Génération de nature fractale

Références

L'initiative de la réalisation de ce projet est due à un livre, qui en a aussi été le support principal : « *The Algorithmic Beauty of Plants »,* écrit par Przemyslaw Prusinkiewicz et Aristid Lindenmayer, respectivement chercheur en informatique et biologiste.

La page wikipédia https://en.wikipedia.org/wiki/L-system aura aussi beaucoup servi à la compréhension du sujet.

But

Le but de ce projet était d'une part de porter les systèmes de Lindenmayer décrit dans la page wikipédia citée plus haut à la troisième dimension, puis de tenter de les utiliser afin de générer des fractales tendant à ressembler à des plantes. Ces systèmes sont particulièrement adaptés à cela, la preuve en est que leur inventeur, Aristid Lindenmayer, les a créés dans le but de modéliser le développement et la croissance des plantes.

Réalisation

Le choix a été fait de réaliser ce projet dans le langage python afin de passer moins de temps sur l'implémentation à proprement parler et de pouvoir avancer plus loin dans la recherche de la ressemblance entre les fractales générées et la nature elle-même. La bibliothèque pyopengl aura servi de lien entre openGL et python.

Pygame est utilisé uniquement afin de générer le contexte pour openGL.

Génération de L-System en 2 dimensions

Un L-System est une grammaire formelle composée d'un alphabet, d'un ensemble de règles et d'un axiome de départ. Voici un exemple de L-Système simple :

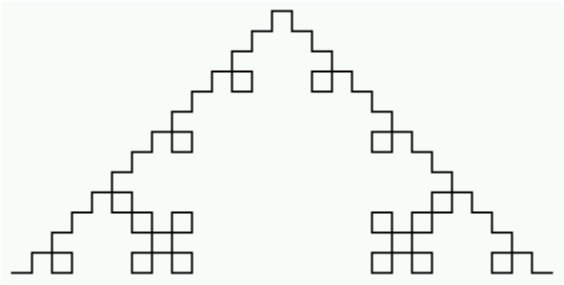
Alphabet : { F, +, - }
Axiome : F
Règles : { F -> F+F-F-F+F }

L'interprétation d'un tel système se fait en deux étapes :

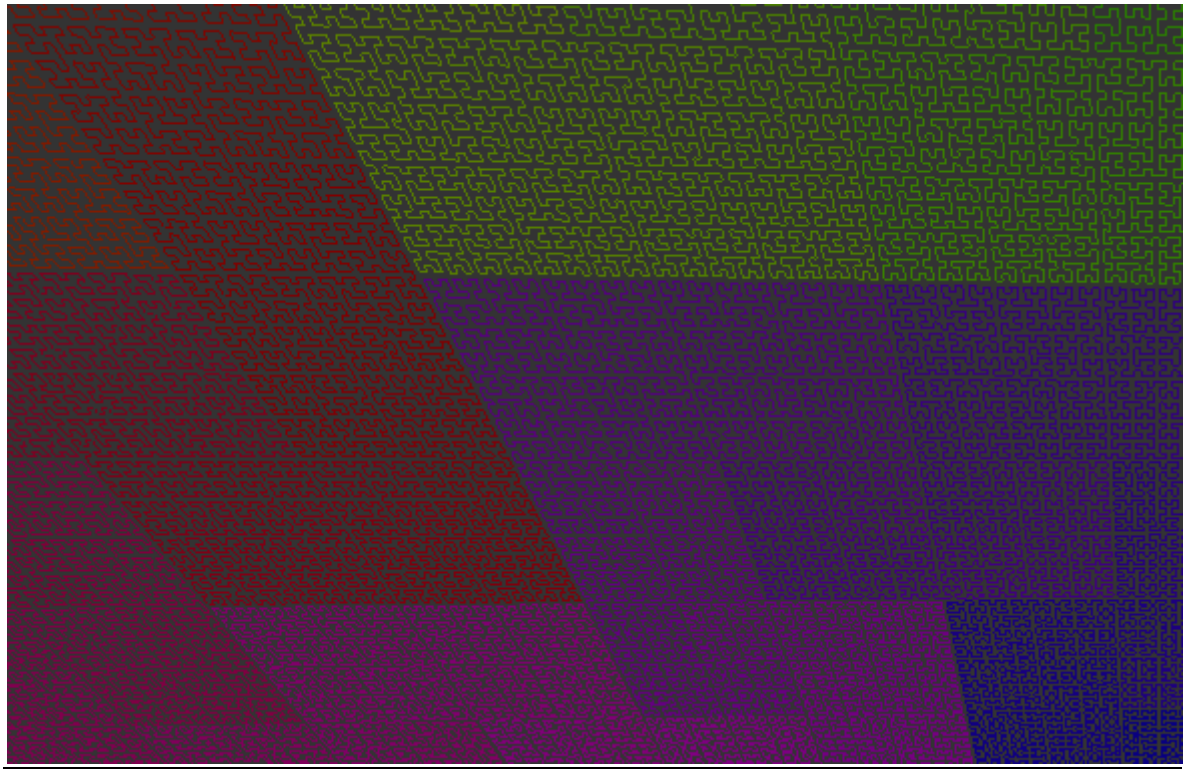
La première est la génération. Chaque lettre de l'axiome de départ est remplacée par le résultat de la règle correspondante. La nouvelle chaîne ainsi générée sert ensuite d'axiome de départ pour l'itération suivante.

La seconde étape est l'interprétation graphique de la chaîne générée. Dans l'exemple ci-dessus, F signifie « avancer », + signifie « tourner à gauche » et – signifie « tourner à droite ».

Le L-System peut maintenant être dessiné.



Courbe de koch



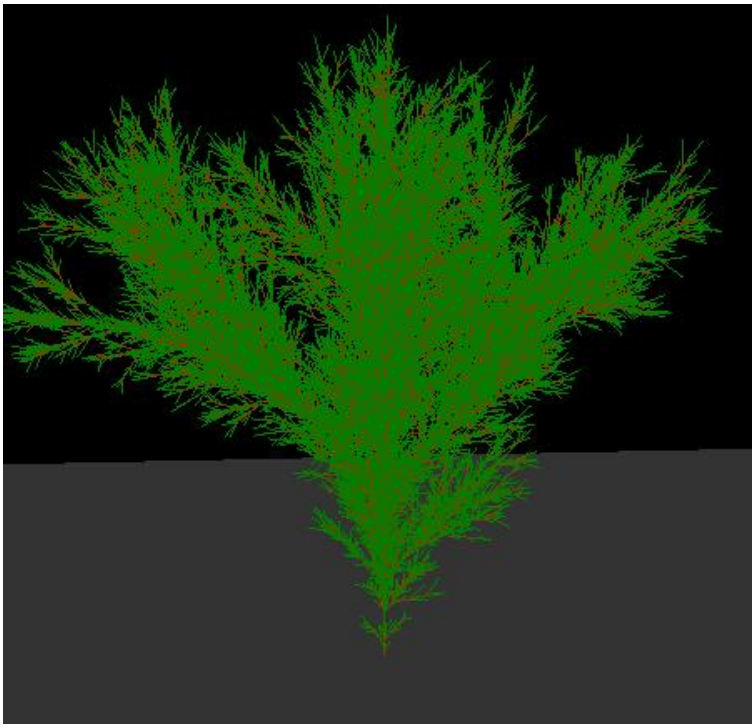
Courbe de Hilbert

Pour générer une structure arborescente, les caractères [et] ont été ajoutés à la grammaire de façon à pouvoir empiler et dépiler l'état (position, direction, couleur) de la « tortue » qui dessine la fractale.

Arbre fractal 2D

Ajout de la troisième dimension

La troisième dimension a été ajoutée en ajoutant simplement à l'alphabet du L-Système un symbole pour une rotation dans chaque dimension de l'espace à l'aide des matrices de rotations standard. Voici un exemple d'arbre généré à l'aide d'un L-System en trois dimensions :



Arbre fractal 3D, utilisant la pile d'états

Lors de la génération de l'image ci-dessus, un facteur aléatoire a été ajouté aux différentes caractéristiques de la fractales telles que la longueur des branches ou l'angle de rotation. Cela permet ainsi de mieux ressembler à la nature qui garde un caractère fortement aléatoire.

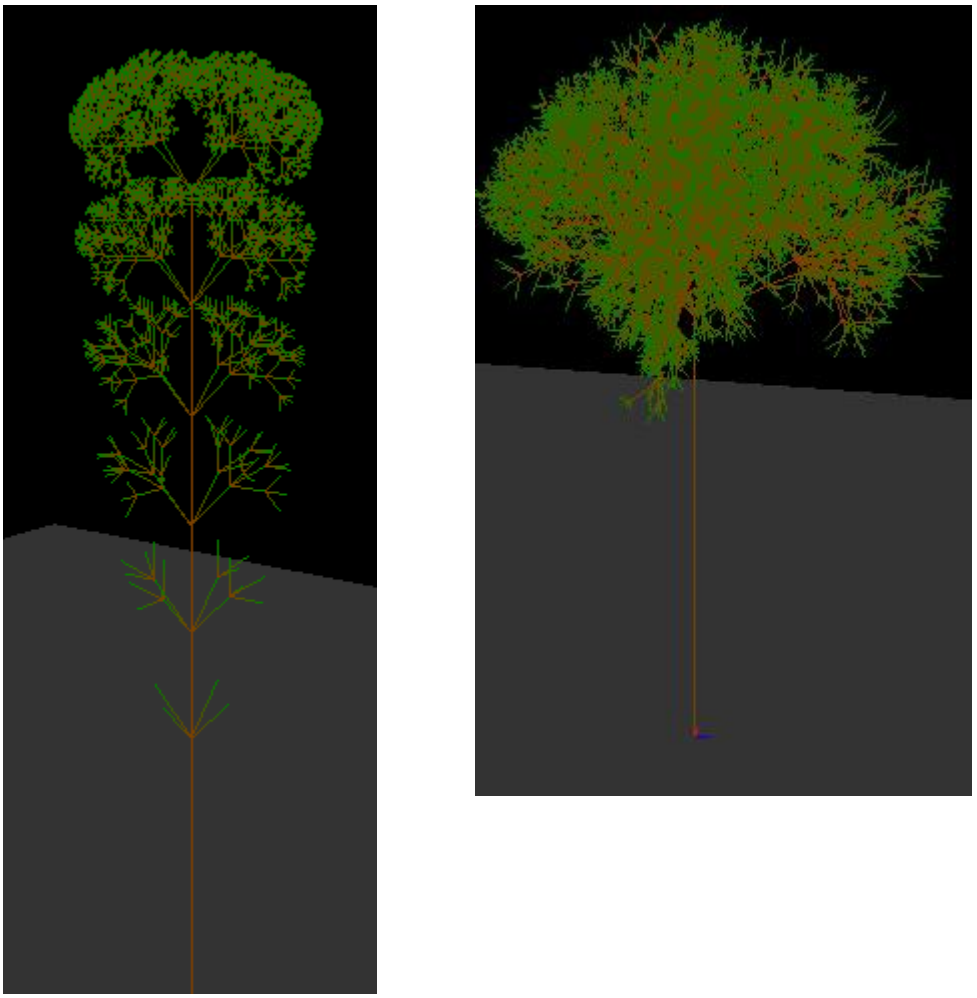
Multiplication des types d'arbres

Dans la suite du projet, le but était de trouver des L-Systèmes différents permettant de générer plusieurs types d'arbre différents. Pour cela, la notion de règles paramétrées a été ajoutée. Un exemple de règle paramétrée pourrait être :

$$F(x) \rightarrow F(x * 2) + F(x * 2)$$

Dans cet exemple, à chaque itération, la valeur de x qui correspond à la taille du déplacement F est doublée.

Cela a permis de générer de nouvelles possibilités de fractales plus ressemblantes à des arbres

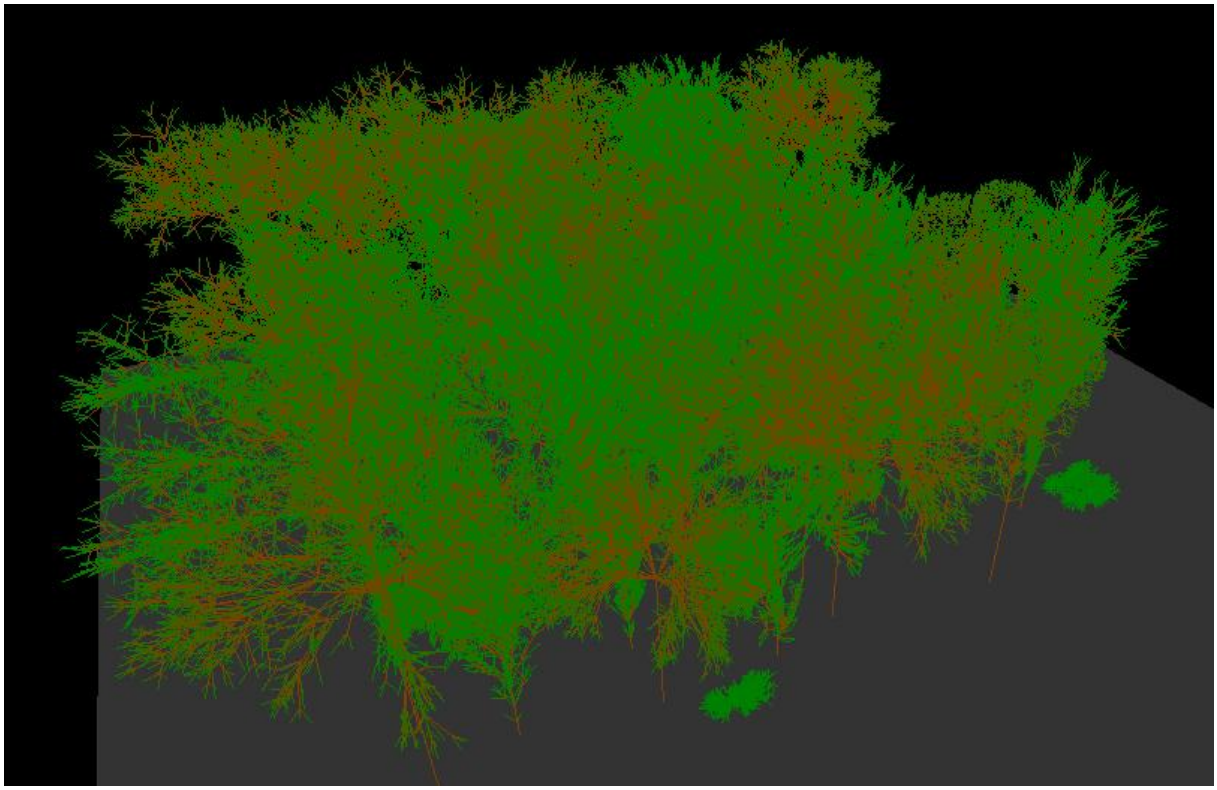


Arbres fractals 3D, utilisant les règles paramétriques

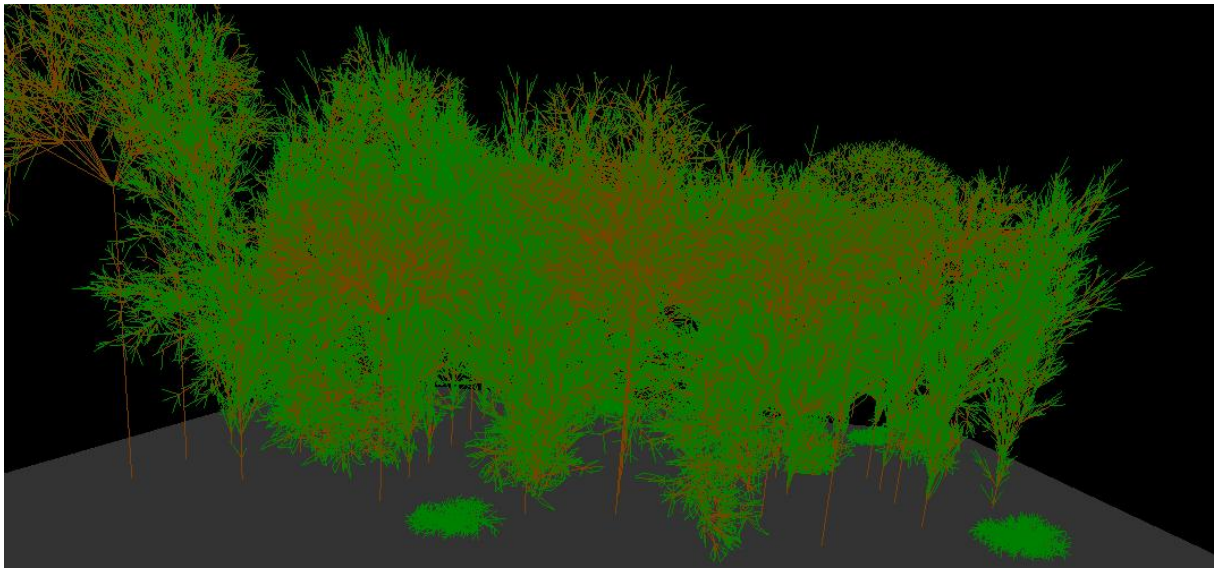
Génération d'une forêt

Grâce à l'ensemble d'arbres différents, et de leurs variantes possibles, générés par ces L-System, il a été envisagé de générer une forêt pour donner un aperçu du rendu visuel lors de la multiplication du nombre d'arbres à l'écran.

Pour cela, un nombre paramétrable d'arbres aléatoirement choisis parmi un ensemble de fractales disponibles est reparti sur une grille approximative. Voici un aperçu du résultat obtenu :



Forêt fractale, image 1



Forêt fractale, image 2

Perspectives d'évolution

Plusieurs pistes ont été explorées durant la réalisation de ce projet, pouvant permettre d'améliorer le rendu visuel.

De prime abord, question optimisation, la plus grande part du temps de génération est passée lors de la création de tableau de vertices à envoyer à la carte graphique. Il a été envisagé de paralléliser la chose, mais ce n'est pas possible en raison du caractère évidemment itératif de l'algorithme. Cependant, il serait intéressant lors de la génération de la forêt, en imaginant une forêt de plusieurs milliers d'arbres, de paralléliser directement sur la carte graphique (via openCL ou CUDA) le calcul d'une itération de l'algorithme de dessin pour chaque arbre dessiné. Ainsi, il serait possible d'obtenir un visuel beaucoup plus rapide et fluide de la génération de la forêt.

Deuxièmement, l'utilisation des « geometry shaders » serait intéressante. Cela permettrai en effet, sans augmenter la complexité des L-Systèmes, ni le temps de génération, de transformer les lignes brunes (les troncs) en cylindres et les lignes vertes en feuilles. Le résultat serait beaucoup plus réaliste, quoique moins détaillé, et nécessiterait beaucoup moins de points.

Enfin, des recherches pourraient être effectuées dans le sens des IFS (« Iterated Function System »), autre moyen de générer des plantes fractales, dont Przemyslaw Prusinkiewicz parle à la fin de son livre.

Annexes

Ce document est accompagné d'une série d'images réalisées à l'aide du programme développé dans le cadre de ce projet.