de routes la configuration urbaine

Géométrie computationnelle	Diverses autres méthodes utilisent des algorithmes de géométrie de calcul (une combinaison de), directement à l'aide d'un langage de programmation. Il s'agit souvent d'algorithmes de subdivision, de méthodes de remplissage d'espace ou de techniques inspirées du traitement d'images. Ces méthodes peuvent en outre être classées en division, en expansion (croissance) et en optimisation	le terrain, la végétation les masses d'eau les réseaux de routes la configuration urbaine les intérieurs de bâtiment
Axé sur les données	On peut également comparer les modèles de procédure du point de vue des données qu'ils traitent. Par exemple: les générateurs de bruit peuvent fonctionner avec des ensembles de données pratiquement arbitraires; les générateurs de terrain sont généralement limités à des MNT, des maillages ou des représentations volumétriques; et des algorithmes plus spécialisés, par exemple pour la construction de structures, peuvent nécessiter des données spécialisées avec lesquelles travailler. Après les lignes du tableau 1, il est à noter que beaucoup d'attention a déjà été portée aux terrains végétation et aménagement de la ville. Plus récemment, les bâtiments ont reçu plus d'attention et de nombreuses nouvelles approches apparaissent chaque année.	le terrain la végétation les masses d'eau les réseaux de routes la configuration urbaine les bâtiments

Tableau 1.1: Classification des méthodes de gestion des particules

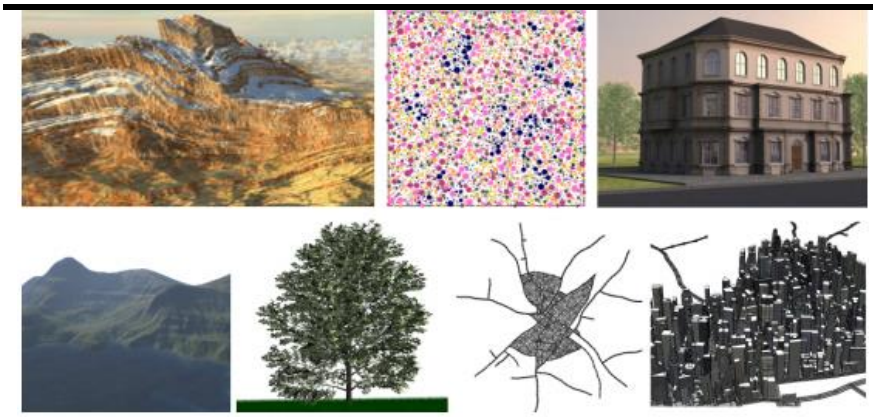

Figure 1.2: monde virtual procedural(a) Terrain interactively created using procedural brushes (De Carpentier and Bidarra, 2009). b) Plant distribution simulation (Deussen et al., 1998). c) Complex building facade (Finkenzeller and Bender, 2008). d) A 3D render of a height-map generated using L3DT (Torpy, 2009). e) A plant model generated using XFrog (Greenworks, 2009). f) and g) A road network and corresponding city generated using CityEngine (Procedural, inc., 2009)).[3]

6. Exemples de modélisation procédurale des mondes virtuels

6.1 Génération de terrain

De nombreuses solutions sont proposées pour la modélisation de terrains, des méthodes entièrement procédurales ou simulations physiques en passant par celles combinant exemple ou synthèse de texture avec une interaction à base d'esquisse.

6.1.1 Bruit : De nombreuses méthodes de modélisation de terrain procédurales sont basées sur le fait que les terrains sont auto-similaires, c'est-à-dire statistiquement invariant sous grossissement.

L'approche de la génération de l'énergie consiste en une édition pseudo-aléatoire des valeurs de hauteur sur un terrain plat à l'aide de bruit fractal ou de bruit Perlin[4] . En effet, en combinant plusieurs octaves (fréquences organisées en niveaux) de bruit, qui sont ensuite interpolées, il est possible de générer des cartes d'élévation détaillées et plausibles, en utilisant la valeur de bruit mise à l'échelle comme élévation du terrain à un emplacement donné. Ces méthodes sont un choix populaire pour la modélisation de paysages en raison de leur implémentation facile et de leur calcule. Pour plus d'informations sur les méthodes de génération de terrain fractal, voir le livre de Ebert et al. [5].. Les approches basées sur les fractales peuvent générer une vaste gamme de terrains de grande taille avec un niveau de détails illimité. Cependant, ils sont limités par le manque de contrôle de l'utilisateur, la manipulation de paramètres non intuitifs et l'absence d'effets d'érosion tels que les modèles de drainage.

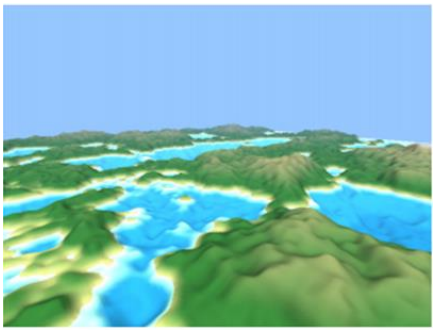

Figure 1.3: terrain généré à partir d’un bruit de perlin

6.1.2 Subdivision : Les premiers travaux sur la génération de terrain reposent également sur des méthodes de subdivision, telles que la méthode de déplacement en point médian. Dans cet algorithme de subdivision, chaque nouveau point est déplacé de façon aléatoire en hauteur par rapport aux altitudes moyenne et relative de son voisinage. Ce processus est appliqué de manière itérative avec une plage de jeu décroissante, jusqu'à ce que la résolution souhaitée soit atteinte. Cette technique a récemment été adaptée au GPU. Comme pour les méthodes basées sur le bruit, ces méthodes sont difficiles à contrôler et ne permettent pas de produire des terrains plausibles [6]

6.1.3.Érosion : Une autre classe d'algorithmes de modélisation de terrains est basée sur la simulation de phénomènes naturels, y compris la simulation de l'érosion. Ces techniques peuvent compléter les générateurs basés sur le bruit ou sur les subdivisions pour améliorer le réalisme des scènes générées. Musgrave et al. présente la première méthode d'érosions thermiques et hydrauliques basée sur les règles de la géomorphologie. La simulation de l'érosion est réalisée à l'aide d'automates cellulaires, dans lesquels le matériel cellulaire est progressivement dissous et déplacé vers les cellules voisines. La plupart des algorithmes suivants reposent sur des automates cellulaires similaires et se concentrent sur l'amélioration des règles de dissolution et de déplacement, tout en introduisant plusieurs nouveaux matériaux. Roudier et al. présentent une simulation d'érosion hydraulique qui utilise différents matériaux à différents endroits, ce qui entraîne des interactions différentes avec l'eau. Cette méthode a été enrichie par Beneš et Forsbach ,le domaine étant alors composé de plusieurs couches de matériaux. Nagashima combine des érosions thermiques et hydrauliques dans un réseau fluvial pré-généré avec une fonction fractale 2D. Chiba et al. génèrent un champ vectoriel de débit d'eau qui contrôle ensuite la manière dont les sédiments se déplacent pendant l'érosion.

Ce processus produit des structures de crêtes hiérarchiques et améliore ainsi le réalisme. Beneš et Arriaga effectuent l'érosion d'un matériau tendre sur un matériau dur pour modéliser des montagnes de la table avec des modèles d'érosion réalistes. Neidhold et al. présentent une simulation physiquement correcte basée sur la dynamique des fluides et des méthodes interactives permettant de saisir des paramètres globaux, tels que les précipitations ou les sources d'eau locales. Beneš et al. [7] présentent des simulations 3D volumétriques complètes de l'érosion hydraulique, leur permettant de créer des cascades, tandis que S'avua et al. sculptez de manière interactive des paysages complexes (figure 1.4). Kristof et proposent une érosion hydraulique rapide basée sur l'hydrodynamique des particules lisses (SPH), une méthode générale et répandue de simulation de fluides. Pytel et Mann présentent un système d'érosion hydraulique capable de simuler les avalanches, un phénomène important à prendre en compte pour modéliser des terrains avec des formes réalistes.

Le principal inconvénient de toutes ces méthodes est qu'elles ne permettent un contrôle indirect de l'utilisateur que par des essais et des erreurs, ce qui nécessite une bonne compréhension de la physique sous-jacente, du temps et des efforts nécessaires pour obtenir les résultats attendus. [8]

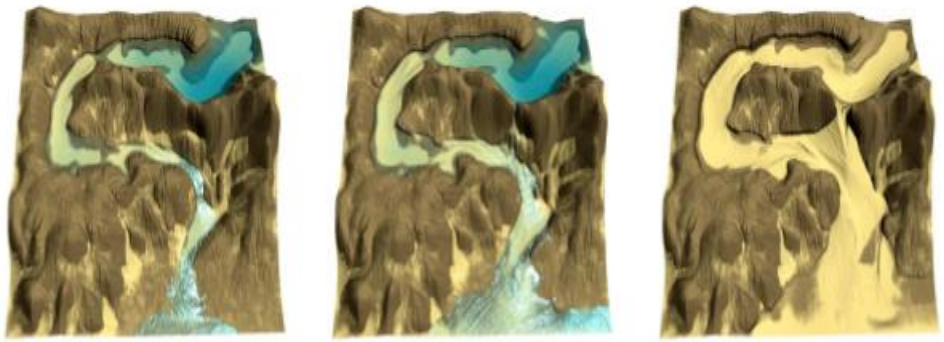

Figure 1.4: Erosion creusant progressivement un terrain lors d'une session de modélisation [9].

6.2 Plans d'eau

La génération procédurale de plans d'eau, tels que les rivières, les lacs, les ruisseaux, les océans et les cascades, a été moins étudiée que les terrains. Puisque nous nous intéressons aux débits d'eau conçus par les utilisateurs, les méthodes de simulation des eaux peu profondes utilisées pour calculer la trajectoire d'une rivière sur un terrain, sortent du cadre de cet examen.

6.2.1. Rivières : Le rendu des rivières est généralement effectué à l'aide de géométries planes et de textures animées. [10] Yu et al. dérive une texture animée du mouvement de particules simulées à la surface d'une rivière, qui donne l'apparence d'un fluide complexe. Ces textures peuvent être augmentées avec leur écoulement en contournant les pierres et les berges de la rivière Au lieu de cela, Van Hoesel dalle les textures et module leur application sur les polygones de surface de l'eau en fonction de la vitesse d'écoulement (Figure 1.5). [11]

Figure 1.5: Rendu en temps réel de rivières à l'aide de pavés de textures [12]. Gauche: la surface de la rivière est subdivisée en "tuiles". Chaque tuile a sa propre direction et vitesse de vague, qui sont interpolées en douceur pour produire l'animation de la surface de la rivière. Au milieu: rivière avec ondes directionnelles. Droite: surface animée du marais

6.2.2. Réseaux fluviaux : Plusieurs algorithmes ont été proposés pour générer des réseaux fluviaux, notamment pour les générer sur un terrain, voire pour générer des terrains. En effet, au lieu de générer le système hydraulique après la génération du terrain, des procédés ont été proposés pour générer le terrain en fonction de la génération du réseau hydraulique. Kelley et al. crée un terrain par la génération de bassins versants. Derzapf et al. génère des réseaux fluviaux à

l'échelle planétaire. Teoh présente Riverland, un algorithme qui génère un terrain complet basé sur la génération de réseau hydraulique. Généraux et al. [13] Crée de manière procédurale des terrains à partir d'un réseau de drainage des rivières cohérent (Figure 1.6). Un réseau de drainage hiérarchique se développe à partir de plusieurs endroits autour de la limites d'une île, en fonction des contraintes de terrain et de pente de la rivière fournies par l'utilisateur. En estimant plusieurs paramètres hydrauliques, ils déterminent les types et les formes de rivières et génèrent la géométrie du terrain à l'aide d'opérateurs de combinaison. L'utilisation de réseaux fluviaux dans le processus de génération améliore considérablement le réalisme des terrains générés. Cependant, ces méthodes sont limitées aux terrains façonnés par l'érosion hydraulique. [14]

Figure 1.6: Génération de terrain sur un réseau hydraulique cohérent [13]

6.2.3. Cascades : Deux méthodes principales sont spécifiquement appliquées pour créer et restituer des cascades animées: les systèmes de particules et les polygones texturés. Les systèmes de particules même optimisés avec des méthodes hiérarchiques ou des méthodes d'écran , dépassent leur complexité inhérente pour simuler efficacement des réseaux de chutes d'eau dans de grands environnements. Systèmes de particules préconfigurés pour des cascades spécifiques, assemblés pour générer un réseau de cascades. La cohérence entre le terrain et la cascade résultante doit toutefois être assurée par l'utilisateur. Les textures animées superposées sur des polygones constituent une approche beaucoup plus efficace et évolutive. Elles pourraient même être complétées par une forme plus légère de systèmes de particules . [15].

6.3.Plantes et écosystèmes

Certains des algorithmes procéduraux les plus courants sont basés sur des grammaires formelles, comme le LSystems spécialement conçu pour la génération d'arbres et de plantes. Un système L est une grammaire formelle qui utilise un ensemble de symboles et de règles de réécriture, appliqués en parallèle. [16]

Voici un exemple simple pour illustrer le principe de substitution de L-Systems:

Symboles: A, B

Règles: $A \rightarrow B$, $B \rightarrow AB$

Axiome: A

L'application itérative des règles de cette grammaire donne les résultats suivants:

A, B, AB, BAB, ABBAB, BABABBAB, ...

En considérant les arbres et les plantes comme des structures récursives il est possible de décrire les règles de croissance sous forme de grammaires pour générer des troncs, des branches et des feuilles qui composent les arbres (Figure 1.7). Cette méthode est également utilisée pour générer des plantes et des fleurs en croissance. L-Système ouvert[17]

Figure 1.7 Génération des arbres par L-System . [18]

est une extension de Měch et Prusinkiewicz qui prend en compte les contraintes externes lors de la génération. Ces contraintes permettent par exemple de simuler une compétition pour les ressources lors de la croissance et d'obtenir des résultats plus réalistes (figure 1.7, à droite). [18] Deussen et al. génère des écosystèmes complets à partir de ces grammaires, en définissant des règles permettant de générer des semences de plantes distribuées. Peyrat et al. proposent un procédé de génération de feuilles avec un processus de vieillissement codé dans un système L, simulant ainsi les changements de couleur, les trous et les fissures. Des modèles de procédure ont également été proposés pour modéliser d'autres types d'éléments de végétation. Par exemple, Desbenoit et al. générer du lichen par simulation. À l'aide d'un processus d'ensemencement à base de particules et de tests d'agrégation, le lichen se développe sur des objets 3D de manière plausible, en fonction des contraintes de l'environnement, telles que l'humidité et la forme. [19]

6.4. Environnements urbains

6.4.1 Réseaux de rue : La génération de villes procédurales est un domaine de recherche très actif car ces environnements sont parmi les plus complexes et les plus coûteux à produire à la main. Les méthodes existantes de modélisation des villes commencent par générer un réseau de rues. Les cycles formés par les rues voisines sont divisés en blocs servant d'empreintes de bâtiments. Inspiré de L-Systems le travail de pionnier de Parish et Müller pour la génération de réseaux urbains est entièrement automatique. Le réseau routier se développe de manière itérative sur le terrain, conformément aux règles de grammaire. Pour contrôler les axes de croissance, cette méthode prend en compte diverses contraintes telles que la présence d'eau ou la densité de population, les centres d'affaires, les zones résidentielles, etc.dans un environnement de grande ville.La figure 1.8 illustre les résultats de cette méthode et la puissance de tels algorithmes. Parallèlement, d'autres approches ont été introduites, telles que la génération de structure de ville basée sur des champs ou sur des exemples. Une autre approche simule la croissance des villes sur la base d'une simulation urbaine, ce qui donne des villes extrêmement complexes et réalistes. Notez que ces méthodes sont généralement dédiées aux villes de type américain, avec des réseaux semi-réguliers de rues principalement parallèles et perpendiculaires. [20]

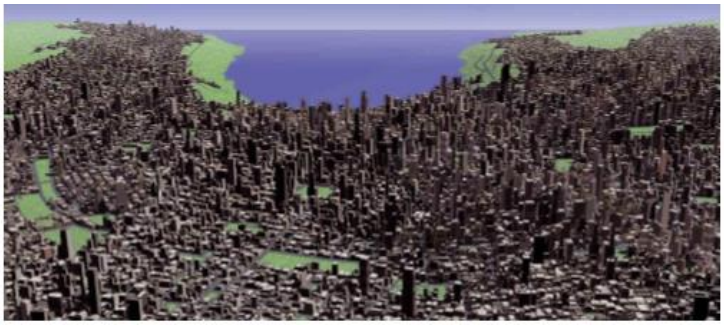

Figure 1.8: Utiliser L-Systems pour générer des villes . [20].

6.4.2.Établissements non urbains :Les méthodes précédentes pour générer des villes définissent le réseau de rues puis créent des parcelles de construction dans un deuxième temps. Par conséquent, elles ne peuvent pas être appliquées aux établissements dispersés, où la modélisation de l'interaction entre l'établissement progressif et l'extension du réseau routier est obligatoire. À notre connaissance, le seul ouvrage sur la génération de personnes non urbaines

6.4.3.Réseaux routiers : Les réseaux routiers sont également des éléments importants sur les terrains en dehors des villes. Sun et al. présentent une méthode pour créer des réseaux routiers à l'aide de règles de croissance et de gabarits de route fondés sur les tendances observées dans les réseaux routiers réels. Galin et al. présente une technique de génération de routes sur des terrains arbitraires, basée sur un algorithme de détermination de trajectoire qui optimise la trajectoire de chaque route en fonction de divers coûts, tels que les pentes du terrain et les virages. Il permet de prendre en compte l'environnement tel que la présence de végétation ou d'eau. Cette méthode permet de créer des routes réalistes, notamment des autoroutes et des routes de montagne. Les auteurs étendent ensuite leurs travaux pour générer des réseaux routiers complets reliant différentes villes et optimisant leurs connexions . [21]

6.4.4Immeubles : Lorsqu'un tracé de rue définit des empreintes de pas pour des bâtiments, une approche populaire de la génération procédurale de bâtiments utilise des grammaires. Parish et Müller ont utilisé L-Systems pour créer des bâtiments et leurs résultats ont démontré la puissance de telles applications. L'algorithme construit progressivement la géométrie du bâtiment en suivant les règles fournies, telles que l'extrusion de blocs de construction à partir de l'empreinte du bâtiment et la segmentation au sol à partir du bloc de construction nouvellement généré.

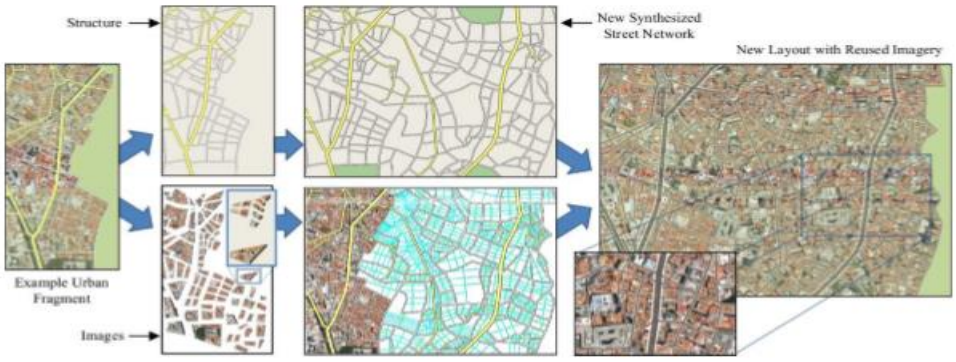

Figure 1.9: Synthèse interactive de la mise en page urbaine basée sur des exemples [22].

Wonka et al. présentent des grammaires divisées, un type grammatical particulièrement efficace pour la génération de façades de bâtiments. Les grammaires de forme ont ensuite été introduites

par Müller et al. , où les symboles des règles sont des formes géométriques qui sont remplacées et raffinées au cours du processus de génération. Ces grammaires sont jusqu'à présent la méthode la plus courante pour la génération d'extérieurs de bâtiments. Plutôt que d'utiliser des grammaires, Leblanc et al. présentent une approche basée sur des composants permettant de générer de manière procédurale des bâtiments entiers, avec des formes internes et externes cohérentes. Un composant est défini par une forme 2D ou 3D et un ensemble d'attributs. Les bâtiments sont générés de manière procédurale en utilisant une série d'énoncés qui modifient progressivement les composants existants ou en créent de nouveaux [23].

7. La force des méthodes procédurales

La force des méthodes procédurales réside dans leur capacité à générer une grande quantité de contenu avec seulement quelques règles, dont les paramètres sont définis par l'utilisateur. Ces règles permettent l'émergence de formes complexes qui incitent les artistes à s'efforcer, par exemple, de créer des algorithmes d'érosion offrant des schémas d'érosion réalistes sur des terrains difficiles à réaliser à la main. Cependant, les approches procédurales ne fournissent des contrôles indirects que via l'édition des règles. Le processus de modélisation est donc généralement une succession d'essais et d'erreurs jusqu'à l'obtention d'un résultat satisfaisant. De plus, il est souvent nécessaire de bien comprendre le modèle et les rôles des paramètres pour obtenir le résultat souhaité, en limitant l'utilisation de ces outils à des spécialistes. Enfin, ces méthodes se sont révélées puissantes dans leurs contextes respectifs, mais elles sont souvent fastidieuses à contrôler, même pour des utilisateurs expérimentés.

L'augmentation de la puissance de calcul combinée à des méthodes de visualisation de plus en plus performantes ont permis de créer des terrains réalistes.

8. Les avantage et les inconvénients de la modélisation procédurale

8.1 Avantage :

- La modélisation procédurale est la solution alternative d'excellence à l'utilisation excédentaire de ressources en temps humain humain d'infographiste 3D mais aussi en ressources matérielles
- Bien que toutes les techniques de modélisation sur un ordinateur nécessitent des algorithmes pour gérer et stocker des données à un moment, la modélisation procédurale met l'accent sur la création d'un modèle à partir d'un ensemble de règles, plutôt que de modifier le modèle via l'action de l'utilisateur. La modélisation procédurale est souvent appliquée quand la création d'un modèle 3D avec modeleurs 3D génériques est trop lourd (en temps et en coût de production), ou lorsque des outils plus spécialisés sont nécessaires. Ceci est souvent le cas pour les plantes, l'architecture ou les paysages.
- L'un des principaux avantages des particules, et probablement la principale raison de son grand attrait, réside dans ses capacités d'amplification des données .En bref: un simple jeu de paramètres d'entrée ou quelques règles de génération du modèle procédural génèrent une grande variété de modèles. [24].
- Une autre propriété essentielle de PM est la compression des données: un modèle géométrique plutôt complexe peut être représenté par un modèle procédural compact et un ensemble de paramètres, tandis que la géométrie réelle est générée uniquement lorsque cela est nécessaire. Cet avantage des particules devient encore plus pertinent en raison de l'évolution du

matériel informatique, qui permet de transférer davantage de fonctionnalités vers les unités de tessellation et de shaometry de géométrie du GPU. Au lieu de stocker les données, on peut simplement exécuter des instructions pour les générer.

- PM peut créer une variété de résultats à partir d'un ensemble de paramètres d'entrée. par exemple. un grand nombre de modèles d'arbres, tous de la même espèce et du même âge, mais ayant chacun une structure et une forme uniques.
- Les avantages de PM rendent cet environnement particulièrement attrayant pour la création d'environnements virtuels. Les mondes virtuels sont importants pour de nombreuses applications, y compris la diversité considérable des simulations et des jeux (sérieux). Cependant, les méthodes actuelles de création manuelle de mondes virtuels sont trop coûteuses en main-d'œuvre, car le niveau et le niveau de détail requis de leur
- contenu en 3D augmentent continuellement. Il est de plus en plus reconnu que le fait de générer de manière créative des techniques de création peut résoudre ce problème.

8.2 Inconvénient :

- Cependant, malgré les promesses d'un gain de productivité élevé, d'une représentation compacte et d'une variation finale apparemment interminable du contenu, la plupart des méthodes de gestion de projet actuelles n'offrent toujours pas d'alternative appropriée à la modélisation manuelle.

9. Comparaison entre les méthodes génération procédural

En comparant (Table 1.2), les différents types de génération procédurale évolutionnaire étudiés comme les automates cellulaires , les grammaires de formes et les L-Systèmes , nous avons choisi de nous intéresser aux L-Systèmes. En tant que grammaires formelles elles décrivent une topologie et non une géométrie avec une croissance par insertion. Cela induit que l'espace se construit au fur et à mesure de l'évolution ce qui le rend plus facile manipuler quand on ne sait pas à quel type de forme s'attendre. Méthode Croissance Représentation Automate cellulaire Par agrégation Topologique Grammaire de formes Par insertion Géométrique L-Système Par insertion Topologique Table 3 Comparaison des méthodes de génération procédurale [25].

Méthode	Croissance	Représentation
Automate cellulaire	Par agrégation	Topologique
Grammaire de formes	Par insertion	Géométrique
L-Système	Par insertion	Topologique

Tableau 1.2 ; comparaison entre les méthodes génération procédural [25]

10. Conclusion

Nous avons étudié les méthodes de modélisation procédurale visant à générer une grande variété de fonctionnalités pour les mondes virtuels 3D. Nous avons observé que la modélisation procédurale est un domaine de recherche de plus en plus actif qui a généré de nombreuses applications de haute qualité.

Chapitre 2 : Modélisation procédurale de terrain

1. Introduction :

un monde virtuel comprend de nombreux composants dans ce chapitre nous allons nous présenter des techniques de générations de terrain qu'il joue un rôle clé pour la modélisation des mondes virtuels et la création d'une scène naturelle. Les techniques de génération automatique ont de nombreuses applications comme les simulateurs de vol, les effets spéciaux au cinéma ou les jeux vidéo.

2. Catégorisation Les méthodes de génération de terrain :

On peut divisées Les méthodes de génération de terrain en trois catégories principales

2.1 Modélisation basée sur la physique

Ces méthodes simulent des processus du monde réel en appliquant des lois physiques sur des objets virtuels

2.2 Algorithmes d'approximation

Ces méthodes ne simulent aucune loi physique mais essaient seulement d'approximer le résultat final des processus de mots réels

2.3 les méthodes de synthèse à partir d'exemples

Ces méthodes ne permettent de reproduire qu'un seul type de relief homogène aux images données. Où d'algorithme permettant une modification interactive de terrains complexes avec un haut niveau de détails au sol, et permettant un placement rapide et intuitif d'éléments caractéristiques géologiques et géographiques en utilisant des opérations d'édition simples

2.4 Bilan

Tableau 3 présente une bilan de catégorie de technique de génération de terrain

	Avantages	Inconvénients
Modélisation basée sur la physique	conduit à des résultats très réalistes, qu'ils soient statiques ou dynamiques.	ces méthodes sont très exigeantes en termes de performance et impliquent généralement beaucoup d'itérations avec des milliers à des millions de petites particules. Pour cette raison, utilisés uniquement dans la recherche et les applications scientifiques.
Algorithmes d'approximation	Ils visent généralement une approximation visuellement suffisante tout en gardant le coût de calcul final aussi bas que possible. La base théorique de ces méthodes réside habituellement dans la théorie fractale ou la synthèse du bruit.	La solution ni pas exacte
les méthodes de synthèse à partir d'exemples	les terrains sont modélisés sur des structures de grilles qui représentent des reliefs à une précision fixe. Ces terrains sont souvent stockés sous la forme de cartes de hauteurs qui peuvent être converties, plus tard, en des maillages adaptés à une visualisation rapide avec des mécanismes de niveaux de détails.	Ne reproduisez qu'un type de dilution homogène des images rendues.

Tableau 2.1 : Catégorisation et prétérite Les méthodes de génération de terrain

3. Techniques Génération de terrains

Il y a trois techniques de génération de terrains existent : les modèles fractals (modèles à synthèse procédurale), les méthodes de simulation d'érosion et les algorithmes de synthèse à partir d'exemples.

3.1 Les algorithmes de synthèse à partir d'exemples :

consiste à générer un terrain automatiquement par des constructions géométriques ou des combinaisons de fonctions de bruit . Ces méthodes produisent de très grands terrains qui manquent parfois de réalisme. Plusieurs problèmes ressortent de ces méthodes. [26]

L'utilisation de méthodes automatiques ne permet pas de contrôler la forme finale du terrain. L'utilisation de cartes d'élévation de données limite fortement la topologie et la géométrie du terrain.

Gamito et Musgrave améliorent les cartes d'élévation à l'aide de courbes de profils pour générer procéduralement des terrains avec des surplombs. Leur approche ne permet cependant pas la création de toutes les formes volumiques comme les grottes ou les arches. [27]

3.2. La méthode de simulation d'érosion :

Les méthodes de simulation d'érosion permettent de générer des terrains physiquement plausibles . Ces méthodes s'appuient sur des simulations plus ou moins précises pour déterminer le transport et le dépôt de matière sur des cartes d'élévation de données. Plusieurs méthodes ont été portées sur GPU pour accélérer les temps de calcul couteux de la simulation. Pour simuler l'érosion sur des terrains tridimensionnels, Ito et Beardallutilisent des grilles de voxels mais la taille du terrain reste limitée sur à cause du coût de la structure de données. L'utilisation d'au plus deux matériaux (la roche et les sédiments) pour simuler l'érosion est une autre limitation de ces approches. Pour contrôler la génération du terrain, Zhou a mis en place une procédure s'appuyant sur la synthèse de textures. Cette méthode permet de contrôler globalement la forme finale du terrain, mais s'appuie également sur une représentation à base de cartes d'élévation de données.[28]

Ces méthodes ont deux principales limitations. Elles utilisent essentiellement des cartes d'élévation de données limitant la topologie et la géométrie du terrain en ne permettant pas de modéliser des surplombs, des falaises, des arches ou des grottes. D'autre part, le terrain est souvent considéré comme un unique matériau homogène ce qui diminue le réalisme et les possibilités de modélisation.

Pour la dernier modelé on s'intéresse aux **modèles fractals ou (modèles à synthèse procédurale)** qui permet la génération automatique des terrains, parce qu'il ne nécessite pas un grand espace de stockage et ses résultats plus réalistes .

3.3. les modèles fractals (modèles à synthèse procédurale)

3.3.1 Principe et théorie: Une approche intuitive consiste à considérer qu'une fractale [29] est la transformation récursive d'un objet par n copies de lui-même a l'échelle r (avec r < 1).

- Cette propriété est appelé l'autosimilarité.
- On à coutume de caractériser un objet fractal à partir d'un paramètre numérique généralement noté D appelé dimension fractale.

➢ D est égal à (D = $\frac{\log(n)}{\log(\frac{1}{r})}$) Il caractérise la manière selon laquelle une fractale évolue dans l'espace ou elle est dessinée. Plus D est grand plus l'évolution est "chaotique".

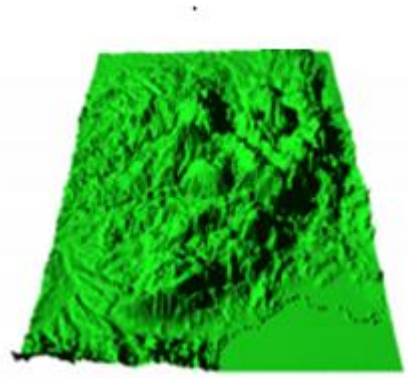

Figure 2.1 surface original

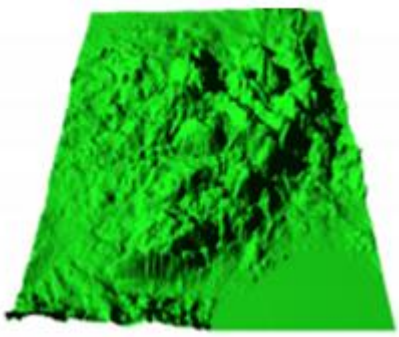

Figure 2.2 surface fractale approximante

3.3.2. Types des fractales

Contient deux types des fractales :

3.3.2.1 Fractales déterministes: On appelle fractale déterministe une fractale dont le mode de réplication ne fait pas intervenir de composante aléatoire.

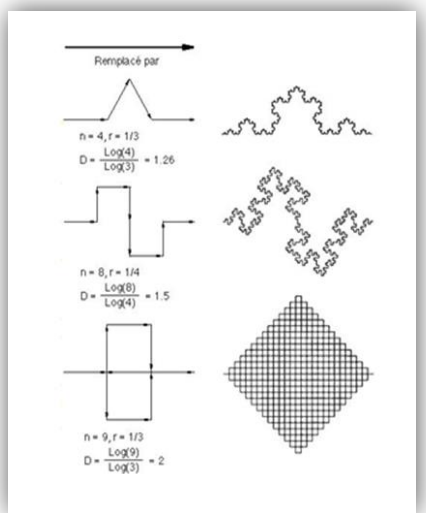

Figure 2.3 - Fractales déterministes

3.3.2.2.Fractales non déterministes: Les fractales stochastiques, par opposition aux fractales déterministes, permettent de générer des dessins non réguliers. Pour générer une fractale stochastique, on n'a plus un seul mode de réplication, mais deux ou plusieurs choisis aléatoirement. On obtient de bons résultats avec D voisin de 1,2 pour générer des rivages océaniques (lignes) et avec D voisin de 2,2 pour la génération de montagnes (surfaces).

Plus D croît plus la ligne ou la surface devient chaotique et irréaliste.

Historiquement les fractales sont issues des travaux de Mandelbrot et Van Ness sur le "mouvement Brownien fractionnaire" (fractional Brownian motion, fBm).

Il s'agit de caractérisation une famille de processus stochastiques Gaussiens unidimensionnels VH(t) (H : réel compris entre 0 et 1, t : le temps).

Plus H est proche de 0 plus la courbe est bruitée. Une valeur de H égale à 0,5 quantifie un mouvement Brownien normal.

Auto affinité Le principe de l'auto affinité correspond aux faits suivants :

Deux morceaux quelconques de la courbe représentative se ressembleront,

Si on effectue une mise à l'échelle d'un facteur r sur le temps et d'un facteur rH sur VH(t), la courbe garde toujours le même aspect.

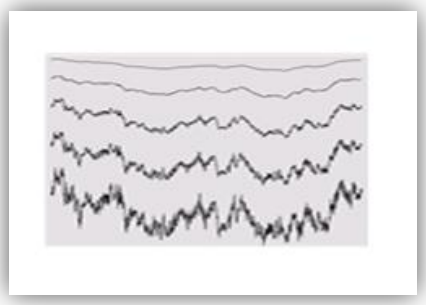

Figure 2.4 - FBm avec des valeurs de H égales à 0.8, 0.6, 0.4, 0.3 et 0.2.

L'extension multidimensionnelle des fBm permet de modéliser un large éventail de phénomènes naturels.

Pour les reliefs, le paramètre temps t est transforme en un couple (x,y) indiquant une position dans le plan, VH(x,y) donne l'altitude du point P.

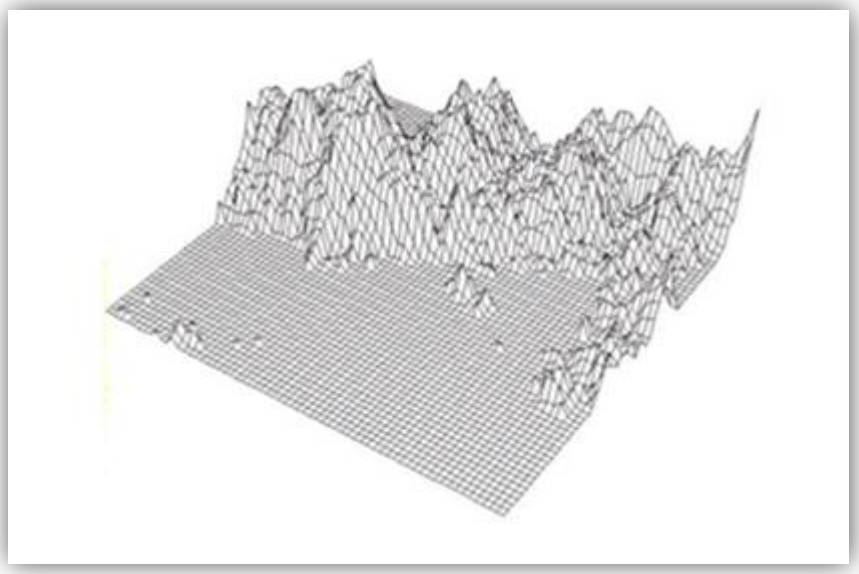

Figure 2.5 - Un relief fractal généré par fBm représenté par facettes

Un objet fractal de dimension fractale D apparaît comme un fBm de coefficient H tel que (D = E + 1 – H) où E est le nombre de variables de la fonction fBm, c'est à dire la dimension euclidienne de l'objet construit. Cet objet n'est évidemment pas strictement auto similaire mais seulement auto affine.

Implantation des fractales stochastiques La programmation exacte des lois mathématiques requiert une quantité de calcul très importante. Les implantations sont essentiellement des approximations du mouvement Brownien fractionnaire Soit par des techniques spatiales (subdivision récursive), Soit par des techniques spectrales (par filtrage de Fourier).

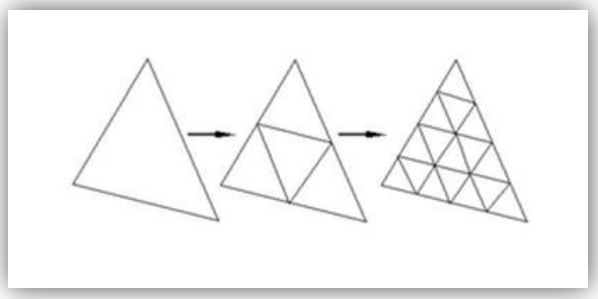

Figure 2.6 - Fractalisation d'un triangle

D'autres outils aident Fractal à modéliser des terrains

3.3.3 les cartes hauteurs et les terrains

Une carte d'élévation représente le terrain comme une grille régulière à deux dimensions . La carte est dénie comme une fonction de hauteur f qui associe à chaque noeud (x, y) de la grille une altitude f(x, y).Cette représentation permet de connaître en tout point de l'espace la hauteur du terrain. Cette représentation est la plus utilisée en informatique graphique car elle est simple et compacte.

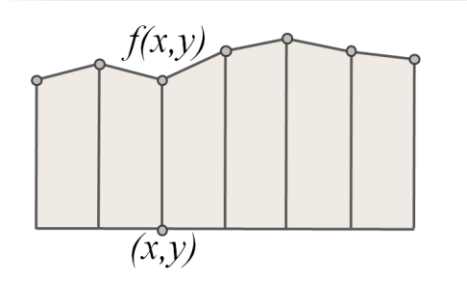

Figure 2.7 - la carte hauteurs

De plus, cette structure peut être représentée sous la forme d'une image où la couleur définie l'altitude du terrain, ce qui permet d'optimiser certaines opérations en utilisant les capacités des cartes graphiques actuelles [30].

Les opérations de modification du terrain sont également simplifiées. La génération de la carte consiste uniquement à créer une fonction de hauteur en tout point de l'espace. Une limitation de ce modèle est de ne pas représenter des terrains avec des caractéristiques géologiques complexes comme les arches et les grottes.

La représentation la plus utilisée pour modéliser un terrain est la carte d'élévation . Avec cette représentation, un terrain est stocké sous forme d'un tableau à deux dimensions, ou chaque case contient la hauteur du terrain. De plus on peut associer une couleur à chaque case, ce qui revient à associer une texture au terrain. La souplesse de cette représentation provient de la possibilité de considérer ces cartes comme des images 2D dont la manipulation est très intuitive :

- modélisation possible avec des outils de dessin 2D,
- facilite de génération de niveaux de détails, ...etc.

4 .Quelques algorithmes de génération de terrain

4.1 L'algorithme de subdivision en triangles

Principe : L'algorithme de subdivision en triangles prend un terrain triangle, et dans chaque itération on subdivise chaque triangle en quatre triangles plus petit, puis on applique un facteur de perturbation a chaque nouveau sommet crée. [31]

1. Commencer par un triangle. Choisir une altitude aratoire pour les points finaux.

2. Pour chaque arête de triangle. On détermine le point milieu et on applique une perturbation aléatoire. La perturbation est distribué entre -kr et kr, (k est la longueur de l'arrête subdivisée et r est un paramètre de rudesse).

3. Diviser le triangle en quatre plus petit triangles par la connexion des points milieu, répéter le processus pour chaque petit triangle

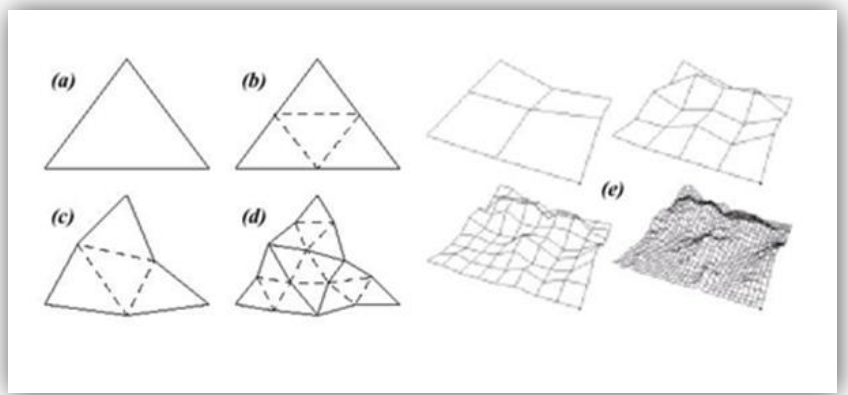

Figure 2.8 - Technique de subdivision

On répète le processus jusqu'au niveau de détail désiré. A chaque itération la perturbation aléatoire se réduit car les arêtes serons plus petites.

Cet algorithme possédé un paramètre rudesse qui contrôle le terrain résultat. avec un grand r la perturbation décroître plus rapidement après chaque itération qu'avec un petit r.

Alors avec un grand r, la perturbation devient petite dans le haut niveau de détail. Le résultat

L'algorithme peut être appliqué à une grille carrée par division au diagonal en deux triangles.

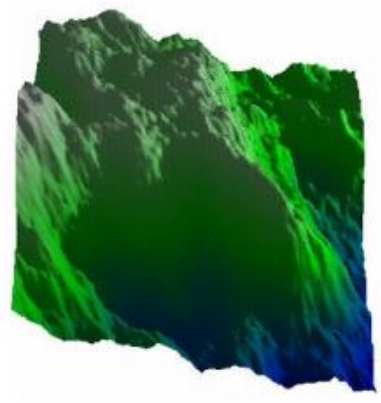

Figure 2.9 - Subdivision en triangles r = 1.0

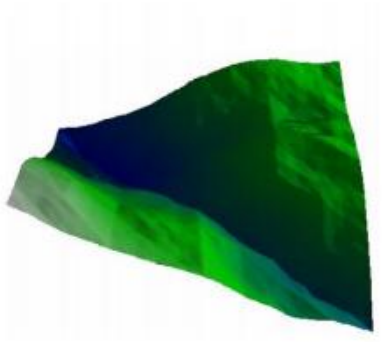

Figure 2.10 - Subdivision en triangles r = 1.5

Un défaut clair de cet algorithme est l'apparence des rainures entre les arêtes des triangles. Ceci est plus clair quand le terrain est plus lisse, comme dans la figure précédente il y a une rainure qui forme une diagonale principale. Ce résultat est du a la géométrie de l'algorithme. La raison pour la quelle cette rainure apparaît est que l'altitude de chaque point se détermine en fonction de deux points seulement et les autres points ne sont pas pris en considération [31].

Avantage: Le principal avantage de cette méthode est le fait que l'on est certain du paysage que l'on va représenter, étant donne qu'il est fige sur fichier. On peut aussi le retoucher afin de l'améliorer ou alors, on a la possibilité d'ajouter facilement des décors selon nos envies [32].

Inconvénient: Le principal inconvénient est que le terrain généré consomme beaucoup d'es pace mémoire pour être stocke, un terrain d'une taille de 1024 x 1024, avec les altitudes stockes sur 16 bits, prend effectivement 2 Méga octets, sans compter les informations complémentaires que l'on désire y ajouter [32].

Figure 2.11 Triangular interloper

Figure 2.12 Triangular Fractal Norma

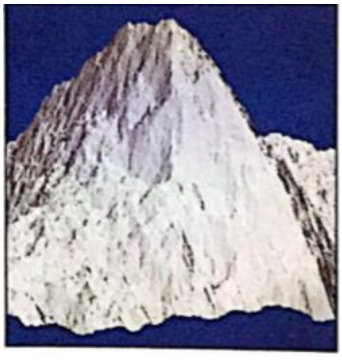

Figure 2.13 Triangular Fractal Mountain

4.2 L'algorithme Diamond-Square :

L'algorithme Diamond-Square [31], comme son nom l'indique, se base sur 2 phases : la phase du carre et la phase du diamant. Il s'agit d'itérer ces 2 phases jusqu'a calculer tous les points de la matrice représentant le terrain 3D.

En effet, la largeur de la matrice représente les abscisses, la longueur correspond aux ordonnées et enfin chaque "case" de la matrice représente la hauteur relative du point dont on vient de décrire les 3 composantes pour le situer dans un repère : pour accéder a une case de la matrice on fait tab[x] [y] = z; ce qui indique que les 3 coordonnes sont présentes.

Pour mieux comprendre le fonctionnement, prenons un exemple simple pour expliquer l'algorithme. Soit une matrice 5X5 :

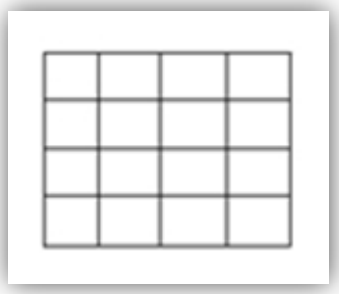

Figure 2.14- Matrice initiale

La phase du carré consiste à élever le centre du carré d'une valeur aléatoire (le point en rouge):

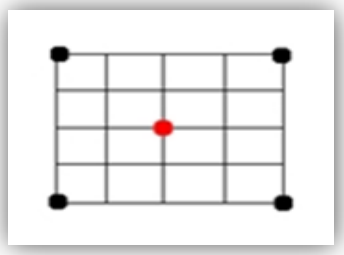

Figure 1.15- La matrice après la phase du carré

La phase du diamant consiste à élever le centre de chaque losange formé par le centre et les coins du carre précédent d'une valeur aléatoire (les points en bleu). On peut constater déjà qu'on se retrouve confronte a un problème : il n'y a pas quatre points complet pour former le diamant. Il faut donc tenir compte du cas particulier des bords :

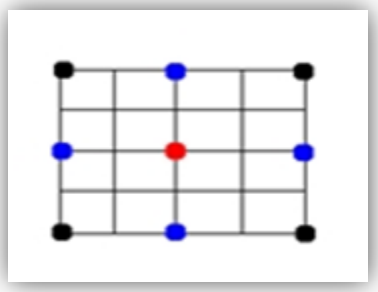

FIG. 2.16 - La matrice après la phase du diamant

On réitère avec les carrés obtenus (vert), puis les diamants (orange) et ainsi de suite jusqu'à parcourir l'ensemble de la matrice :

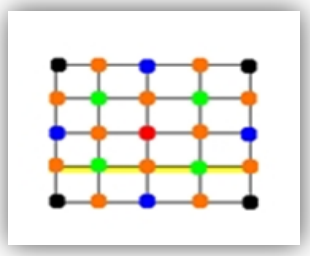

Figure 2.17 - La matrice finale

On obtient donc l'algorithme suivant :

— espace = Largeur ;
— tant que espace > 1 faire
— {
— espace = espace/2;
— pour chaque carre de largeur espace faire
— {
— étape du carre;
— }
— pour chaque diamant de largeur espace faire
—{
— étape du diamant;
—}
—}

La phase du carre calcule la moyenne des 4 points formant le carre. Lors de cette phase, on est positionne sur le centre ayant pour coordonnes (x,y). Sur un carre (a, b, c, d), on retrouve les coordonnes suivantes :

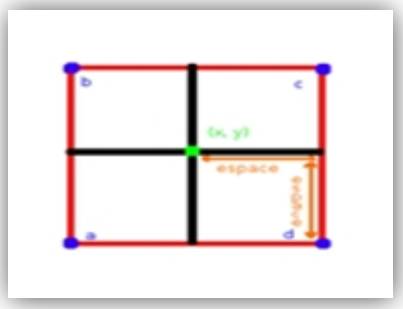

Figure 2.18- Le carré

a = (x - espace, y - espace)

b = (x - espace, y + espace)

c = (x + espace, y + espace)

d = (x + espace, y - espace)

Mais il faut faire attention aux limites de la matrice. Il faut tester les coins du carré pour que leurs coordonnes soient comprises entre 0 et (largeur Matrice - 1).

La phase du diamant calcule la moyenne des 4 points formant le losange (représentant le diamant). on est positionne sur le centre qui a pour coordonnes : (x, y). Sur un diamant (a, b, c, d).

On retrouve les coordonnées suivantes :

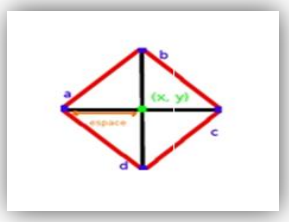

Figure 2.19 Le losange

a = (x - espace, y)

b = (x, y + espace)

c = (x + espace, y)

d = (x, y - espace)

Il faut encore faire attention aux limites de la matrice !

Comme le terrain est aléatoire, la hauteur d'un point doit être choisit au hasard. Néanmoins, pour ne pas avoir n'importe quoi, il faut quand même contrôler sa valeur. On parle alors de hauteur pseudo aléatoire car cela n'est pas totalement du au hasard.

Tout d'abord, la hauteur d'un point est calculée en fonction de la moyenne de la hauteur des points que l'algorithme a utilise pour l'obtenir.

On y ajoute une valeur aléatoire, a laquelle on applique un coefficient de lissage pour ne pas avoir un terrain en dents de scie. Ensuite, on multiplie a cette valeur aléatoire, l'espace entre le point cible et les points sources, afin de garder la cohérence du terrain. Voici donc la formule pour obtenir la hauteur :

hauteur = moyenne () + random () * espace * lissage

Le coefficient de lissage est fractionnaire et donc inferieur a 1. Plus la valeur est élevée, moins le terrain sera lisse :

Figure 2.20- Lissage à 0.1

Figure 2.21 - Lissage à 0.5

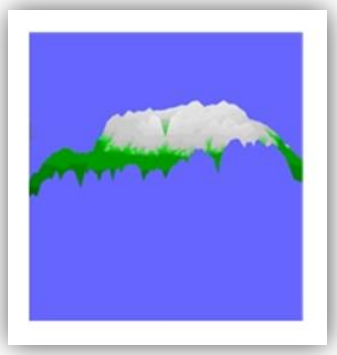

Figure 2.22- Lissage à 0.9

Les figures suivantes représentent deux terrains générés par l'algorithme du Diamond-Square:

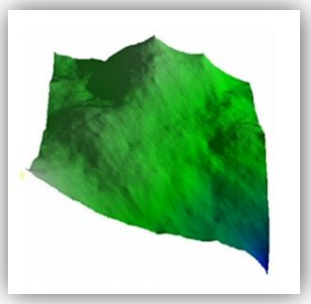

Figure 2.23 – Diamond-Square lisse

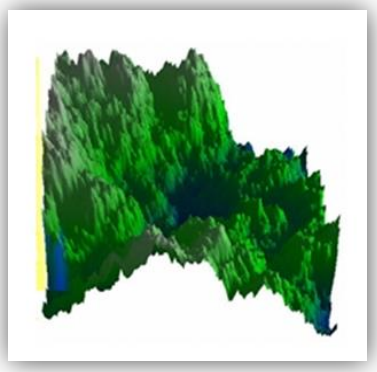

Figure 2.24 – Diamond-Square rude

Le défaut posé par l'algorithme décomposition en triangles ; l'apparence des rainures entre les arêtes des triangles, a été règle ici par la dépendance de chaque point des autres points dans les quatre directions, mais pour un terrain lisse on aura le problème d'apparition des petites crêtes (sommets) [31]. Cet algorithme est mieux que le précédent mais le terrain obtenu n'est pas vraiment aléatoire.

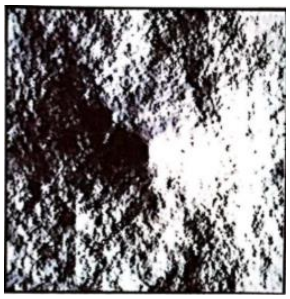

Figure 2.25 Diamond-Square Interpolant

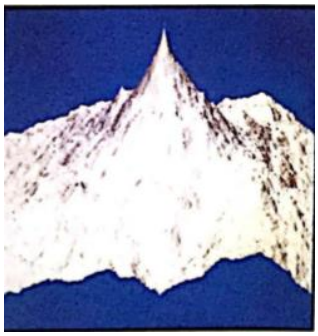

Figure 2.26 Diamond-Square Fractal Normal Map.

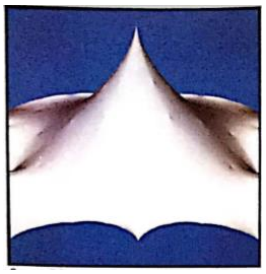

Figure 2.27 Diamond-Square Fractal Mountain

4.3 L'algorithme de Perlin

4.3.1.Principe de l'algorithme

Pour générer un terrain aléatoire, la première idée (naïve) qui vient à l'esprit est de générer une hauteur aléatoire pour chaque position du terrain. Ceci donne des résultats très inexploitables. En effet, le terrain généré n'a aucune cohérence, il ressemble a tout sauf a un terrain.

En donnant une valeur entièrement aléatoire à chaque position du terrain, on obtient le résultat suivant :

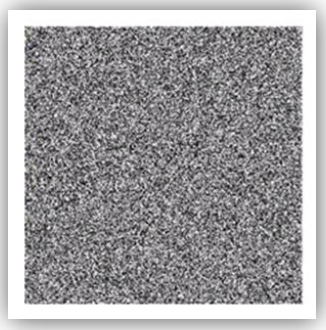

Figure 2.28- Terrain purement aléatoire

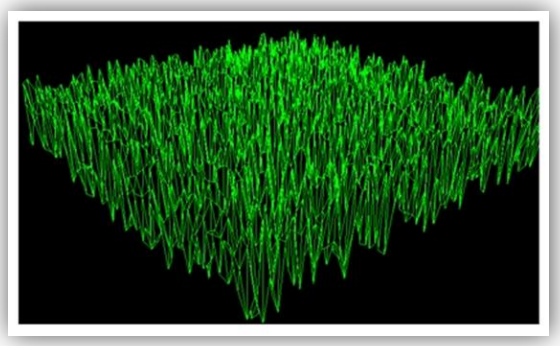

Figure 2.29 - Même terrain dans le viewer de terrain de Fearyourself

L'idée de l'algorithme de Perlin est d'affiner un terrain par itérations successives tout en interpolant entre les valeurs connues. La génération commence en fixant quelques valeurs puis en interpolant tous les autres points a partir de ces valeurs. A ce stade la, le résultat commence a ressembler a un terrain, mais il n'est pas naturel. Il va falloir ensuite lui ajouter des "calques" successifs pour affiner la pente du terrain et lui donner un aspect réaliste .

La structure de données utilisée est celle d'un "calque". Un calque va représenter une image en niveau de gris, c'est sur un calque qu'on fera toutes les interpolations. Chaque calque sera aussi muni d'un attribut note persistance qui définira un niveau sur les données de calque [33].

On va commencer par générer un calque entièrement aléatoire, sans aucune cohérence, ce calque nous servira à stocker toutes les valeurs aléatoires.

On va ensuite choisir certains points régulièrement espacés dans le calque pour prendre les valeurs aléatoires associées. Toutes les positions entre ces points choisis seront interpolées

(linéairement ou autrement). On voit déjà apparaître l'un des paramètres de l'algorithme : le caractère régulièrement espacé des points choisis.

Ce paramètre est noté la fréquence. Choisir une fréquence de 2 signifie qu'on prend 9 points sur notre calque aléatoire pour commencer les interpolations. En effet, on coupe un segment en 2 parties, on a donc 3 points (le premier, celui du milieu et le dernier). Et comme on se place en deux dimensions, on a 3 × 3 valeurs à prendre. Une fois que tout le calque aura été interpolé entre ces (fréquence + 1)² valeurs, on va créer un autre calque avec d'autres interpolations, plus fines.

Chaque nouveau calque sera créé exactement de la même manière que le premier, à la différence près qu'on prend une fréquence située un octave au dessus de la fréquence associée au calque précédent. Il faut donc connaître la fréquence de chacun des calques et la fréquence de départ. La fréquence d'un calque étant la fréquence du calque précédent × la fréquence de départ.

Ainsi, avec une fréquence de départ de 2, le premier calque aura une fréquence de 2, le second aura 4, le troisième aura 8, puis 16, 32 . . .et ainsi de suite. On voit ici apparaître un autre paramètre de notre algorithme : le nombre de calques à utiliser. Ce paramètre est en fait un nombre d'octaves à utiliser.

Notons qu'il n'est pas nécessaire que toutes les valeurs choisis aléatoirement sur un calque le soient aussi sur le suivant. L'essentiel étant que les calques correspondent à des interpolations de plus en plus fines, même s'ils ne correspondent pas aux même valeurs aléatoires choisies.

Et la partie astucieuse de l'algorithme consiste à additionner tous ces calques de manière pondérée, de sorte à ce que les calques interpolés finement influent moins que ceux interpolés grossièrement. C'est justement cette pondération qui permet de générer des montagnes, des vallées . . .

On voit apparaître le dernier paramètre de notre algorithme : les pondérations des calques. C'est ce qu'on a appelé persistance dans la définition d'un calque.

Pour avoir un résultat correct, la persistance doit être comprise entre 0 et 1. Une persistance de 0.4 signifie que le premier calque sera pondéré de 40%, que le suivant sera pondéré de :

0.4 × 0.4 = 16%, le suivant de 0.4 × 0.4 × 0.4 = 6.4% . . .

On obtient donc ce résultat réaliste

Figure 2.30 – Résultat cohérent

Figure 2.31- Résultat lissé

4.4 Bilan

Le tableau comparatif suivant, représente un bilan des algorithmes pour la génération des terrains que nous avons élaborés :

Méthode	Caractéristiques	Avantages	inconvénients
L'algorithme de subdivision en triangles	– Basé sur une subdivision récursive du modèle. -L'introduction d'un facteur aléatoire.	- Méthode simple à implémenter - Possibilité de contrôle du résultat via le paramètre de rudesse.	- Apparence des rainures entre les arêtes des triangles. - Coûteuse en terme de calculs. - Prend de l'espace pour stocker les informations.
L'algorithme Diamond-Square	– Processus itératif. - La hauteur d'un point dépend des points voisins.	- Possibilité de contrôler le résultat via le coefficient de lisage. - Problème de rainures réglé	- L'apparence de petites crêtes (sommets) surtout pour les terrains lisses.
La modélisation des terrains par l'algorithme de Perlin	– Processus itératif par affinement. - Basé sur les interpolations.	- Le résultat est un terrain homogène et réaliste.	- Les interpolations sont très coûteuse en terme de calculs.

5. Génération de textures de terrain

Pour affiche la terrain après la modélisation utilise à le colorer ou à utiliser une texture pour rendu les triangle genre par la modélisation de terrain

Le problème réside dans la génération de cette texture. Cette partie montrera les idées générales sur la génération de textures, et servira a présenter la façon avec la quelle un objet (terrain) sera afficher en espérant obtenir un résultat suffisamment réaliste.[5]

5.1 Le mélange de textures de base :

- l'idée est de générer une texture qui sera faite parfaitement par rapport au terrain qui sera affiché.
- Utiliser trois textures différentes pour afficher correctement le terrain.

Voici les trois images qu 'on va utiliser :

Figure 2.32 Image de niveaux

Texture générée à partir des trois images de base

Figure 2.33images degré

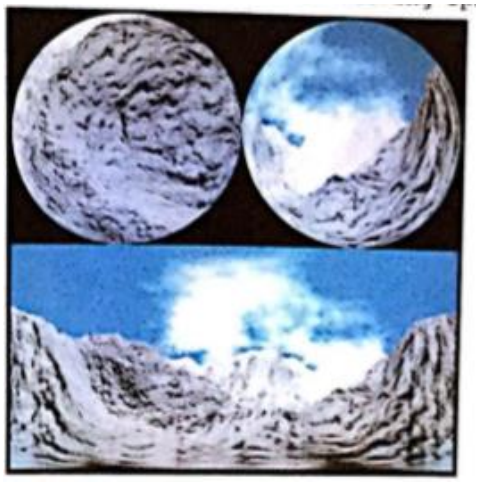

Figure 2.34: mélanger de texteur

❖ Un inconvénient à la méthode de mélange de textures de base [34] est le fait qu'on doit effectuer une passe par texture que l'on module. Il faut ainsi retracer le paysage autant de fois qu'il y a de textures de base différentes. Cela signifie que lors de la première passe on trace le paysage avec la texture qui représente l'herbe, et ensuite on retrace le paysage avec la texture qui représente les rochers. Heureusement, en pratique, effectuer cette opération n'est pas deux fois plus lente comme on pourrait le penser.

5.2 Calcul de la couleur du pixel :

* Pour avoir la couleur du pixel, on utilise directement la hauteur du pixel.

* Dépendant de la hauteur, on va utiliser un mélange entre les différentes images de niveaux[5]

Pour générer un terrain comme celui-ci, l'algorithme qu'on va utiliser est relativement simple :

Il faut donc savoir calculer la hauteur h du pixel associé et calculer la couleur du pixel.

5.3 Algorithme générer texture du terrain

→ Pour chaque pixel de la texture du terrain

→ Faire

→ Soit h la hauteur dans le terrain du pixel associé

→ Utiliser la valeur de h pour calculer la partie des trois textures de base

→ Affecter la couleur au pixel

→ Fin

Figure 2.35 Trois images de base

Figure 2.36L Texture générée à partir des trois images de base

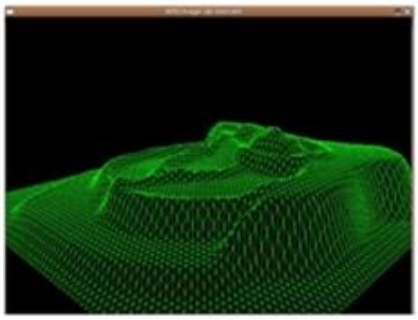

Figure 2.37- Affichage de terrain 3D en filaire

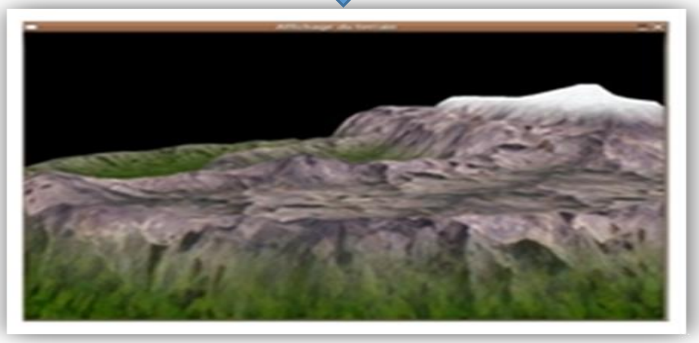

Figure 2.38 - Terrain texturé

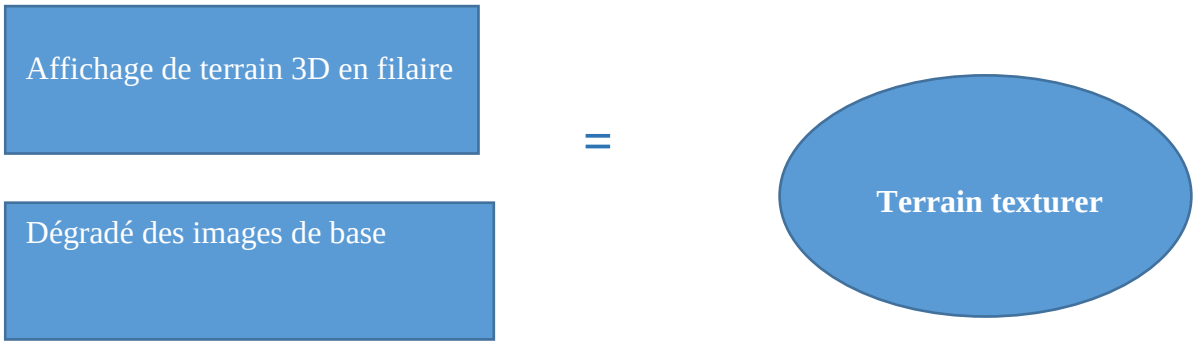

Figure 2.39 – structure générale de texteur

5. Conclusion

L'ensemble de ces méthodes d'édition offre un bon contrôle à l'utilisateur. Cependant, la création de terrains réalistes exige une expérience considérable dans les outils d'édition et un travail méticuleux et long est nécessaire pour reproduire des formes de terrains spécifiques . En outre, les techniques d'édition existantes ne permettent pas de modéliser simultanément des terrains à grande échelle et un haut niveau de détail.

Chapitre 3 : Modélisation procédurale de plante

1.Introduction :

Les plantes ç'est la base de vie dans la terre, Le monde virtuel comprend de nombreux composants. Et nous ne pouvons pas tout modéliser et dans ce chapitre nous allons nous concentrer sur les plantes qui ce sont le deuxième élément qui joue un rôle clé pour la modélisation des mondes virtuels et la création d'une scène naturelle.

2. plante virtuelle :

Une plante virtuelle est un objet informatique qui décrit la plante comme un ensemble de composants représentant les organes (feuilles, tiges ou structures géométriques plus complexes) et dont le fonctionnement et l'interaction reproduisant plus ou moins fidèlement le processus et les mécanismes de croissance de plantes [35].

3 . Modélisation de plante

Il y a trois catégorie de Modélisation de plante existent : Modèles physiologiques et Les modèles morphologiques et Les modèles structure/fonction.

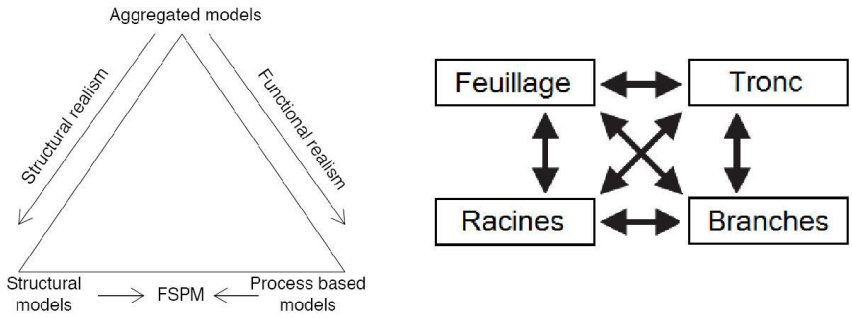

Figure 3.1. A gauche. Triangle de la modélisation des plantes. A droite. Compartimentation d'un modèle physiologique[36]

2.1 Modèles physiologiques.

1.1 Modèles physiologiques

Les modèles physiologiques sont orientes vers la représentation des processus métaboliques au sein d'une plante. L'attention est surtout portée sur les substances carboniques obtenues par photosynthèse car elles sont

déterminantes pour sa croissance. Pour cet objectif, il n'est souvent pas nécessaire de modéliser en détails l'architecture morphologique de la plante. Elle est généralement divisée en un certain nombre de compartiments essentiels tels que feuillage, tronc, branches et réseau racinaire. Le développement global est déduit des processus qui se déroulent à l'intérieur et entre ces composants en interaction avec l'environnement [37]. Les dynamiques sont typiquement exprimées par un système d'équations différentielles. La figure 3.2 montre un exemple d'une telle compartimentation.

Les premiers modèles physiologiques ont et conçus au début des années 70 et comprenaient un compartiment pour les feuilles et un autre pour les racines[38] Des lors, ce type de modèle a constamment et étendu et enrichi, par exemple par l'ajout de nouveaux compartiments tels que le tronc[39]ou de nouvelles substances telle que l'eau[40]. Ces modèles ont évolué jusqu'à devenir un outil éprouve pour la description, l'analyse et la compréhension de la croissance d'arbres et de forêts[41].

Concernant l'utilisation du carbone au sein d'une plante, plusieurs processus clef peuvent être identifiés[42] :

Assimilation: La photosynthèse permet d'accumuler du carbone en fonction de paramètres climatiques et l'état physiologique des feuilles. Le gain est ensuite mis à disposition pour son utilisation dans les autres processus.

Respiration: La perte en carbone a généralement deux composantes, la croissance et la maintenance. D'une part, la plante respire lors de la synthèse de nouvelle biomasse, d'autre part elle consomme une partie de son carbone pour maintenir sa biomasse existante.

Stockage: La gestion d'une réserve est généralement liée aux saisons, impliquant une accumulation de carbone avant l'hiver et sa mobilisation au printemps. Toutefois, la plupart des modèles physiologiques actuels ignorent ce processus.

Croissance: L'allocation du carbone représente le problème central des modèles physiologiques, due à un manque de connaissances des

mécanismes contrôlant ce processus chez les plantes naturelles[43]. Différentes conceptions ont et proposées. Les quatre approches les plus importantes seront présentées dans les sections suivantes.

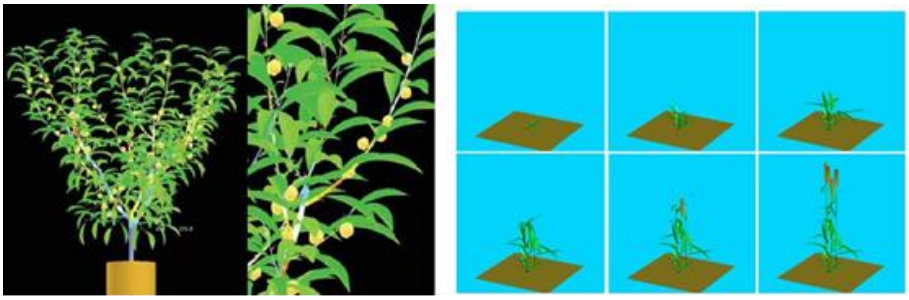

Figure 3.2. A gauche. Croissance d'un arbre fruitier par L-PEACH . A droite. Croissance du blé par ADEL-wheat[44]

2.2 Les modèles morphologiques.

Cette approche suit le développement architectural des plantes. Beaucoup de chercheurs et botanistes se sont intéressés à ce domaine comme Francis Halle et Roelof Oldeman qui ont conçu des modèles de croissance architecturale des arbres, donnant à cette un intérêt scientifique en recherche fondamentale. Elle a également eu des applications dans le domaine des images de synthèse [45].

Selon Goden, il existe trois types importants dans le processus de décomposition des informations morphologiques [46].

Type global : modéliser la structure d'une plante à un niveau grossier où l'agencement spatial n'est pas important. Ce type représente les comportements par des formes géométriques. Ainsi, cette approche est convenable pour les modèles physiologiques.

Type modulaire : c'est la méthode la plus utilisée dans la modélisation des plantes, elle considère une plante comme une composition itérative de modules élémentaires par exemple : les organes « feuille, tige, racine... » qui sont reliés par une topologie arborescente.

Type multi-échelles : c'est le type le plus détaillé, il définit une hiérarchie des unités constitutives.

2.3- Les modèles structure/fonction.

Ces modèles sont orientés pour étudier la morphologie et la physiologie des plantes, ils expliquent l'effet du processus physiologique sur l'accroissement morphologique d'une

plante. Selon Sievänen et al., les modèles morphologique et physiologique se complètent mutuellement [47].

Les représentants typiques des modèles structures/fonction.

L-PEACH : il simule la reproduction des arbres fruitiers, c'est un modèle physiologique qui a été amélioré par une description morphologique. Ainsi, ce modèle est convenable pour étudier l'effet de la morphologie sur la productivité d'un arbre fruitier [48].

AMAP : des chercheurs ont utilisé différentes générations des logiciels AMAP pour modéliser la morphologie des plantes, « Barczi et al. » ont conçu le logiciel AMAPSim qui permet de représenter l'accroissement d'une plante en fonction de plusieurs processus de construction où une plante est considérée comme la conséquence du fonctionnement d'un ensemble de méristèmes qui passent par plusieurs tests de développement. [49]

Nous pouvons également citer le modèle AMAPpara qui s'intéresse à la simulation de la croissance des plantes en prenant compte des connaissances physiologiques telles que la photosynthèse et la transpiration. Ce modèle est attentif à des interactions avec l'environnement qui peuvent être induites par la lumière et le climat. [50]

Dans ce contexte, un travail de collaboration a été entrepris en concertation entre des laboratoires de recherche français et chinois et ayant donné naissance au modèle Greenlab a été apparue. Ce modèle avait pour but d'optimiser la formalisation mathématique des modèles de plantes afin d'arriver à simuler les interactions des plantes pour des utilisations dans des applications de modélisation en agronomie [51].

ADEL : Fournier et Andrieu ont conçu un logiciel ADEL-Maiz[52] qui est destiné à la simulation des dynamiques d'une plantation agricole, ils ont créé un modèle qui utilise le formalisme des L-systèmes pour représenter la croissance morphologique du maïs. Nous pouvons également citer le modèle ADEL-Wheat conçu par [Fournier et al] pour simuler la croissance architecturale de la partie aérienne du blé [53].

3. Conclusion :

Nous avons présenté les trois catégorie de modélisation de plante existent de synthétiser des plantes 3D de géométrie et topologie quelconque pour représenter des surplombs, des arches et des grottes.

Dans notre étude on s'intéresse aux modèle L-système ou (modèles à synthèse procédurale) qui permet la génération automatique des plantes, parce qu'il ne nécessite pas un grand espace de stockage et ses résultats plus réalistes.

4. Modélisation procédurale de plante

La modélisation procédural de la plante est souvent un sujet qui intéresse et passionne. Lorsqu'on une scène Natural , dans cette partie Nous nous concentrons sur la modélisation procédural les plantes en choisir L-système l'axe de cette étude par ce que cette méthode c'est méthode très facile et procédurale et plus réalisme

4.1. L-système

4.1.1 Définition : Les L-systèmes ou les systèmes de Lindenmayer représentent un formalisme très utilisé pour la modélisation de la morphologie des plantes. Ils ont été proposés par Aristid Lindenmeyer et sont basés sur les grammaires formelles de Chomsky. Une grammaire comporte un ensemble de règles, l'application de ces règles à des entités élémentaires donne des chaines de caractères où la plante est représentée par cette chaine alors que chaque caractère représente un organe comme une feuille ou un bourgeon [46].

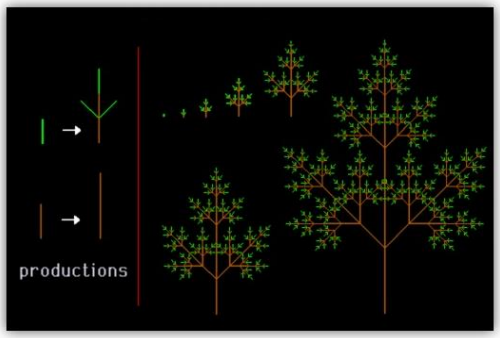

Figure 3.3 : exemple d'une plante générée par les L-systèmes.

Un L-système élémentaire est composé d'un triplet (A,ω,P) :

– A, un alphabet, c'est-à-dire un ensemble fini de symboles,

– ω, un axiome qui désigne le mot initial,

– P, un ensemble de règles de production définissant les transformations, décrites par "symbole prédécesseur → symboles successeurs".

En partant de l'axiome, chaque dérivation consiste à remplacer, de manière parallèle, tous les caractères auxquels une règle de production est associée [54].

Nous voyons dans cette figure des plantes virtuelles générées par les grammaires des L-systèmes ou chaque règle a donné une forme différente.

symboles	signification
F	Avancer de 1 (branche fertile - vert)
S	Avancer de 1(branche stérile - brun)
+	Tourner à gauche
-	Tourner à droite
[	Sauvegarder la position
]	Restaurer la position

Table 3.1 : symboles de L- systhem

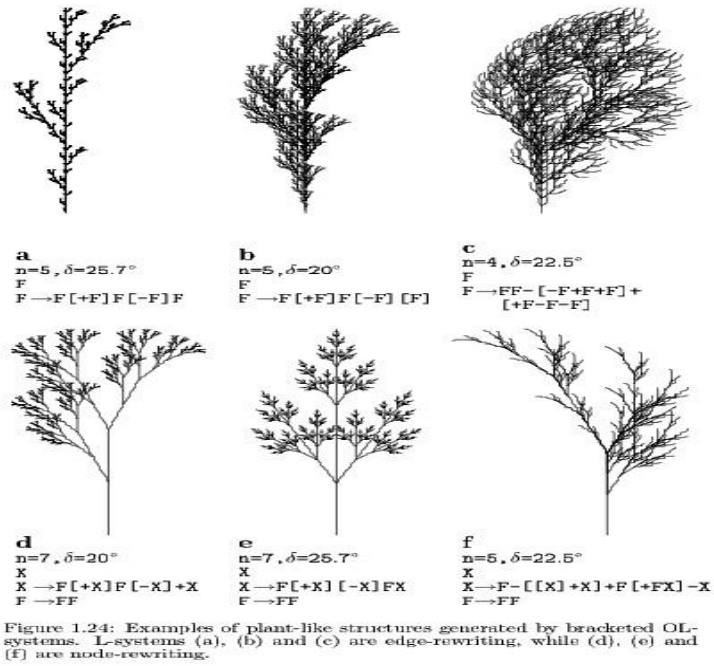

Figure 1.24: Examples of plant-like structures generated by bracketed OL-systems. L-systems (a), (b) and (c) are edge-rewriting, while (d), (e) and (f) are node-rewriting.

Figure 3.4 : Plantes virtuelles de différentes formes.

4.1.2 Les types de L-systèmes.

a) **L-systèmes stochastiques** : ils associent à chaque règle de rendement une possibilité de déclenchement.

b) **Les L-systèmes paramétriques** : ils permettent de représenter encore plus d'informations par le biais d'un symbole.

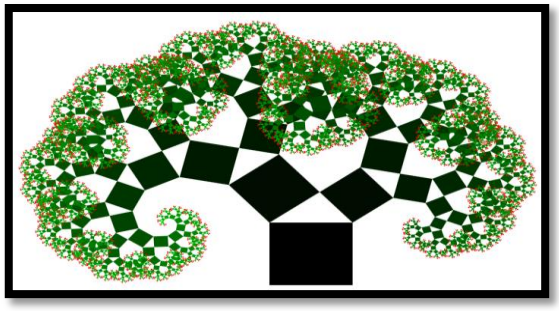

Figure 3.5. Arbre Pythagore créé avec système de L paramétrique.

c) **Les L-systèmes contextuels :** ils autorisent la modélisation de l'échange des ressources dans la plante, car les règles de production dépendent du caractère à remplacer et de ses proches également.

d) **Les L-systèmes ouverts :** ils donnent l'importance à des interactions de la plante avec l'environnement [46].

e) L-systèmes sensibles au contexte

La réécriture de symboles dans L-systèmes est sans contexte; les règles de réécriture sont non utilisées aux symboles indépendamment de leur contexte (les symboles autour d'eux). Cependant, la réécriture d'un symbole peut également dépendre de son contexte. Ceci est utile pour simuler le flux des signaux (nutriments ou des hormones) dans un modèle de plante qui, par exemple, tente de démontrer la croissance naturelle de la plante. [55]

Formellement, il existe deux types de L-systèmes sensibles au contexte, 1L-systèmes et 2L-systèmes. Les règles de réécriture de 1L-systèmes vérifie le contexte à un seul côté (gauche ou droit), alors que les règles de réécriture de 2L-systèmes vérifie le contexte des deux côtés. Depuis 1L-systèmes sont seulement 2L-systèmes avec un cadre vide, nous considérons L-systèmes sensibles au contexte que 2L-systèmes. [55]

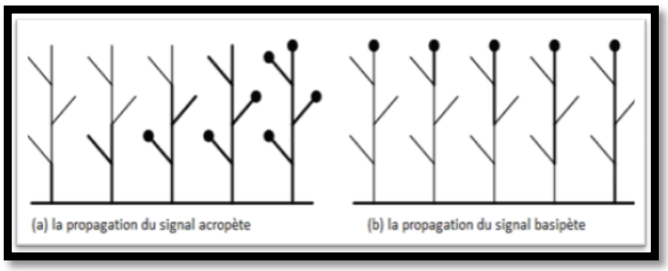

Figure 3.6. La propagation du signal simulé avec crochets L-systèmes sensibles au contexte.

- Les grammaires de forme, les grammaires de chaîne, ou encore les L-systèmes sont les fonctions les plus communes reliant les méthodes procédurales. Elles constituent un dénominateurcommunpourlesméthodesévoluéescommel'intégrationd'interfacevisuelle d'éditionde grammaire, l'apprentissage automatisé de grammaire, les librairies de règle, etc. Un bon aperçu de ces techniques peut s'obtenir en passant en revue les domaines d'application.

4.2. La différence entre multi-échelles et L-système .

C'est un modèle complémentaire aux L-systèmes. Cette approche est plus arriviste par rapport à celle des L-systèmes, elle est très générique et consiste à décomposer la structure globale d'une plante en sous-structures, et décomposer ensuite ses sous-structures jusqu'à arriver à un niveau de détail convenable. Pour chaque niveau, la structure de la plante est décrite par un graphe, après, tous les graphes sont intégrés dans un graphe multi-échelles. [56]

D'après l'auteur, malgré que cette approche offre beaucoup d'informations permettant de représenter plus de détails par rapport à d'autres qui utilisent moins de détails, il y'a lieu de constater qu'elle présente les inconvénients d'être plus complexe et non moins plus coûteuse [46].

4.3. Les algorithmes génétiques avec l-system et modélisation procédural

La contribution de consiste à développer pendant un projet de master, une combinaison de méthodes pour générer des formes 3D artistiques et imprimables car aucun modèle ne propose de solution complète pour générer et construire un objet artistique avec une imprimante 3D. Cet chapiter explique comment s'articulent les méthodes de génération procédurale évolutionnaires et les méthodes de modélisation et de rendu 3D. Le L-Système paramétrique offre une base à la génération de formes. De son interprétation graphique, résulte un squelette 3D leur permettant d'obtenir une forme en une seule composante. La génération supervisée repose sur un algorithme génétique, offrant à l'utilisateur d'explorer un espace de formes 3D qui garantit les critères topologiques qu'ils ont jugés minimum pour l'impression (une seule composante, surface fermée, épaisseur pas trop fine) en ne se focalisant que sur sa démarche artistique. Chaque squelette généré avec le L-Système et l'algorithme génétique est « habillé » d'une surface implicite de convolution offrant une surface de genre 2 (fermée) et le contrôle du rayon de convolution pour éviter une épaisseur trop fine à l'impression. De plus, le rendu par surface implicite est assez esthétique (lisse et coloré). En donnant l'accès à une interface (Figure 3.7) multiplateforme, conviviale et créative (visualisation des formes 3D générées, itération du L-Système, contrôle de l'algorithme génétique…) ainsi que des moyens de concrétisation accessibles, l'utilisateur acquiert une autonomie dans la création et dans la réalisation concrète d'objets en 3D.[57]

Les algorithmes génétiques, sont à la fois utilisés pour la simulation, et la modélisation de formes optimales en ingénierie

L'ensemble des règles peut être soit intégré dans l'algorithme, configurable par des paramètres, ou de l'ensemble des règles est séparé du moteur d'évaluation. La sortie est appelé contenu procédural, qui peut être utilisé dans les jeux vidéo, dans le cinéma, ou l'utilisateur peut modifier le contenu manuellement.

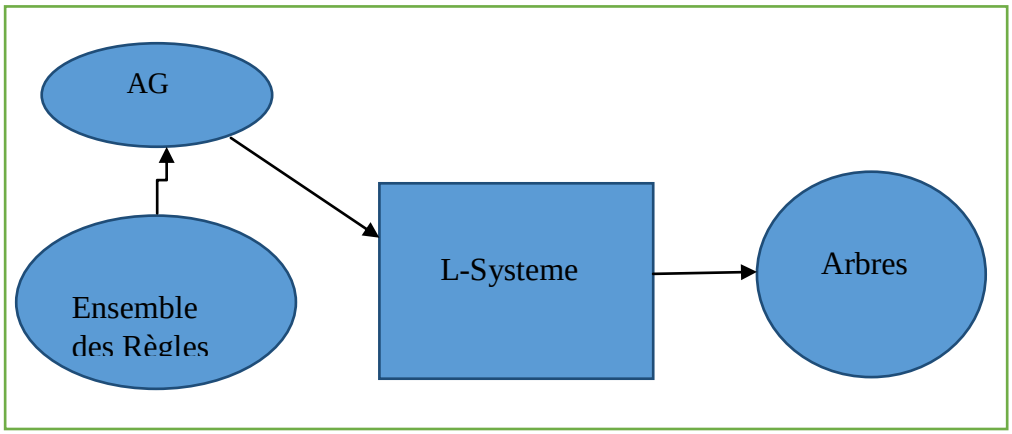

Figure 3.7 Structure Générale d'un L-System avec les AG.

4.4. Définition d'un algorithme génétique (AG) :

C'est un algorithme de recherche basé sur les mécanismes de la sélection naturelle et de la génétique. Il combine une stratégie de "survie des plus forts" avec un échange d'information aléatoire, mais structurée [58].

4.4.1. Opérations génétiques

Lors de l'utilisation d'un codage génétique particulière pour les organismes, il faut définir des opérations de reproduction spéciales ainsi. Ces opérations sont souvent plus élaborées que celles de l'algorithme génétique canonique. Nos chromosomes ont une structure syntaxique bien définie provenant de la L-System formalisme. Pour permettre une dérivation et l'interprétation des génotypes, nos opérations génétiques doivent produire une descendance avec des structures syntaxiques valides. Trois principaux opérateurs ont été conçus: croisement et deux types de mutation.

▪ Le croisement

Le croisement est l'échanger des parties des deux règles de L-systèmes (i.e. les deux individus qui vont subir un croisement pour en obtenir deux fils), cet échange va permettre et de créer deux nouvelles règles (de nouveaux individus).

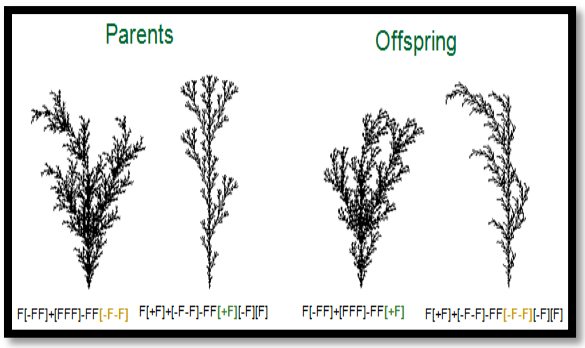

Figure. 3.8. Les parents et les fils d'un croisement.

➢ Mutation

La fonction de la mutation permet d'échanger deux caractères dans une même règle. Après ces deux tâches, les supports doivent être équilibrés pour en faire une règle valide.

Deux types de mutation ont été conçus. Chacun agissant sur distinct les parties de chromosomes:

- Symbole Mutation: Un symbole choisi au hasard du chromosome dans l'ensemble {F, +, -} est remplacée par une chaîne aléatoire mais syntaxiquement correcte.

- Bloc Mutation: Un bloc sélectionné au hasard dans le chromosome est substitué par une chaîne aléatoire syntaxiquement correcte.

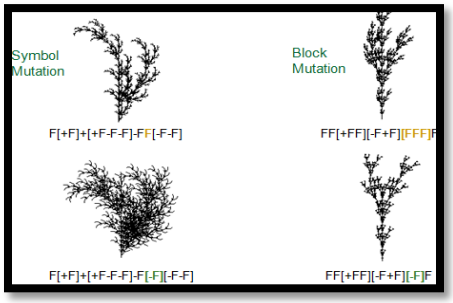

Figure.3.9. Parent et les fils des deux types de mutation.

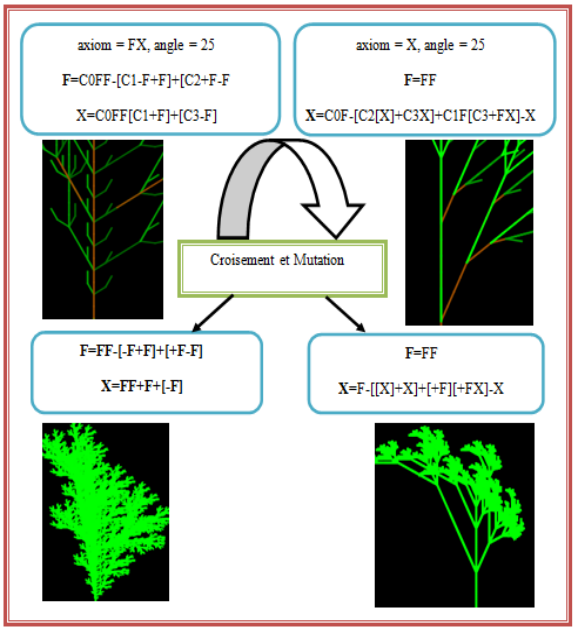

Figure.3.10 Forme arbre après croisement et mutation

4 .5. Simulateurs des arbres :

La simulation à laquelle nous sommes intéressés concerne les plantes. Simuler une plante consiste à la reproduire sur ordinateur en respectant ses caractéristiques de sa naissance jusqu'à sa mort, ceci peut être réalisé en utilisant des modèles virtuels.

Contient deux types de simulation Simulateurs géométriques et Simulateurs

4.5.1 Simulateurs géométriques

4.5.1.1. OnyxTree:

Ce logiciel est orienté pour des applications de type image de synthèse.

Figure.3.11 Exemples d'arbres produits par OnyxTree (thèse de **Jean-François BARCZI**)

4.5.1.2.Xfrog :

Basé sur le formalisme L-System, il permet de modéliser quasiment n'importe quelle espèce de plante mais malheureusement se base essentiellement sur la géométrie et propose peu d'outils basés sur la connaissance biologique des plantes

Il est similaire à SpeedTree et il est également utilisé dans le secteur des jeux vidéo. Il peut générer des scènes très réalistes, avec des objets déférents et une grande variété de plantes. C'est plus abordable que SpeedTree. Sa courbe d'apprentissage est également très raide et prend un certain temps pour obtenir un arbre convaincant. [59]

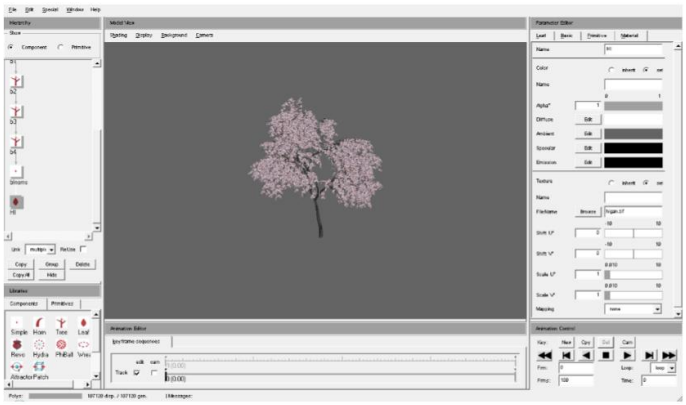

Figure.3.12. Interface utilisateur graphique Xfrog

Figure.3.13 Exemples plante produits par XFrog. (thèse de Jean-François BARCZI)

4.5.2. Simulateurs botaniques

- **Amap**

Issu du laboratoire éponyme du Cirad,

Ce modèle repose sur une approche relativement similaire à celle proposée par OnyxTree.

Figure.3.14 **Exemples d'arbres produits par Amap. (de Reffye, 1988)**

4.6 .Applications générale de simulation les plantes

FractTree

Cette application est l'un des précurseurs de la génération de plantes fractales. Elle crée uniquement des modèles 2D et utilise L-Systems ainsi qu'une génération étape par étape avec un niveau de détail pour les règles de dérivation illustrées à la figure 3.14. L'application crée l'installation en remplaçant les symboles de la dérivation par des primitives de dessin. C'est un programme très simple mais il peut être utilisé pour comprendre les bases de L Systems. [60]

Figure.3.15 **FractTree permet de restituer séparément chaque étape de la dérivation**

L-System4

Il est également basé sur L-Systems et génère des plantes et des objets 3D détaillés (voir la figure 3.15). La navigation est quelque peu restreinte mais il suffit d'examiner l'objet. L'un des problèmes de cette application est que l'utilisateur doit savoir comment L-Systems travaille pour en créer ou en modifier un [61]

Figure.3.16 **L'écran de prévisualisation et de navigation de L-System4**

LStudio

LStudio fournit plusieurs outils pour créer des installations réalistes, comme illustré à la figure 3.16.a. Il est basé sur un système L entre crochets modifié pour générer des troncs, des branches et la position des feuilles, des fleurs et des pétales. Ces terminaux sont modélisés

dans l'éditeur de vecteur interactif illustré à la figure 3.16.b. Ce système convient pour générer de petites plantes telles que des fleurs, des herbes et des arbustes plutôt que des arbres. [62]

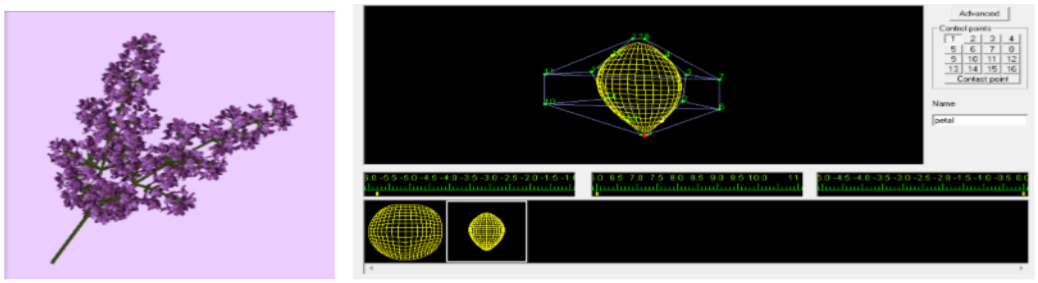

Figure 3.17 . a) Plante créée avec L-Studio, et b) son éditeur pour la modélisation des feuilles, des fleurs et des pétales

Un générateur de lierre

C'est un générateur de plantes de lierre qui permet à l'utilisateur de décider où les cultiver sur une scène 3D importée en définissant une graine. Il a des outils simples pour changer la apparence des plantes mais les résultats sont très réalistes. Il prend en compte la gravité et la capacité de la plante à se développer pour créer des plantes grimpantes ou suspendues (Figure 3.18). [63]

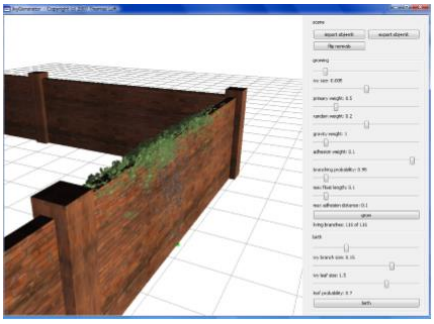

Figure.3.18. Un exemple de générateur de lier

TreeGenerator

Cette application dispose d'un panneau de configuration pour contrôler la génération de l'arbre. La figure 3.19 montre l'éditeur de feuille. Les feuilles qui en résultent semblent réelles lorsqu'elles sont isolées, mais les groupes de feuilles ne semblent pas réalistes. Une des causes est que le programme a un nombre limité de niveaux de récursivité pour générer des branches et des feuilles. Les outils permettant de modifier l'arborescence et de créer différentes instances sont également limités. [64]

Figure.3.19. Treegenerator GUI

TreeMagik

Cet outil fournit des troncs, des branches et un générateur de racines. Le feuillage est fourni par le programme et est affiché sous forme d'un ensemble de panneaux d'affichage. Comme le montre la figure 3.20, il génère de très bons résultats. Il est également capable de générer un panneau d'affichage de l'arborescence complète. Il fournit des textures pour le tronc et les feuilles, et l'utilisateur peut ajouter des textures. [65]

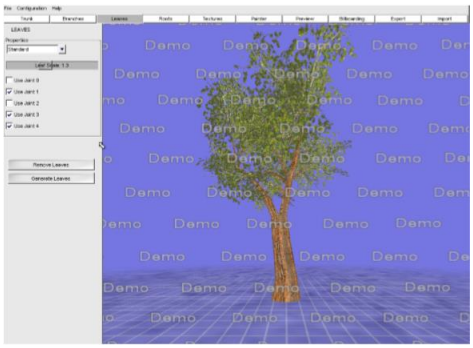

Figure.3.20. Exemple d'arborescence générée par TreeMagikG3

Studio MeshTree

Il produit des arbres très réalistes mais l'interface utilisateur n'est pas très conviviale et il n'y a pas d'aide ou de tutoriel. Une fois que la courbe d'apprentissage initiale est surmontée, l'utilisateur peut générer des arbres très attrayants avec un faible nombre de polygones. Une restriction est que ne génèrent que des fichiers .mesh. [66]

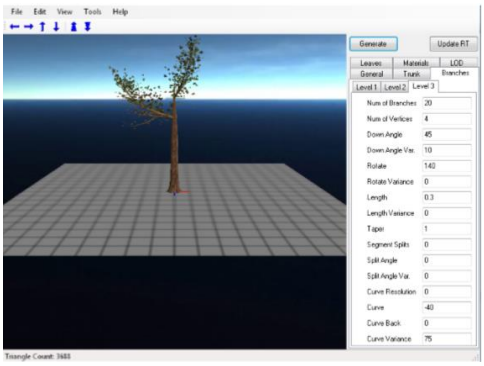

Figure.3.21. Exemple généré avec Meshtree Studio

Dryade

C'est un générateur d'arbre gratuit, mais ce n'est pas une source ouverte. Il fournit une galerie d'arbres en ligne qui ressemble à une forêt, où l'utilisateur peut sélectionner un arbre et modifier ses paramètres. Les propriétés de deux arbres différents peuvent être combinées pour créer un nouvel arbre. Les arbres créés par les utilisateurs peuvent être plantés dans la galerie en ligne et partagés avec d'autres utilisateurs. L'inconvénient de ce système est qu'il ne génère que des arbres à haute résolution et qu'ils ne sont pas très réalistes (Figure 3.22). [67]

Figure.3.22. Dryad génère des arbres rapidement, mais ils ne sont pas très réalistes

Arbaro :

Il s'agit du seul système basé sur Java évalué et qui génère de bons résultats. C'est bien documenté, mais son interface n'est pas très conviviale. Il y a des erreurs lors de l'exportation des arbres. La figure 3.23 montre la souplesse nécessaire pour contrôler des paramètres tels que le nombre de niveaux, le rayon, la division et les courbures des branches et du tronc. Il fournit des aides graphiques en superposition pour aider l'utilisateur à comprendre chaque paramètre. [68]

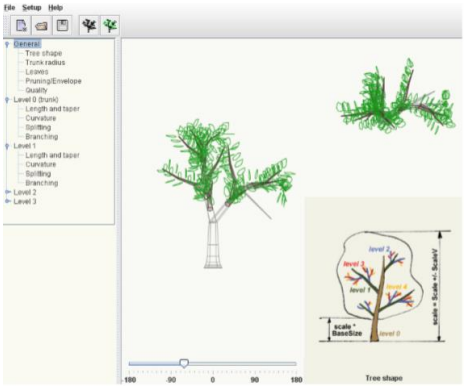

Figure.3.23. Le système d'aide d'Arbaro fournit des informations sur les paramètres permettant de contrôler la génération.

SpeedTree :

C'est l'un des produits les plus utilisés et les plus renommés dans la création de jeux vidéo présentant des scènes naturelles. Il fournit un moteur de rendu puissant, complet et efficace, un modélisateur et un générateur de temps réel. Les scènes créées avec ce moteur sont très

réalistes et il a également été conçu pour être utilisé dans des systèmes interactifs. Il s'agit du système le plus coûteux décrit dans cette enquête et nous n'avons pas pu le tester. [69]

Tree[d] :

Il est très facile de créer des arbres aléatoires et génère des exemples très réalistes. Il est di ffi cile d'en modifier ou d'en créer un nouveau, et il n'y a pas beaucoup de types di érents d'arbres. [70]

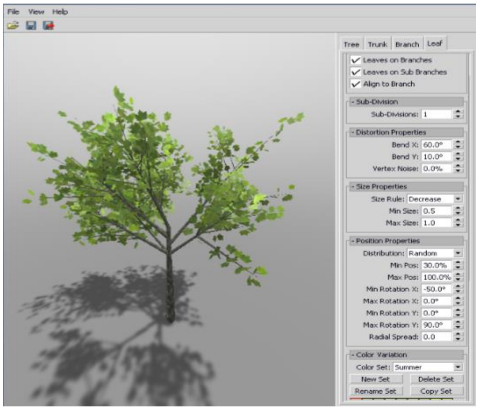

Figure.3.24. Tree donne de bons résultats mais il est difficile de contrôler les variations

5. Domaines d'application.

La simulation des plantes est très intéressante, elle est utilisée dans plusieurs domaines variés, comme le cinéma 3D, les simulateurs de vols, l'industrie du jeu vidéo « des jeux éducatifs qui aident à découvrir et comprendre les étapes de la croissance des plantes ». Elle est également utilisée dans le domaine de l'agronomie, de la foresterie, et de l'écologie, pour gérer, contrôler et prévoir l'état de la végétation. En effet, la simulation aide à comprendre le métabolisme des plantes et savoir réagir contre les obstacles qui empêchent leur développement. Ainsi, dans le cas de la sécheresse, nous pouvons l'utiliser pour savoir quel type de plante résiste le plus à ce défi [71].

6. Conclusion

Un problème avec cette approche est que les arbres en panneau d'affichage sont statiques et nous essayons de créer des arbres animés dynamiques. Une approche pour la mise à l'échelle du niveau de détail lors de la simulation d'arbres est décrite par Beaudoin et Keyser . Ils suggèrent de simplifier plusieurs branches en une seule branche 'moyenne' de telle sorte que la branche moyenne se rapproche des branches d'origine. Ce processus peut être répété sur les branches moyennes à mesure que la distance au téléspectateur augmente et que les détails sont laissés pour compte.

Chapitre 4 : Conception

4.1 Introduction:

La conception et la mise en œuvre sont deux activités qui sont importantes pour la réalisation d'une application.

Dans les chapitres précédents, nous avons étudié la modélisation procédural de terrain et des plantes.

Dans ce chapitre nous allons présenté la description de l'application, la conception globale et la conception détaillée de l'application.

4.2 Objectif

Notre travail concerne l'étude les méthodes procédural de terrains et plante qui sont considérés comme des objets naturels Notre objectif principal est généré un monde virtuel en utilisant la modélisation procédural afin d'aboutir un aspect simulation la plus naturelle possible.

4.3 Représentation général du system:

Voici la représentation du noter système d'une vue général.

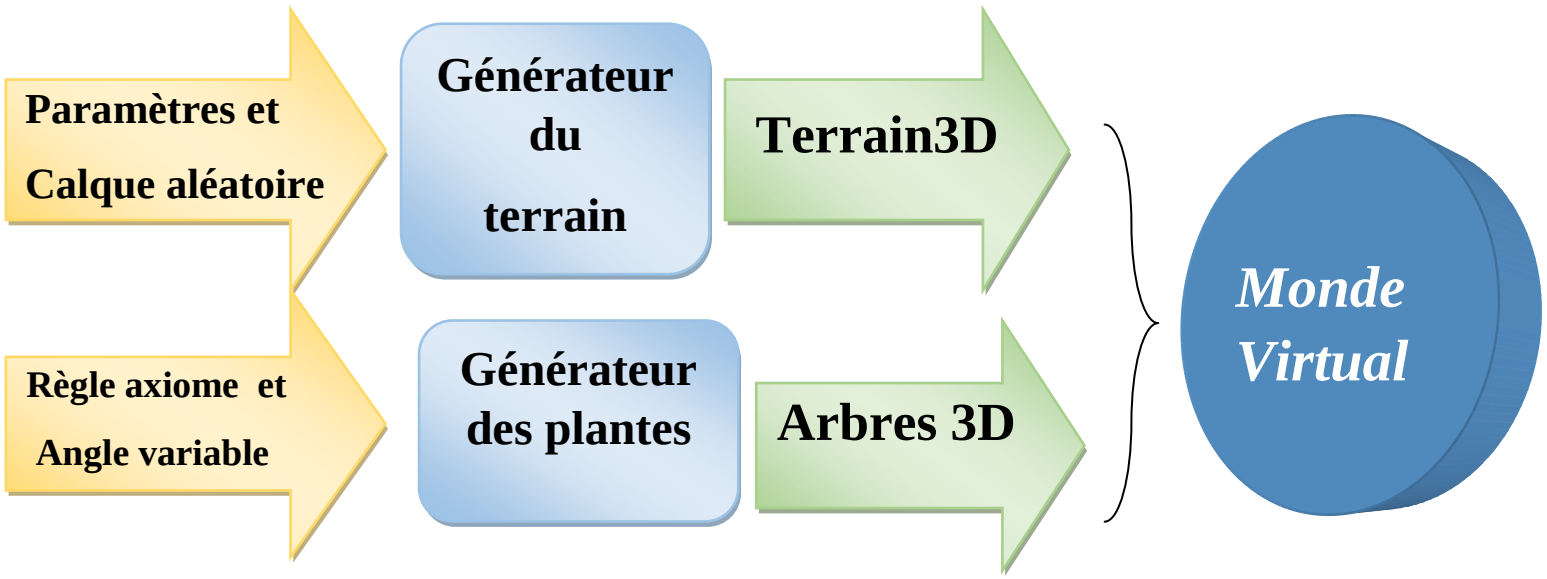

Figure. 4.1 - Représentation du system

4.4 Architecture détaillée du système:

Dans cette étape, nous allons expliquer l'architecture du système de l'application.

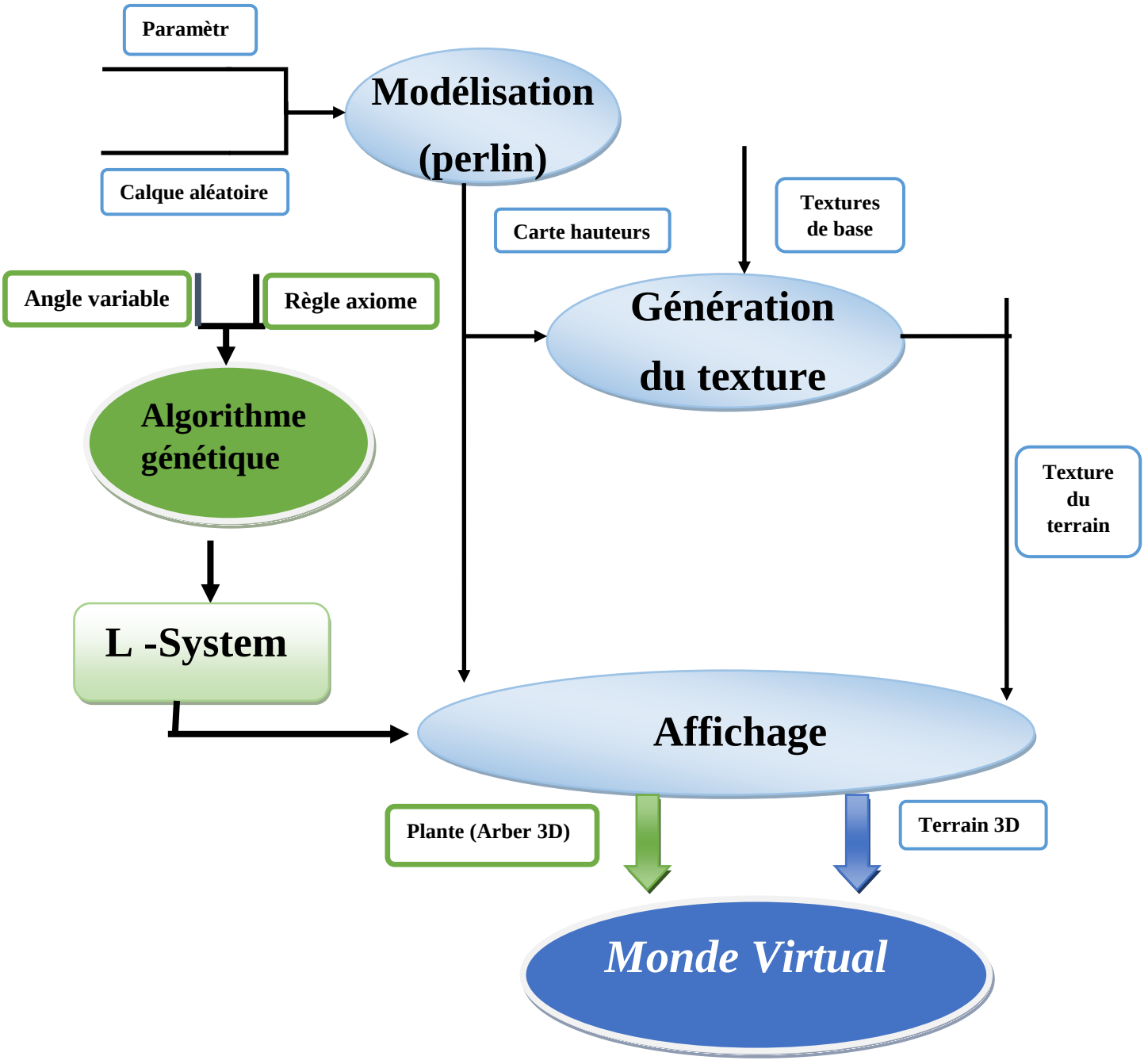

Figure 4.2 - architecture du system

Ce schéma représente l'architecture de notre système ,dans la première partie qui est la modélisation (l'algorithme perlin, nous avons choisi ce algorithme car il est facile à mettre en œuvre et nécessite moins de temps que d'autres, et aussi parce que les résultats sont plus réalistes), ses entrées sont la fréquence et le nombre des calques (octaves) comme paramètre et les calques aléatoires , et son résultat c'est le calcul de la carte des hauteurs que nous allons utiliser dans les différents étapes suivantes ,dans la génération de textures qui nécessite plus de la carte d'hauteurs, les textures de base (les images) , son sortie c'est la texture du terrain .

Deuxième partie composer deux méthode premier c'est algorithme génétique pour optimiser et Diversité dans les arbres

En fin la modélisation (L-système nous avons choisi ce méthode par ce que très facile et plus réalisme et plus rapide et aussi par ce que méthode procédural) .

4.5 Conclusion

Dans ce chapitre, nous avons détaillé la conception de notre système, ce qui constitue une étape importante vers la prochaine étape qui est la mise en œuvre

Chapitre 5 : implémentation (mise en œuvre)

5.1 Introduction

Nous découvrons dans ce dernier chapitre les différents outils que nous avons utilisés pour l'intégration et la mise en œuvre de notre approche. Le scénario de la génération de terrain, ainsi que les résultats seront présentés.

5.2 Environnement et matériel de développement

Notre système a été développé sous l'environnement :

5.2.1 Matériel :

MICRO PORTABLE: c'est un ordinateur portable DELL,

processeur Intel® CORE™ i5-5200U CPU @ 2.20GHz 2.20 GHz et 4GB de RAM, Système d'exploitation 64 bits. une carte graphique NVIDIA GeForce GTX 950M, en utilisant Microsoft Visual Studio 2010. Nous avons utilisé les bibliothèques suivantes : l'API OpenGL, la boîte à outils FreeGlut, les mathématiques OpenGL(GLM) et l'extension OpenGL(GLEW).

5.2.2.Logiciel :

Windows 10: Windows est le système d'exploitation vendu par Microsoft.

Nous avons utilisé les versions 10 pour quelques manipulations.

5.3 Outils de développement

5.3.1 C++ :

est un langage de programmation de niveau intermédiaire développé par Bjarne Stroustrup à partir de 1979 chez Bell Labs.

est un langage de programmation compilé, permettant la programmation sous de multiples paradigmes comme la programmation procédurale, la programmation orientée objet et la programmation générique. Le langage C++ n'appartient à personne et par conséquent n'importe qui peut l'utiliser sans besoin d'une autorisation ou obligation de payer pour avoir le droit d'utilisation. C++ est l'un des langages de programmation les plus populaires, avec une grande variété de plates-formes matérielles et de systèmes d'exploitation.

5.3.2.Visual C++ 2010

C++ est un langage de programmation d'usage universel. C'est un langage statique soutenant la programmation procédurale, l'abstraction de données, la programmation orientée objet, et la programmation générique. Depuis les années 90, C++ a été l'un des langages de programmation commerciaux les plus populaires. Visual C + + 2010 (également appelé Visual C + + 10.0) a été publié en Novembre 20110.

5.3.3 OpenGL (Open Graphics Library) :

est un ensemble normalisé de fonctions de calcul d'images 2D ou 3D lancé par Silicon Graphics en 1992. Cette interface de programmation est disponible sur de nombreuses plateformes

où elle est utilisée pour des applications qui vont du jeux vidéo jusqu'à la CAO en passant par la modélisation.

OpenGL permet à un programme de déclarer la géométrie d'objets sous forme de points, de vecteurs, de polygones, de bitmaps et de textures. OpenGL effectue ensuite des calculs de projection en vue de déterminer l'image à l'écran, en tenant compte de la distance, de l'orientation, des ombres, de la transparence et du cadrage.

5.4 Génération de terrain

Displacement map est une texture qui contient le déplacement d'une surface à chaque emplacement. Chaque patch représente une petite région d'un paysage qui est subdivisée en fonction de l'espace d'écran. Chaque sommet tessellé est déplacé le long de la tangente à la surface par la valeur stockée dans la carte de déplacement. Cela ajoute des détails géométriques à la surface sans avoir besoin de stocker explicitement les positions de chaque sommet tessellé. Au contraire, seuls les déplacements d'un paysage par ailleurs plat sont stockés dans la carte de déplacement et sont appliqués à l'exécution dans le shader d'évaluation de pavage. Displacement map (également appelée carte de hauteur) utilisée dans le projet est illustrée à la figure 5.1

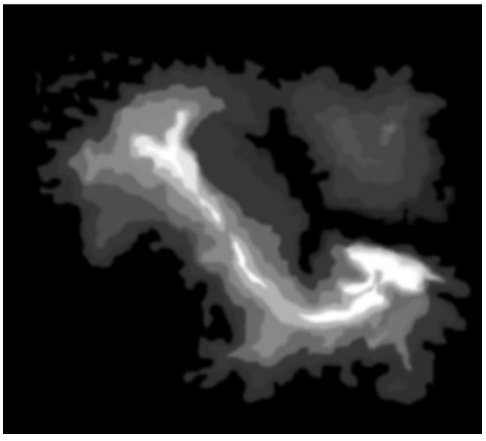

Figure 5.1 – Displacement map utilisée dans le terrain.

5.4.1Création du calque aléatoire

```
struct calque *random;
random = init_calque(taille,1);

for (i=0; i<taille; i++)
        for (j=0; j<taille; j++)
                random->v[i][j]=aleatoire(256);
```

5.4.2 Remplissage des calques

```
for (n=0; n<octaves; n++){
        for(i=0; i<taille; i++)
                for(j=0; j<taille; j++) {
                        a = valeur_interpolee(i, j, f_courante, random);
                        mes_calques[n]->v[i][j]=a;
                }
        f_courante*=frequence;
}
```

5.4.3 Interpolations des valeurs

```
int valeur_interpolee(int i, int j, int frequence, struct calque *r){
        /* déterminations des bornes */
        int borne1x, borne1y, borne2x, borne2y, q;
        float pas;
        pas = (float)r->taille/frequence;

        q = (float)i/pas;
        borne1x = q*pas;
        borne2x = (q+1)*pas;

        if(borne2x >= r->taille)
                borne2x = r->taille-1;

        q = (float)j/pas;
        borne1y = q*pas;
        borne2y = (q+1)*pas;

        if(borne2y >= r->taille)
                borne2y = r->taille-1;

        /* récupérations des valeurs aléatoires aux bornes */
        int b00,b01,b10,b11;
        b00 = r->v[borne1x][borne1y];
        b01 = r->v[borne1x][borne2y];
        b10 = r->v[borne2x][borne1y];
        b11 = r->v[borne2x][borne2y];

        int v1  = interpolate(b00, b01, borne2y-borne1y, j-borne1y);
        int v2  = interpolate(b10, b11, borne2y-borne1y, j-borne1y);
        int fin = interpolate(v1, v2, borne2x-borne1x , i-borne1x);

        return fin;
}
```

5.4.4 Ajout de tous les calques

```
for (i=0; i<octaves; i++)
        sum_persistances+=mes_calques[i]->persistance;

/* ajout des calques successifs */
for (i=0; i<taille; i++)
        for (j=0; j<taille; j++){
                for (n=0; n<octaves; n++)
                        c->v[i][j]+=mes_calques[n]->v[i][j]*mes_calques[n]->persistance;

                /* normalisation */
                c->v[i][j] =  c->v[i][j] / sum_persistances;
        }
```

5.5 Génération de planet (arbre):

5.5 .1 les procédure utiles :

➢ Procedure rotation a gauche

```
void rotL(){
                        glRotatef(ANGLE, 1, 0, 0);
                         glRotatef(ANGLE*4, 0, 1, 0);
                         glRotatef(ANGLE, 0, 0, 1);
}
```

➢ Procedure rotation a droite

```
void rotR(){
                         glRotatef(-ANGLE, 1, 0, 0);
                         glRotatef(ANGLE*4, 0, 1, 0);
                         glRotatef(-ANGLE, 0, 0, 1);
}
```

➢ Procedure de feuille

```
void leaf(){
                         glPushAttrib(GL_LIGHTING_BIT);//saves current lighting stuff
                                 //glColor3f(0.50, 1.0, 0.0);
                                 GLfloat ambient[4] = { 0.50, 1.0, 0.0 };    // ambient reflection
                                 GLfloat specular[4] = { 0.55, 1.0, 0.0 };   // specular reflection
                                 GLfloat diffuse[4] = { 0.50, 0.9, 0.0 };   // diffuse reflection
                                 // set the ambient reflection for the object
                                 glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT, ambient);
                                 // set the diffuse reflection for the object
                                 glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, diffuse);
```

```
                    // set the specular reflection for the object
                    glMaterialfv(GL_FRONT_AND_BACK, GL_SPECULAR, specular);
                    // set the size of the specular highlights
                    glMaterialf(GL_FRONT_AND_BACK, GL_SHININESS, 20.0);


                glBegin(GL_TRIANGLES);
                glVertex3f(0, 0, 0);
                glVertex3f(1.7, 0, 0.4);
                glVertex3f(0, 1, 0);
                glVertex3f(0,0, 0);
                glVertex3f(-1.7, 0, -0.4);
                glVertex3f(0, 1, 0);
                glEnd();
                glPopAttrib();
}
```

5.5.2condition de création de feuille :

```
if (num < 0.4){
                        //LSystem.replace(i, 1, "D[LX]D[RX]LX");
                str.replace(i, 1, "D[LXV]D[RXV]LX ");/*"D[LXV]D[RXV]LX"*/

                } else {
                        //LSystem.replace(i, 1, "D[RX]D[LX]RX");
                        str.replace(i, 1, "D[RXV]D[LXV]RX");
                }
```

5.5.3 procedure L-system

```
        for (int i = 0; i < LSystem.length(); i++){
                ch = LSystem.at(i);

                if (ch.compare("D") == 0 || ch.compare("X") == 0){ // D =debut
                        drawLine();
                } else if (ch.compare("[") == 0){
                        push();
                } else if (ch.compare("]") == 0){
                        pop();
                } else if (ch.compare("V") == 0){
               leaf();rotR();leaf();rotR();leaf();rotR();leaf();rotL();leaf();rotL();leaf();
                } else if (ch.compare("R") == 0){
                        rotR(); // rotation a droite
                } else if (ch.compare("L") == 0){
                        rotL();// rotation a gauche
                }
        }
```

}

5.6 Structure d'application générale

les entrées

Triangle

Ligne

Les sorties

Image 3D

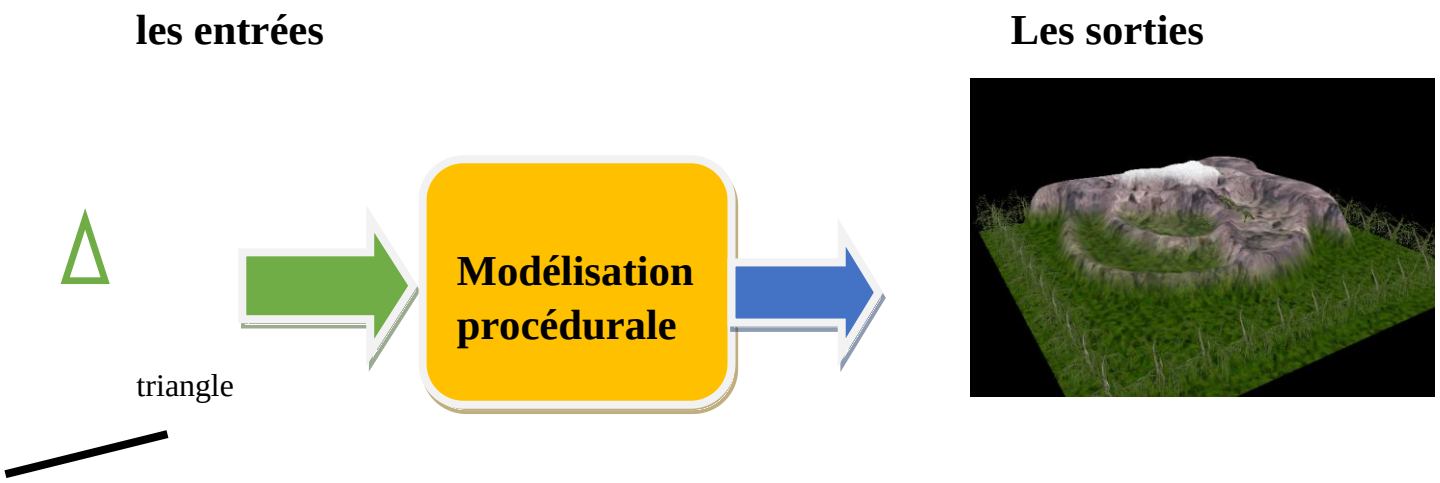

Figure 5.2 Structure d'application générale

5.7 Résulta :

Voici quelque capture qui démontre le fonctionnement de notre application :

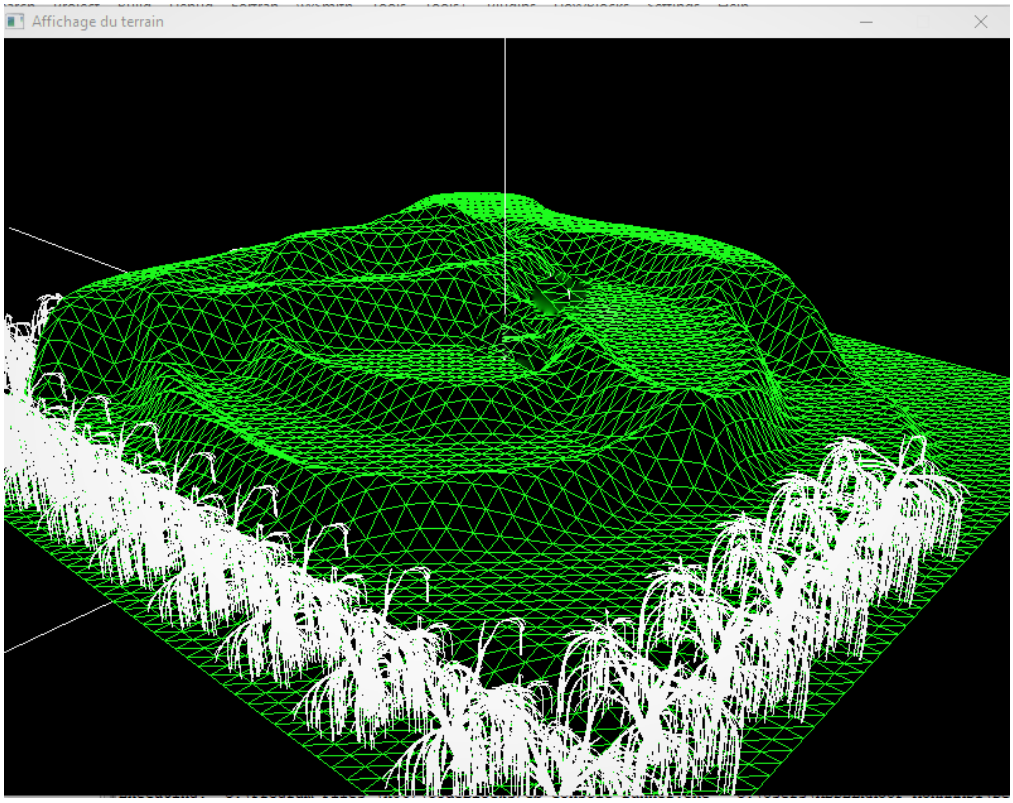

Figure 5.3 - Mode filiare

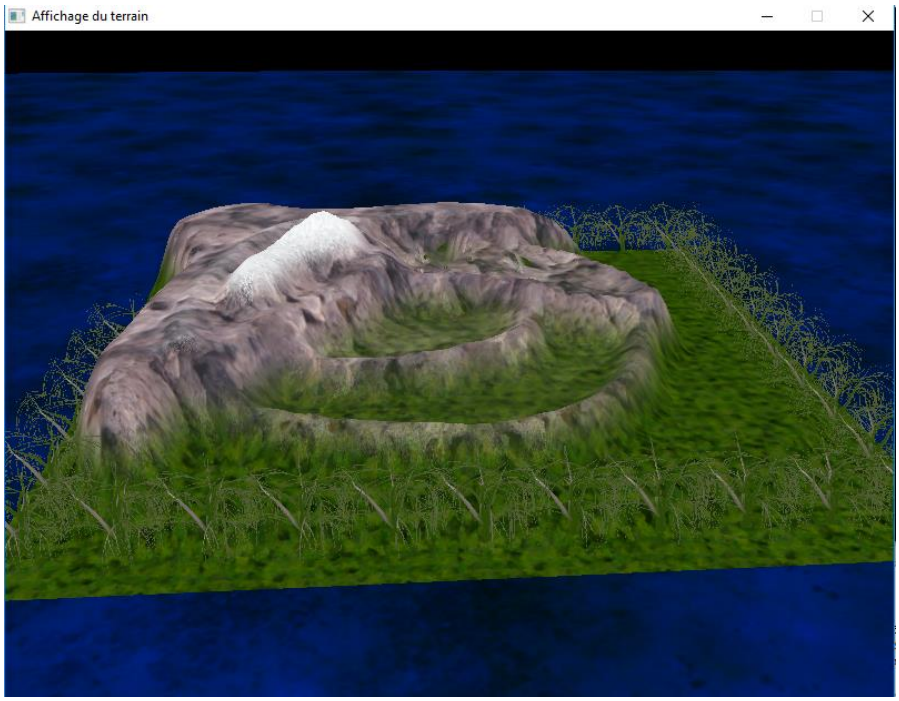

Figure 5.4 – Mode Texturer 1

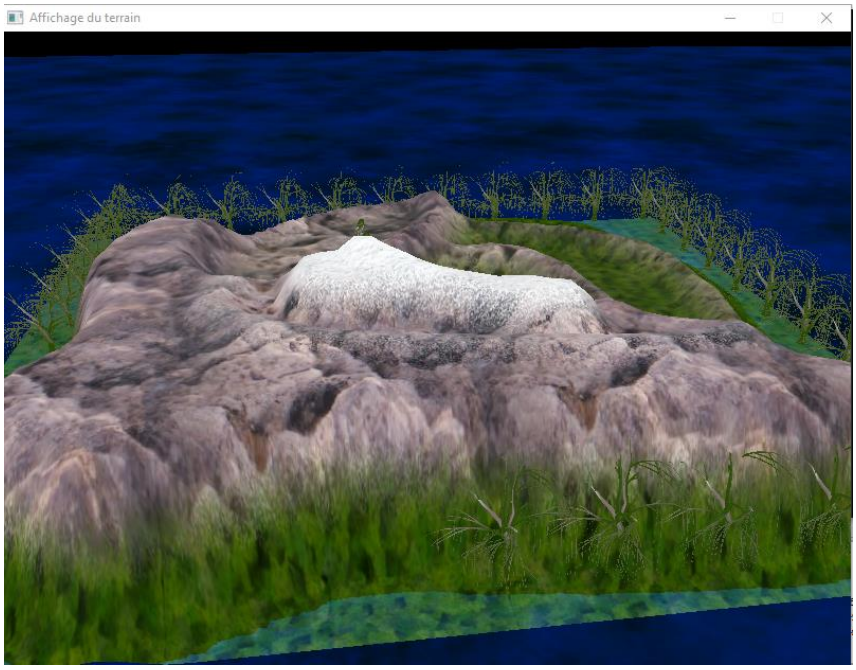

Figure 5.5 – Mode Texturer

Conclusion générale et perspectives

A travers ce travail, nous avons effectué une étude sur les méthodes procédurales pour la génération des terrains et des plantes qui jouent un rôle fondamental dans la création d'une scène naturelle. Les techniques de génération automatique ont de nombreuses applications comme les simulateurs de vol, les effets spéciaux au cinéma ou les jeux vidéo. L'augmentation de la puissance de calcul combinée à des méthodes de visualisation de plus en plus performantes ont permis de créer des terrains et des plantes réalistes.

En réalisant ce travail, nous avons pu acquérir :

– Des connaissances sur la modélisation procédural des terrains et des plantes qui ouvre une grande porte vers la recherche des techniques spéciales pour générer les terrains et les plantes .

– Des idées générales sur des catégories modélisation procédural.

Nous avons exploité ces connaissances pour développer et générer un monde virtuel plus réalistes et plus interactives. L'objectif global du projet est atteint. Cependant, plusieurs perspectives commencent à être envisagées. Nous pensons que ce travail peut être une brique de base pour des travaux plus affinés qui peuvent traiter :

– Simuler plus d'effets de réalisme tel que les inter réflexions par approximation sans utiliser des modèles globaux et augmenter la complexité.

– Intégrer l'aspect multi-résolution (LOD, programmation via le matériel graphique,...).

– Possibilité d'intégrer des objets (conçue par un logiciel graphique) telque et les masses d'eau, les réseaux de routes, la configuration urbaine, les bâtiments et les intérieurs de bâtiment dans le terrain de synthèse pour crée un monde Virtual plus réaliste .

Les fondements théoriques pour ces différents axes sont élaborés. Mais un travail important est exigé, pour passer de cet aspect théorique, vers une réalisation fructueuse.

Bibliographies

[1] Adrien Peytavie THESE DE L'UNIVERSIT ` E DE LYON : Génération procédurale de monde

[2] Ruben M. Smelik1, Tim Tutenel2, Rafael Bidarra2 and Bedrich Benes3A Survey on Procedural Modeling for Virtual Worlds

[3] A Survey of Procedural Methods for Terrain Modelling∗ Ruben M. Smelik, Klaas Jan de Kraker and Saskia A. Groenewegen

[4] Perlin K.: An image synthesizer. SIGGRAPH Comput. Graph. 19, 3 (1985), 287– 296.

[5] Ebert D. S., Musgrave F. K., Peachey D., Perlin K., Worley S.: Texturing and Modeling: A Procedural Approach, 3rd ed. Morgan Kaufmann, 2002.

[6] Bokeloh M., Wand M.: Hardware accelerated multi-resolution geometry synthesis. In Proceedings of the Symposium on Interactive 3D Graphics and Games (I3D) (2006), ACM, pp. 191–198.

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

[8] Pytel A., Mann S.: Self-organized approach to modeling hydraulic erosion features. Computers & Graphics 37, 4 (2013), 280–292.

[9] Št'ava O., Beneš B., Brisbin M., Křivánek J.: Interactive terrain modeling using hydraulic erosion. In Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA) (2008), pp. 201–210.

[10] Yu Q., Neyret F., Bruneton E., Holzschuch N.: Scalable Real-time Animation of Rivers. Computer Graphics Forum (Eurographics) 28, 2 (2009), 239–248.

[11] Yu Q., Neyret F., Steed A.: Feature-based vector simulation of water waves. Computer Animation and Virtual Worlds 22, 2-3 (2011), 91–98.

[12] van Hoesel F.: Tiled directional flow. In SIGGRAPH Posters (2011), ACM, pp. 19:1–19:1.

[13] Génevaux J.-D., Galin E., Guérin E., Peytavie A., Beneš B.: Terrain generation using procedural models based on hydrology. ACM Transactions on Graphics (SIGGRAPH) 32, 4 (2013), 143:1–143:13.

[14] Kelley A. D., Malin M. C., Nielson G. M.: Terrain simulation using a model of stream erosion. SIGGRAPH Comput. Graph. 22, 4 (1988), 263–268.

[15] Bokeloh M., Wand M., Seidel H.-P.: A connection between partial symmetry and inverse procedural modeling. ACM Transactions on Graphics (SIGGRAPH) 29, 4 (2010), 104:1–104:10.

[16] Lindenmayer A.: Mathematical models for cellular interactions in development i. filaments with one-sided inputs. Journal of Theoretical Biology 18, 3 (1968), 280–299.

[17] Prusinkiewicz P., Lindenmayer A., Hanan J. S., Fracchia F. D., Fowler D. R., de Boer M. J., Mercer L.: The algorithmic Beauty of Plants. Springer, 1990.

[18] Měch R., Prusinkiewicz P.: Visual models of plants interacting with their environment. In SIGGRAPH Comput. Graph. (1996), ACM, pp. 397–410.

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

[20] Parish Y. I. H., Müller P.: Procedural modeling of cities. In SIGGRAPH Comput. Graph. (2001), ACM, pp. 301–308.

[21] Galin E., Peytavie A., Guérin E., Beneš B.: Authoring Hierarchical Road Networks. Computer Graphics Forum (Eurographics) 30, 7 (2011), 2021–2030.

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

[23] Leblanc L., Houle J., Poulin P.: Component-based modeling of complete buildings. In Proceedings of Graphics Interface (2011), pp. 87–94.

[24] SMITH A. R.: Plants, fractals, and formal languages. In SIGGRAPH'84: Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques (New York, NY, USA, 1984), ACM Press, pp. 1–10.

[25] Génération bio-inspirée de forme 3D artistiques pour l'impression3D, Journées de l'Association Française d'Informatique Graphique, 2013, C.Despart, L.Brath et Luga, Université de Toulouse.

[26] MANDELBROT B, The Fractal Geometry of Nature, W. H. Freeman, August 1982.

[27] GAIN J., MARAIS P., STRASSER W. : Terrain sketching. In Proceedings of the 2009 symposium on Interactive 3D graphics and games (2009), pp. 31–38.

[28] Jean-David Génevaux1, Éric Galin1, Éric Guérin2, Adrien Peytavie3, Bedˇrich Beneš4. Génération procedural de rivières et de terrains

[29] F. Musgrave and B. Mandelbrot, The art of fractal landscapes, IBM Journal of Research and Development, 35(4), pp. 535-540, July 1991.

[30] NEIDHOLD B., WACKER M., DEUSSEN O. : Interactive physically based fluid and erosion simulation. In Eurographics Workshop on Natural Phenomena (2005), pp. 25–32.

[31] F. Musgrave and B. Mandelbrot, The art of fractal landscapes, IBM Journal of Research and Development, 35(4), pp. 535-540, July 1991.

[32] Andre Lamothe Focus on 3D Terrain Programming, Premier Press, 2003.

[33] Ken Perlin, Better acting in computer games : the use of procedural methods, Com- puters and Graphics, 26(1), pp. 3-11, February 2002.

[34] ALSWEIS M., DEUSSEN O.: Modeling and visualization of symmetric and asymmetric plant competition. In Proceedings of Euro graphics Work shop on Natural Phenomena(Dublin, Ireland, 2005), Euro graphics Association, pp. 83–88.

[35] Inria. Modéliser le vivant [en ligne] (page consultée le 21/05/2016).

http://www.inria.fr/centre/sophia/actualites/modeliser-le-vivant-creer-des-plantes-virtuelles

[36] Weiner, J. (1990). Asymmetric competition in plant populations. Trends in Ecology and Evolution, 5 :360–364.

[37] Makela, A., Landsberg, J., Ek, A., Burk, T., Ter-Mikaelian, M., Agren, G., Olver, C., and Puttonen, P. (2000). Process-based models for forest ecosystem management : Ccurrent state of the art and challenges for practical implementation. Tree Physiology, 20 :289–298

[38] Thornley, J. (1972b). A model to describe the partitioning of photosynthate during vegetative plant growth. Annals of Botany, 36 :419–430.

[39] Deleuze, C. and Houllier, F. (1997). A transport model for tree ring width. Silva Fennica, 31(3) :239–250.

[40] Sperry, J., Adler, F., Campbell, G., and Comstock, J. (1998). Limitation of plant water use by rhizosphere and xylem conductance : Results from a model. Plant, Cell and Environment, 21 :347–360.

[41] Landsberg, J. (2003). Physiology in forest models : History and the future. Forest. Biometry, Modelling and Information Sciences, 1 :49–63

[42] Le Roux, X., Lacointe, A., Escobar-Gutierrez, A., and Le Dizes, S. (2000). Carbonbased models of individual tree growth : A critical appraisal. Annals of Forest Sciences, 58 :469–506.

[43] Cannell, M. and Dewar, R. (1994). Carbon allocation in trees - a review of concepts for modeling. Advances in Ecological Research, 25 :59–104.

[44] Fournier, C., Andrieu, B., Ljutovac, S., and Saint-Jean, S. (2004). ADEL-Wheat : A tool for simulating the 3D architectural development of wheat., Montpellier, France, page 288.

[45]Weber, J. and Penn, J. (1995). Creation and rendering of realistic trees. In Computer Graphics Proceedings, ACM SIGGRAPH'95, Los Angeles, CA, USA, pages 119–128

[46]StefanBornhofen,ClaudeLattaud,(2007).Simulationdecommunautésdeplantes et dynamique des populations, 26.

[47] Go din, C. and Sino quet, H. (2005). Functional-structural plant mo delling. New Phytologist, 166(3) :705–708.

[48]Grossman, Y. and De Jong, T. (1994). PEACH : A simulation model of reproductive and vegetative growth in peach trees. Tree Physiology, 14(4) :329–345.

[49]Barczi, F., de Reffye, P., and Caraglio, Y. (1997). Essai sur l'identification et la mise en œuvre des paramètres nécessaires a la simulation d'une architecture végétale. Science Update, INRA Editions, Paris

[50]de Reffye, P., Fourcaud, T., Blaise, F., Barthelemy, D., and Houllier, F. (1997). A functional model of tree growth and tree architecture. Silva Fennica, 31 :297–311.

[51] de Reffye, P. and Hu, B. (2003). Relevant choices in botaniy and mathematics for bulting efficient dynamic plant growth models, editors, Plant Growth Models and Applications. Tsinghua University Press, Beijing

[52] Fournier, C. and Andrieu, B. (1999). ADEL-Maize : An L-system based model for the integration of growth processes from the organ to the canopy. Agronomie, 19 :313–327.

[53]Fournier, C., Andrieu, B., Ljutovac, S., and Saint-Jean, S. (2003). ADEL-Wheat : A 3D architectural model of wheat development. In Proceedings of the International Symposium on plant Growth Modeling, Simulation, Beijing, China, pages 54–66.

[54] Armando de la Re, Francisco Abad, Emilio Camahort, and M.C. Juan Tools for Procedural Generation of Plants in Virtual Scenes

[55] JosefPelikán. L-systems (2012), Programme d'études: InformatiqueSpécialisation, Prague2 012.

[56] Marie-Paule Cani ,Przemyslaw Prusinkiewicz Modélisation multi-échelle procédurale de scènes animées

[57] Génération bio-inspirée de forme 3D artistiques pour l'impression3D, Journées de l'Association Française d'Informatique Graphique, 2013, C.Despart, L.Brath et Luga, Université de Toulouse.

[58] Nicolas Barnier, Pascal Brisset, (1999). Optimisation par algorithme génétique sous Contraintes,18.

[59]. Xfrog: http://www.xfrog.com/

[60]. FracTree: http://archives.math.utk.edu/software/msdos/fractals/fractree

[61]. L-System4: http://www.geocities.com/tperz/L4Home.htm

[62]LStudio: http://algorithmicbotany.org

[63]. An Ivy Generator: http://graphics.uni-konstanz.de/~luft/ivy_generator

[64]. TreeGenerator: http://www.treegenerator.com

[65]. TreeMagik G3: http://www.aliencodec.com/product_treemagik.php

[66]. MeshTree Studio: http://www.ogre3d.org/forums/viewtopic.php?t=25909

[67]. Dryad: http://dryad.stanford.edu

[68]. Arbaro: http://arbaro.sourceforge.net

[69]. SpeedTree: http://www.speedtree.com

[70]. Tree[d]: http://www.frecle.net/forum/viewtopic.php?t=780

[71] Ludovic Hamon, (2011). Modélisation et interaction temps réel avec des structures dynamiques de type L-système : Application aux plantes virtuelles. Thèse de Doctorat. Angers : École Doctorale Sciences et Technologies de l'Information et des Mathématiques (STIM).