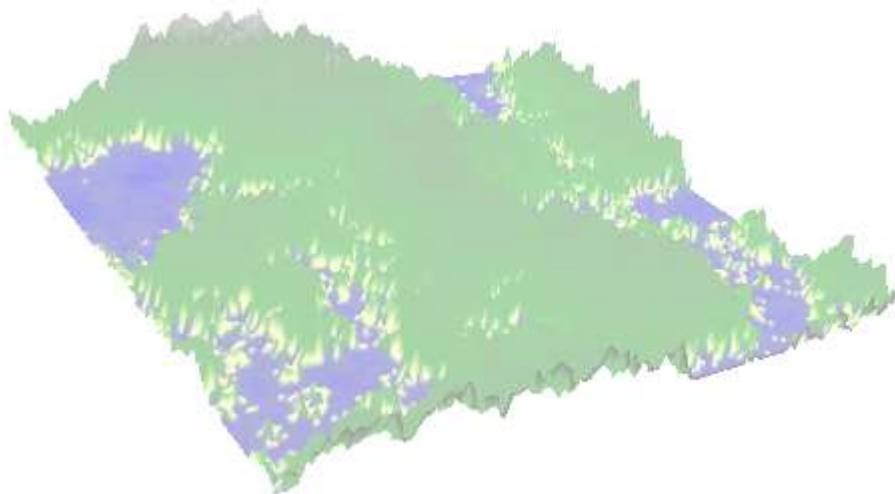


DANIEAU Fabien
TESSIER François
Groupe S4P2C

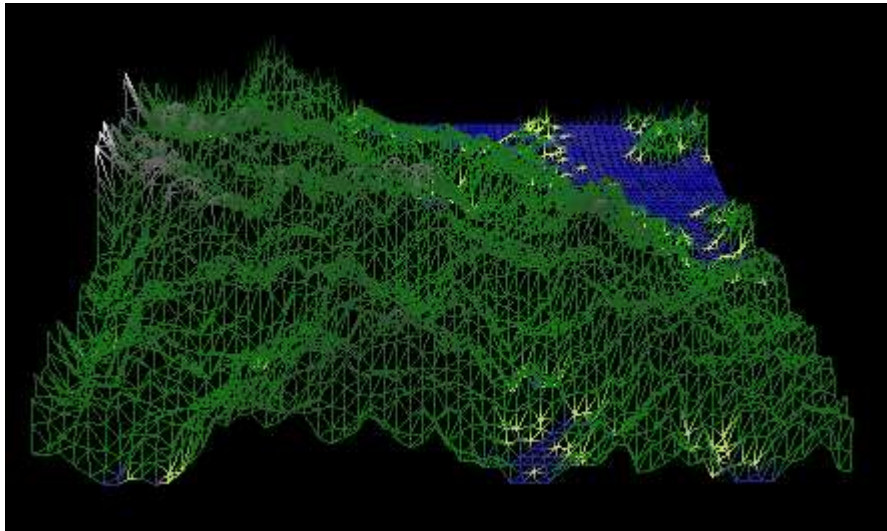
Génération de terrain en 3 dimensions

Projet tuteuré 2007



Introduction :

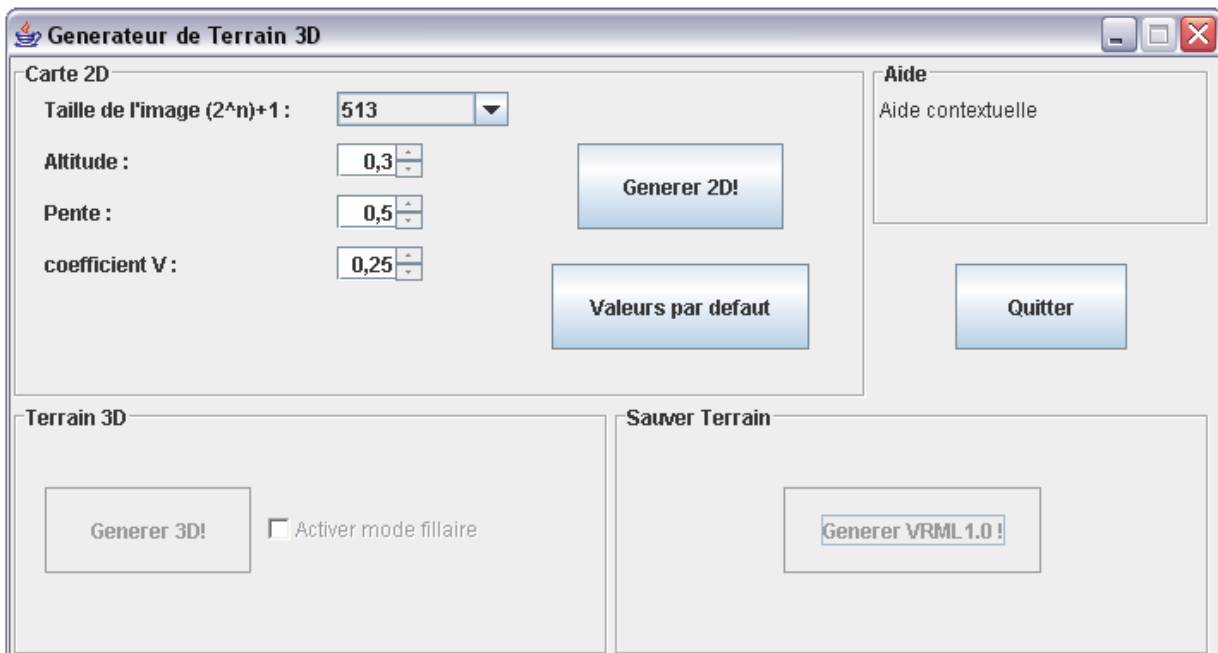
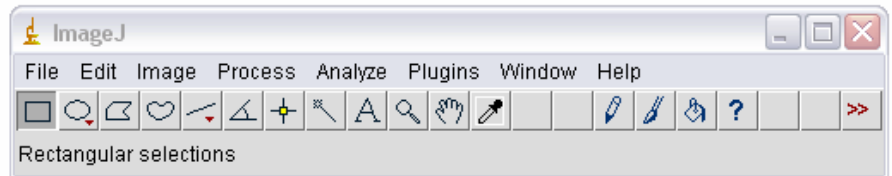
Le but de ce projet est de réaliser un outil pour générer un terrain en 3D et de le visualiser. Nous avons choisi ce projet pour plusieurs raisons. La première est notre intérêt pour l'imagerie numérique et particulièrement la 3D. La deuxième raison est tout simplement le sujet en lui-même qui nous a parut très intéressant dans le sens où nous partions de rien pour arriver à de la génération d'un terrain le plus réaliste possible. Enfin, nous affectionnons particulièrement le langage Java et avons pu ainsi découvrir une de ses composantes : Java3D.



Terrain filaire généré grâce au plugin réalisé

L'avant projet :

Avant de commencer concrètement ce projet, nous avons eu à choisir entre la création d'une application indépendante, ou la création d'un plugin pour ImageJ, un logiciel open source existant de dessin et de retouche d'image. C'est cette deuxième solution que nous avons décidé de mettre en œuvre. En effet, il nous a semblé que notre application seule ne serait pas d'une très grande utilité ou du moins serait limitée en termes de fonctions. Associée à un logiciel capable d'en modifier certains paramètres cet utilitaire prenait beaucoup plus de sens. De plus, l'API ImageJ est particulièrement bien faite et permet une intégration de plugin simple et propre. Dans le même esprit, nous expliquerons un peu plus loin notre implémentation de l'export en VRML afin d'utiliser le terrain en 3D généré.



Logiciel ImageJ et plugin « générateur de terrain 3D »

Le projet :

I – L'image 2D

La première étape dans la réalisation du projet a été la génération d'une image en 2D en niveau de gris. Afin de créer cette image, nous avons eu recours à un algorithme, appelé algorithme Diamant/Carré. Il nous a permis de générer une image aléatoire de ce type :

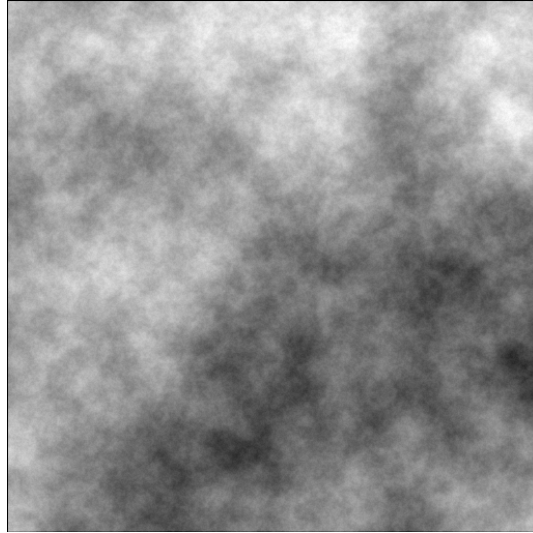
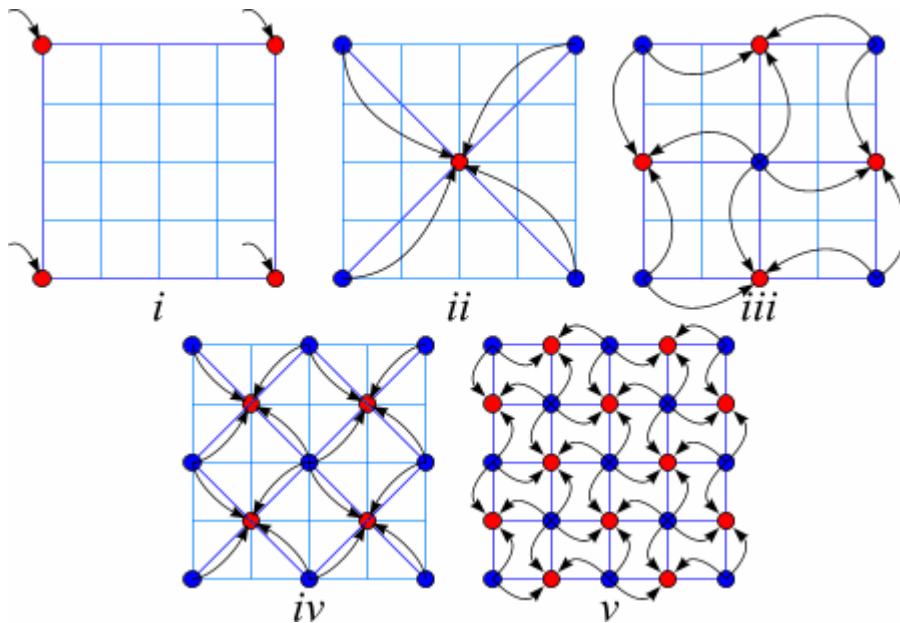


Image générée par l'algorithme Diamant/Carré

Typiquement, cette image servira de base à la génération en 3D. Les points les plus blancs correspondront à une plus grande altitude, et inversement, les points les plus noirs correspondront à une altitude plus basse. Voici un schéma du principe de cet algorithme :

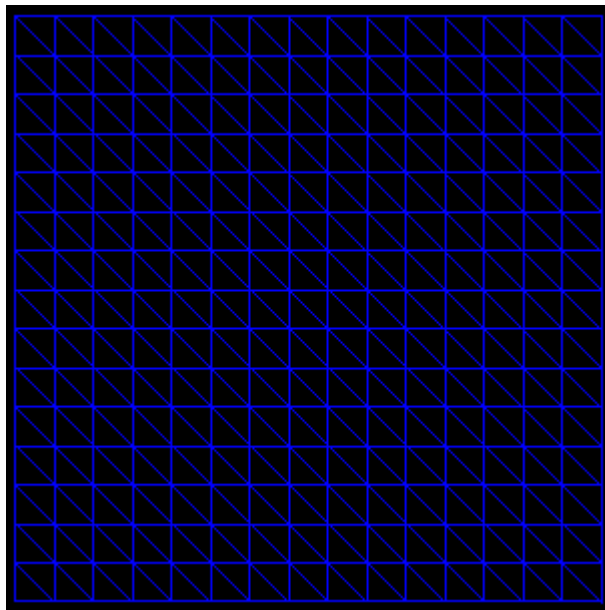


Principe de l'algorithme Diamant/Carré

Cet algorithme est basé sur des calculs de moyenne avec intervention d'une variable aléatoire. Dans l'exemple précédent, le point rouge central de la seconde image est une moyenne des valeurs des quatre points aux extrémités. On répète l'opération de façon itérative afin de calculer la valeur en tous les points de l'image.

II – La génération du terrain 3D

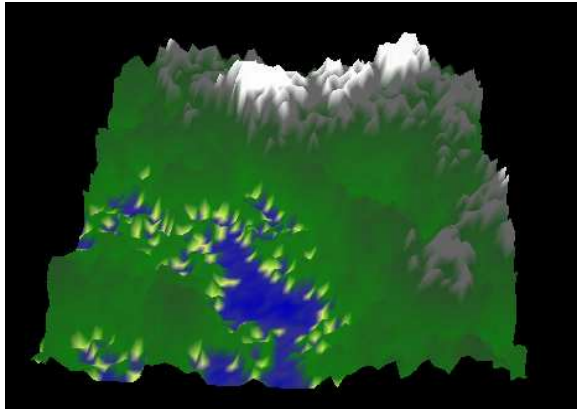
Une fois l'image en niveau de gris créée, on procède à sa « transformation » en terrain en 3D. Le principe est la construction de triangles dont chaque point a pour abscisse et pour coordonnée les coordonnées du point de l'image correspondant. La valeur du point de l'image correspond alors à l'altitude (axe des z) du point du triangle. On utilise pour ce faire un objet de type *TriangleStripArray*. Ces triangles forment une structure de ce type :



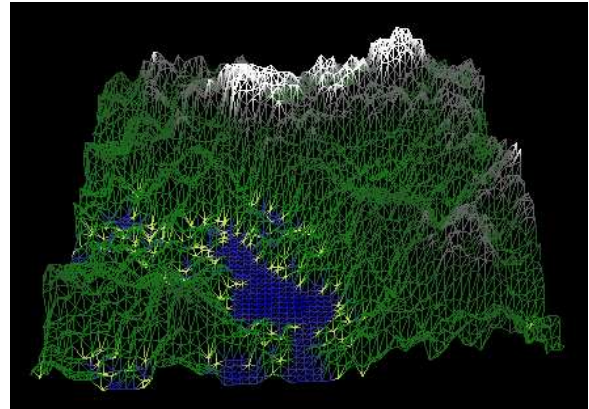
Structure des triangles créés

Dans le même temps, on colore le sommet en question en fonction de son altitude. Cette coloration se fait par pallier (suivant l'altitude) et est modifiée suivant une variable. Ainsi on obtient des couleurs plus dégradées. Enfin, toujours pour chaque triangle construit on crée une normal au triangle qui permet les effets de lumières une fois cette dernière appliquée.

Ces opérations terminées, nous obtenons ce type de terrain :



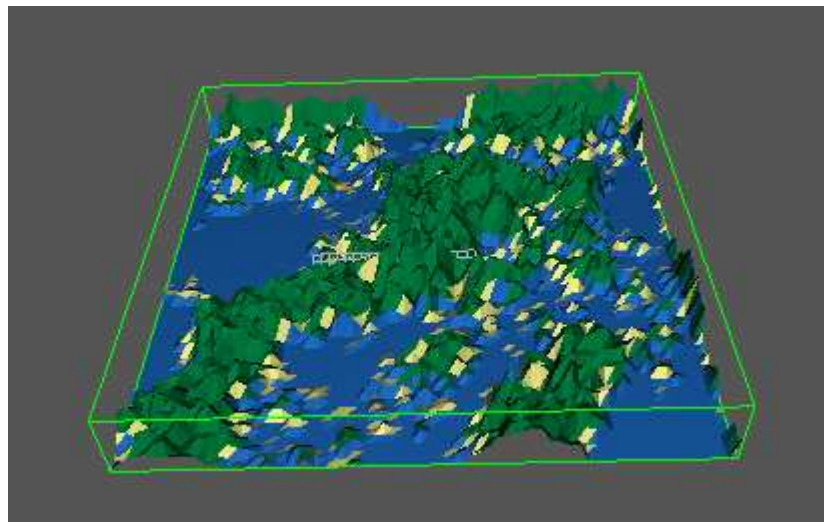
Terrain en 3D généré



Même terrain en mode filaire

III – L'export en VRML

Après plusieurs essais grâce à des logiciels de 3D comme AC3D et à l'analyse de fichiers, nous avons réussi à implémenter une sauvegarde en VRML. Cette sauvegarde nous a parut particulièrement importante dans le sens où elle donnait une finalité au projet. En effet, sans cette sauvegarde, l'image 3D générée était inutilisable. C'est le VRML 1.0 que nous avons utilisé étant donné sa compatibilité avec un plus grand nombre d'éditeur 3D (comme Blender ou AC3D) contrairement au VRML 2.0. De plus la documentation bien que faible restait malgré tout plus abondante pour cette version. L'export en .wrl nous a donné ce résultat :



*Export en *.wrl et affichage sous AC3D*

Conclusion :

Ce projet nous a beaucoup intéressés et dans ce sens, nous nous y sommes vraiment investis. Nous avons principalement axé notre logiciel vers l'utile grâce notamment à son intégration en tant que plugin d'ImageJ et l'exportation en VRML du terrain en 3D. Une amélioration pour l'avenir pourra être un rendu encore plus réaliste grâce à l'ajout de textures « réelles » et pourquoi pas d'autres types de fichiers d'export.
