ECOLE SUPERIEURE EN SCIENCES INFORMATIQUES

PROJET de 3ème année / DESS :

Octobre 2002 – Avril 2003

KartoMNT V2

Logiciel de Numérisation Altimétrique

BRESSON Jean et DERHI Roland

Réalisé à l'ESSI

Encadrement :
Fernandes Gilbert, Lingrand Diane

Résumé :

Le but du projet est de reprendre un logiciel existant nommé KartoMNT et de lui rajouter de nouvelles fonctionnalités. L'application existante permet d'analyser une carte géographique de manière à en extraire les lignes de niveaux, afin de réaliser un Modèle Numérique de Terrain (MNT) : une carte des altitudes en chaque point de la carte. Dans un futur proche, elle devra permettre, entre autre, de fusionner plusieurs MNT calculés précédemment, d'afficher ces MNT en 3D, de s'afficher dans la langue choisie par l'utilisateur,…

Sommaire

1. Introduction 3

2. Cahier des Charges 4

3. Description du travail réalisé 6

3.1. Etude du programme KartoMNT 6
3.2. Gestion des formats de MNT 7
3.3. Superposition des lignes de niveaux sur le MNT 8
3.3. Fusion de plusieurs MNT et travail sur les MNT 9
3.5. Visualisation 3D de MNT 11
3.5.1. Etude des techniques 3D en Java 11
3.5.2. Elaboration du visualisateur 12
3.5.3. Utilisation des LOD 12
3.5.4. Texturage du MNT 13
3.5.5. Captures d'image et de vidéos 13
3.5.6. Aperçu du résultat 14
3.6. Amélioration des méthodes de détection des lignes de niveaux 15
3.7. Correction de l'algorithme de génération de MNT 16
3.8. Internationalisation 17
3.9. Rendu du logiciel 17
3.9.1. Du point de vue de l'utilisateur 17
3.9.2. Point de vue des développeurs qui prendront notre succession 18

4. Planning du projet 19

5. Conclusion 20

6. Références 21

1. Introduction

Ce projet est proposé dans le cadre de la commission informatique du Comité Départemental de Spéléologie des Alpes Maritimes [2].

Il s'agit d'un logiciel permettant, à partir de cartes numérisées, de générer et exploiter des Modèles Numériques de Terrain (MNT).

Il existe des outils permettant de visualiser ces informations, comme *Dem3D* [4], ou encore *VTopo* [3] par exemple, mais si de tels logiciels sont disponibles gratuitement, en revanche se procurer un MNT est très coûteux. En effet, il n'existe pas, à notre connaissance, de logiciel gratuit permettant de réaliser ces MNT à partir de cartes numériques (telles que les cartes IGN). Des sociétés d'imagerie disposent de logiciels propriétaires effectuant ce travail, et elles les utilisent pour produire et vendre des MNT. Le but KartoMNT est de fournir au grand public ce type de logiciel.

La version 1.1 du programme est disponible depuis avril 2002. Elle a été réalisée par deux étudiants en 3ème année d'école d'ingénieurs à l'ESSI: Laurent Coustillac et Sylvestre Heit. Cette version permet déjà de générer des MNT à partir de cartes numérisées (calibration de la carte, détection des lignes de niveaux, calcul des altitudes, ...) Ce logiciel est développé sous la licence GNU GPL, et il est possible de le télécharger à l'adresse http://karto.free.fr/KartoMNT/index.html.

Les spéléologues, en particulier, sont très demandeurs de ce type de logiciel, car ils cherchent a visualiser en 3D les réseaux souterrains des grottes qu'ils explorent, ainsi que le relief du terrain situé au dessus de la grotte (afin de localiser d'autres entrées par exemple). Selon les statistiques du site de *Karto* [1] mentionné ci-dessus, le programme KartoMNT a été téléchargé 400 fois depuis avril 2002, et depuis le début du mois de mars, le programme est téléchargé environ 1 a 2 fois par jours. L'intérêt, et les attentes pour ce type de programme sont donc non négligeables.

Il s'agit donc dans ce projet de rajouter des fonctionnalités supplémentaires à KartoMNT afin de passer à la version 2.0 du logiciel. En particulier, les avancées à réaliser se situent au niveau de la manipulation des données altimétriques générées par KartoMNT : visualisation, modification, exportation, des données numériques de terrain.

2. Cahier des Charges

L'objectif du projet est de reprendre le programme KartoMNT version 1.0 existant, le comprendre, l'améliorer, le documenter.

Différentes fonctions doivent être développées pour faciliter le travail de l'utilisateur, ou pour améliorer les fonctionnalités du produit :

- **superposer les lignes de niveaux** détectées sur la carte des altitudes après calcul du MNT. Cela permettra de contrôler la "qualité" du MNT généré.

- permettre de **fusionner plusieurs MNT** (Modèle Numérique de Terrain) en :
  - lisant les différents MNT calculés précédemment,
  - contrôlant que la liste de tous les MNT lus forment bien un rectangle complet,
  - générant un nouveau MNT avec le même pas d'échantillonnage ou un pas différent.

- **afficher le MNT 3D**, permettant différents types de navigation (survol du terrain, ...):
  - déplacement du point d'observation de l'opérateur (translation, rotation, zoom),
  - mapping de surface avec l'image utilisée pour le calcul du MNT ou d'une autre image (déjà calibrée ou non),
  - possibilité d'enregistrer des vues 3D (de tout ou partie du relief: zoom) sous forme d'images,
  - étudier l'enregistrement de séquences vidéo du relief. Son développement sera effectué après accord de l'encadrement,
  - penser que, dans une extension future, il soit possible de "naviguer" sous-terre (topo 3D).

- Ajout de la possibilité pour un utilisateur d'écrire ses propres **classes de formatage** de MNT :
  - mise en place d'un paramétrage dans KartoMNT permettant d'indiquer le nom d'une classe Java pour un format de MNT ,
  - par l'implémentation (par l'utilisateur) d'une classe abstraite et l'ajout de sa classe dans KartoMNT.

- **Passage en multi-langues** du logiciel avec les outils d'internationalisation Java.

- **améliorer la détection** des courbes de niveau :
  - offrir un outil pour éliminer les points "parasites" détectés lors du seuillage (outils paramétrables),
  - étudier la possibilité de faire fusionner automatiquement les segments d'une même ligne (une même ligne est souvent discontinue lors du seuillage). Actuellement, le raboutement de tous les tronçons de lignes de niveau est manuel.

Nous avons, pour réaliser cela, quelques contraintes à respecter:

- Le logiciel est **écrit en Java**. Les librairies Java utilisées sont à la discrétion des développeurs (librairies 3D, traitement d'image, ...), mais elles doivent être libres de droit et diffusables gratuitement. Elles devront être fournies en fin de projet avec leur javadoc pour permettre la poursuite du projet par d'autres équipes.

- Le logiciel devra être **simple d'usage** car il n'est pas destiné à des informaticiens avertis, mais à des utilisateurs souvent novices en informatique, et parfois effrayés par les ordinateurs.

- Une **aide complète** devra être fournie. Le texte d'aide devra être découpé en 3 documents :
  - Le Tutorial : documentation simplifiée de l'utilisation de KartoMNT avec description de toutes les étapes de base
  - La documentation de référence : documentation complète avec le détail de toutes les fonctionnalités
  - La documentation d'installation : documentation décrivant la procédure d'installation de KartoMNT

  L'aide devra être disponible sous le format Word et HTML. Le document HTML étant disponible en ligne (dans le programme KartoMNT). Pour l'affichage de l'aide dans le programme, des librairies Java seront fournies. La documentation reprendra l'architecture de la documentation de Karto. Le logiciel et la documentation devront être fournis en français et en anglais.

- Le logiciel sera fourni avec un **programme d'installation** complet qui sera réalisé avec le logiciel freeware *Install Maker* de *Clickteam* [14].

- Le programme devra fonctionner (avec des **performances raisonnables**) sur des PC de milieu de gamme. Des tests de performance devront être réalisés sur différents matériels et OS, afin de fournir la grille des temps de calcul en fonction des configurations (mémoire, CPU, OS, taille des images, ...).

- **La stabilité du produit est prioritaire sur les fonctionnalités**. Il faut donc privilégier la correction de bugs sur l'ajout de fonctionnalités.

Une part importante du travail consiste donc à fournir un programme stable, avec un programme d'installation et une documentation utilisateur complète (documentation d'installation, tutorial, référentiel complet des fonctions disponibles). En résumé : réaliser le travail d'un ingénieur livrant "clé en main" un programme, en partant d'un existant.

3. Description du travail réalisé

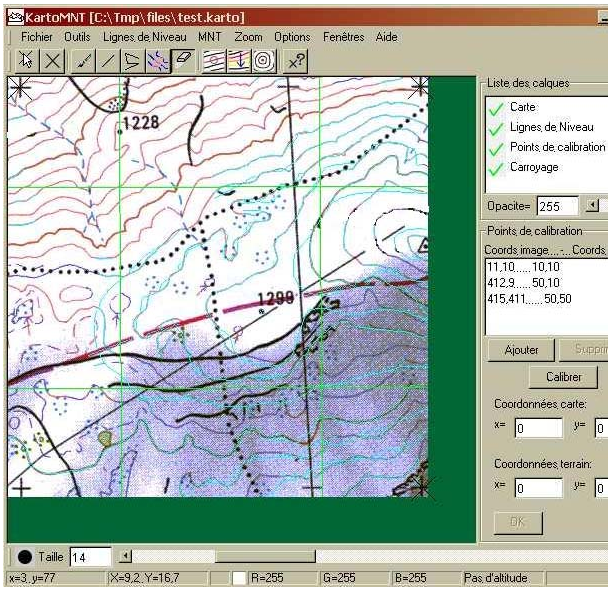
3.1. Etude du programme KartoMNT

Une première partie du travail a consisté à étudier le logiciel existant et ses fonctionnalités, déjà relativement importantes. Pour l'instant, le logiciel permettait de générer un Modèle Numérique de Terrain à partir d'une carte. Il a fallu étudier son fonctionnement du point de vue utilisateur, voir la manière dont s'effectuent les différentes opérations et comprendre, au moins dans les grandes lignes, le code qui se trouve derrière et sa structure, afin de localiser les classes et méthodes qui nous serviront par la suite.

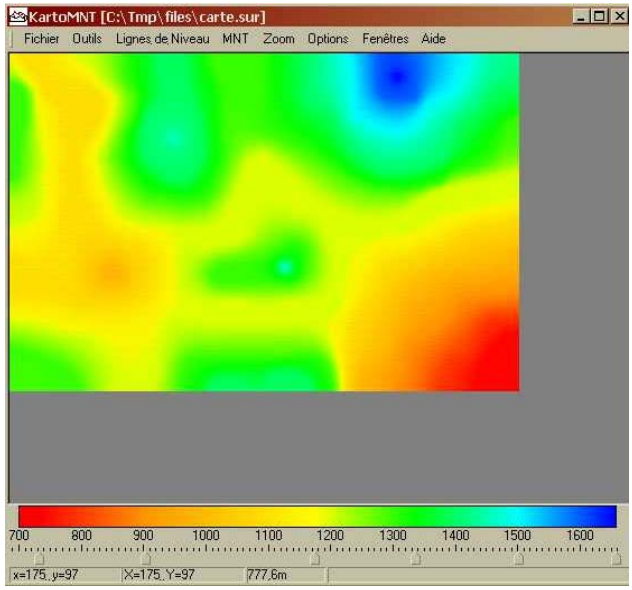
Il existe dans KartoMNT deux types de documents, qui correspondent aux types de fenêtres ouvertes dans l'interface:

- Les documents "karto", qui sont principalement constitués d'une carte, des lignes de niveaux détectées, du carroyage de calibration, tous ces composants étant visualisés avec des calques superposés. C'est sur celui-ci qu'est effectuée la calibration de la carte, la détection des lignes de niveaux et la détermination des altitudes qui permettent la génération d'un MNT.

- Les documents "MNT" qui sont générés une fois le travail sur le document "karto" effectué, et qui contiennent principalement un MNT, c'est à dire une carte des altitudes des points sur une zone rectangulaire donnée.



un document "Karto"



un document "MNT"

Ces deux types de documents sont traités différemment. Les actions proposées dans les menus ne sont souvent ouvertes qu'à un des deux. C'est principalement sur le Document MNT que nous aurons à travailler, autrement dit sur le MNT une fois qu'il a été généré.

Une classe (*FloatDataViewer*) permet de visualiser un tableau de données (*FloatData*) suivant une palette de couleur donnée. Cette classe est utilisée pour visualiser les lignes de niveaux (qui ont des couleurs différentes en fonction du fait que leur altitude ait été déterminée ou pas et, si elle l'a été, de sa valeur), et les MNT, où chaque point est dessiné d'une couleur, calculée à partir de son altitude.

Il est possible de réaliser un zoom sur cette représentation du MNT (avec ou sans interpolation) mais c'est à peu près tout ce que l'on pouvait faire avec un Document MNT dans la version 1 du logiciel.

En particulier, une des premières fonctionnalités que nous avons ajoutée a été la possibilité d'importer un MNT, et d'ouvrir un nouveau document MNT (jusqu'à présent un tel document était ouvert seulement lorsqu'un MNT était généré par un document de type "karto"). Il existait déjà la possibilité d'exporter les MNT gérés sous différents formats: ".mnt", ".sur", ou en VRML (".wrl").

Nous avons permis l'importation des formats .mnt et .sur. Par la suite (cela fait partie des spécifications du projets), nous avons fait en sorte de permettre aux utilisateurs de créer leurs propres classes de formatage de MNT.

3.2. Gestion des formats de MNT

Pouvoir importer et exporter de nombreux formats de MNT est un atout majeur du logiciel, qui le rendra utilisable et compatible avec les autres outils existants ou a venir. Nous avons donc géré les formats de MNT de manière générique en créant une classe abstraite *MntFormat*. Toute classe décrivant un format de MNT et héritant de celle-ci, à condition de répondre à quelques normes, sera reconnue par le logiciel et pourra être utilisée comme telle. Nous avons implémenté certains formats selon ce principe (*mnt*, *sur*, *vrml*, mais aussi, à titre d'exemple, le format *ter*, utilisé par le logiciel *Leveller*).

Le principe est de permettre à un utilisateur quelconque (certes quelque peu initié à la programmation Java) de rajouter une classe de formatage qui permettra au format de son choix d'être reconnu et utilisé par KartoMNT.

Nous proposons donc, dans la version installable du logiciel, un répertoire dans lequel l'utilisateur pourra déposer sa ou ses classe(s) (ou fichier java : il y a un petit exécutable de compilation). Celle-ci devra implémenter les méthodes spécifiées par l'interface *MntFormat*, c'est à dire être capable de fournir des données de calibration, les données du MNT, l'extension de fichier associée au format, les méthodes d'importation et d'exportations.

Nous avons créé à cet effet une petite aide fournissant à l'utilisateur la javadoc dont il a besoin ainsi que quelques exemples. Celle-ci a également été insérée dans le kit d'installation.

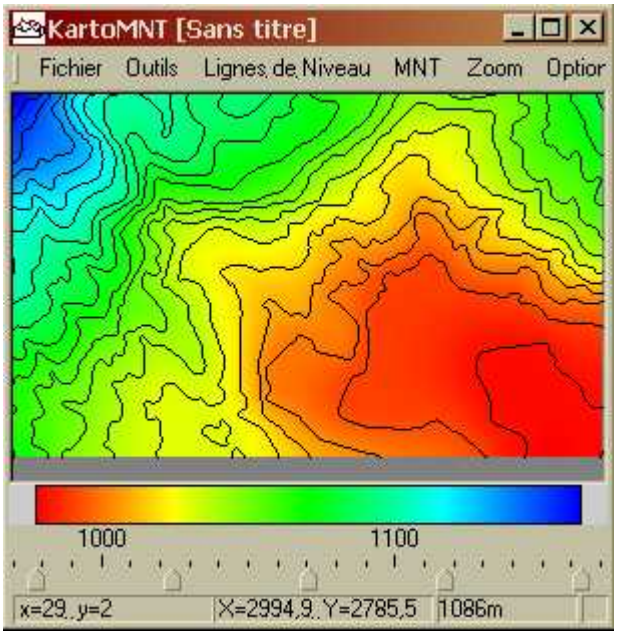
3.3. Superposition des lignes de niveaux sur le MNT

Pour permettre de contrôler la qualité d'un MNT généré, il était souhaitable de pouvoir y superposer les lignes de niveaux détectées qui ont permis sa génération. C'est ce que nous avons fait, en ajoutant au document MNT un attribut (*Lignes*), qui permet d'afficher des lignes de niveaux selon un modèle de couleurs simplifié (les lignes en noir et le reste transparent). Les données concernant les lignes de niveaux sur la zone du MNT généré sont donc fournies lors de la création du document MNT, transformées, et affichées si l'utilisateur le décide.

L'affichage, finalement, n'a pas été très facile, et a demandé une étude plus approfondie des systèmes d'affichage et de superposition des calques, afin d'utiliser les fonctionnalités déjà présentes. Une autre difficulté a été d'ajuster la taille de ce tableau de données que sont les lignes à la taille du MNT, qui peut varier en fonction du pas d'échantillonnage choisi (pour les lignes, il faut garder une certaine précision pour que leur affichage sur le MNT ait un sens). Nous avons à cette occasion pu corriger quelques défauts d'affichage de la version 1 du logiciel, qui ne prenait pas en compte correctement les cas où les pas d'échantillonnages étaient différents dans les deux dimensions.

Cette fonction de superposition, ne servant qu'à vérifier le MNT généré, elle n'est pas accessible dès lors que le MNT n'a pas été directement généré par le logiciel, c'est à dire lorsqu'il a été ouvert à partir d'un fichier par exemple (même si ce fichier a été généré par KartoMNT). En effet les données concernant les lignes de niveaux ne font pas parties des données d'un MNT et ne sont donc pas conservées lors de l'exportation.

Voici ce que l'on obtient en superposant les lignes de niveaux sur le MNT :



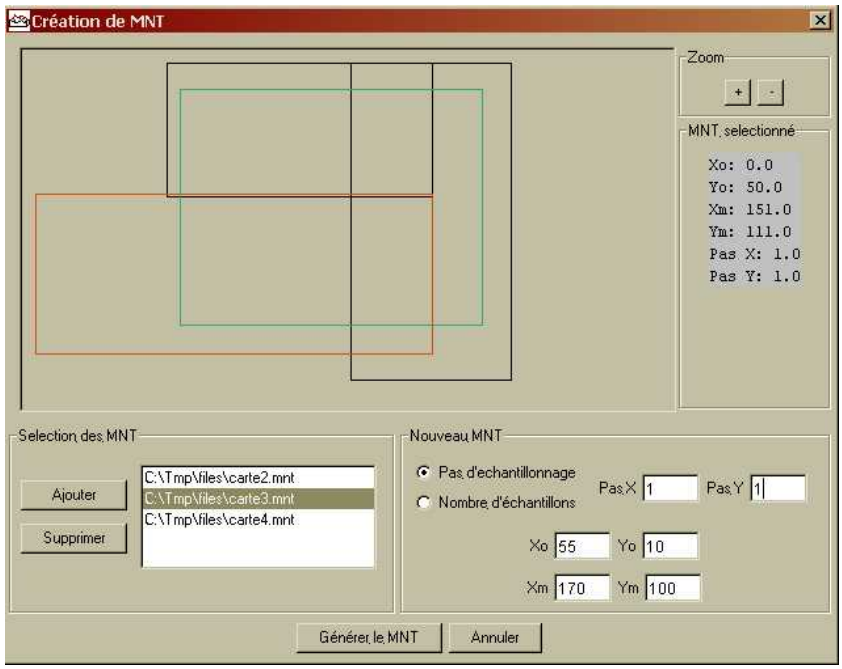
3.3. Fusion de plusieurs MNT et travail sur les MNT

Une des fonctions importantes à ajouter était la possibilité de fusionner plusieurs MNT en un seul. En effet, le travail pour la génération de MNT est un travail assez long et minutieux. Il est donc fort appréciable qu'un MNT à générer soit divisé en plusieurs parties moins importantes, pour plusieurs raisons :

- Le partage du travail entre différentes personnes,
- L'obtention d'un résultat au fur et à mesure de l'avancement du travail sur une grande carte,
- L'espace mémoire nécessaire pour travailler sur des cartes de bonne qualité peu s'avérer insuffisant (une carte entière à traiter impliquerait des coûts de calculs importants lors de la détection des lignes et surtout de l'interpolation pour la détermination des altitudes).
- Etant limité par le dispositif de numérisation (un scanner de taille normale ne permet pas de numériser toute une carte de randonnée), on peut souhaiter recoller plusieurs morceaux de différentes cartes, ou plusieurs parties de la même carte.

La fusion de MNT permettra donc de regrouper différents MNT créées à différents moments ou par différentes personnes afin d'obtenir des MNTs de taille plus conséquentes.

Nous avons donc créé un outil permettant la fusion de MNT :



Fenêtre de fusion de MNT

Sur la fenêtre ci dessus, trois MNT ont été importés afin d'être fusionnés. Les données de calibration qu'ils contiennent permettent de les situer les uns par rapport aux autres et de les positionner sous forme de rectangles dans le cadre supérieur gauche. Dans la liste du cadre inférieur gauche, l'un des MNT a été sélectionné; il apparaît en rouge dans le cadre du dessus et ses propriétés sont affichées dans le cadre de droite. Enfin, dans le cadre inférieur droit, on spécifie les dimensions, position, et pas d'échantillonnage du MNT que l'on souhaite générer. Celui-ci correspond au cadre vert sur le panel graphique.

Plusieurs questions se sont posées à propos de la fusion de MNT :

- La **sélection** des MNT :

Elle se fait à l'aide d'un *FileChooser* dans lequel on sélectionne les fichiers que l'on veut fusionner. Une liste permet de visualiser les fichiers sélectionnés, d'en ajouter ou d'en supprimer. Afin de contrôler les MNT sélectionnés, nous avons vu qu'un panneau sur le dialogue affiche leurs tailles et positions respectives sous forme de rectangles.

- La **dimension** : les MNT sélectionnés peuvent ne pas former un rectangle complet :

Par défaut, le MNT généré est de dimension maximale en fonction des MNT sélectionnés, et une valeur arbitraire (0) est donnée à l'altitude des points pour lesquels nous n'avons pas d'information. Ceci n'est évidemment pas la bonne solution si les MNT ne forment pas un rectangle parfait, à moins d'envisager de combler les trous par une autre fusion. C'est pourquoi il est possible de donner les coordonnées du MNT que l'on souhaite générer. On peut également découper un petit MNT dans un plus grand grâce à cette fonction.

- Le **pas d'échantillonnage**: il peut être différent en fonction des MNT sélectionnés, et l'utilisateur peut encore vouloir en choisir un différent pour le nouveau MNT.

Par défaut, les pas d'échantillonnage en X et en Y sont ceux du premier MNT de la liste des MNT sélectionnés, mais le dialogue offre la possibilité de les modifier. Cette fonction pourra donc permettre également, si un seul MNT est sélectionné, de le ré échantillonner.

- Le **calcul des altitudes** :

Si un pas et une dimension ont été déterminés pour le nouveau MNT, nous pouvons connaître la position de chacun des points qui le constituent. Une fois celle-ci déterminée, pour un point, on cherche dans quel MNT elle est comprise, afin de déterminer l'altitude à partir des données de ce MNT. L'altitude d'un point est calculée par interpolation à partir des altitudes des points environnants.

- **Les MNT peuvent se chevaucher**: quand nous cherchons à quel MNT appartient un point, il se peut que nous en trouvions plusieurs.

Dans ce cas nous considérons le premier MNT sur la liste pour faire le calcul. Il est possible aussi de faire une moyenne, mais théoriquement, quel que soit le MNT, la valeur devrait être la même.

Cet outil peut donc s'avérer très utile pour la fusion des MNT, comme nous l'avons vu précédemment, mais également pour la manipulation plus généralement, des données d'un MNT. Il permet en effet de sur échantillonner, sous échantillonner, redécouper, des MNT. Couplé à l'importation de classes de MNT formatées par l'utilisateur, qui permet l'importation et l'exportation de tous types de MNT, on obtient ainsi un outil très utile dans la manipulation des données altimétriques de terrain.

3.5. Visualisation 3D de MNT

3.5.1. Etude des techniques 3D en Java

Nous avons commencé, pour cette partie, par étudier toutes les manières d'afficher en 3D avec Java, et avons trouvé 3 façons différentes de procéder :

- Créer soit même un moteur 3d en Java.
- En récupérer un : nous avons étudié la possibilité d'utiliser *idx3d* [5] , un moteur entièrement en Java.
- Utiliser une librairie 3D Java. Il en existe de deux principales : Java3d [11][12][13], l'API 3D de Sun, et JGL [6] (ou équivalent comme GL4Java) qui est une interface pour l'utilisation directe de OpenGL en passant par Java.

Voici un tableau simple montrant les avantages et les inconvénients des différentes méthodes :

Méthode	Avantage(s)	Inconvénient(s)
Créer soit même un moteur 3d en Java	- 100 % portable	- Beaucoup de temps à investir - Performances non garanties
Récupérer un moteur 3d **- Idx3d**	- 100 % portable	- Nous avons jugé ces performances insuffisantes (code à l'appui)
Utiliser une librairie 3D Java:		
- JGL	- Librairie légère (1.2 M) - Performances potentiellement les meilleurs (plus bas niveau)	- Oblige à installer OpenGL - Ne fonctionne par conséquent que sous des machines compatibles OpenGL
- Java3d	- API de haut niveau - Nombreuses fonctionnalités intéressantes pour notre application (LOD)	- Librairie lourde (15 M) - Oblige à installer OpenGL ou DirectX - Ne fonctionne par conséquent que sous des machines compatibles OpenGL ou DirectX

Les performances dans KartoMNT v2 étant un critère important, nous avons au début préféré JGL. Seulement, Java3d possède des options fort intéressantes pour notre application, comme le Level of Details (*LOD: voir partie consacrée, page suivante*) qui nous a semblé indispensable pour l'optimisation de l'application, et qu'il aurait fallu autrement programmer. Aussi avons-nous choisi d'utiliser Java3d.

3.5.2. Elaboration du visualisateur

Nous avons cherché sur Internet si une application permettant d'afficher un terrain avec Java3d n'existait pas déjà. Nous en avons trouvé plusieurs mais une seule utilisait les L.O.D. (voir références : [7]). Cette application fonctionnant bien (à quelques erreurs près) et nous offrant un gain de temps considérable, nous avons décidé de nous baser sur elle pour le visualisateur de KartoMNT.

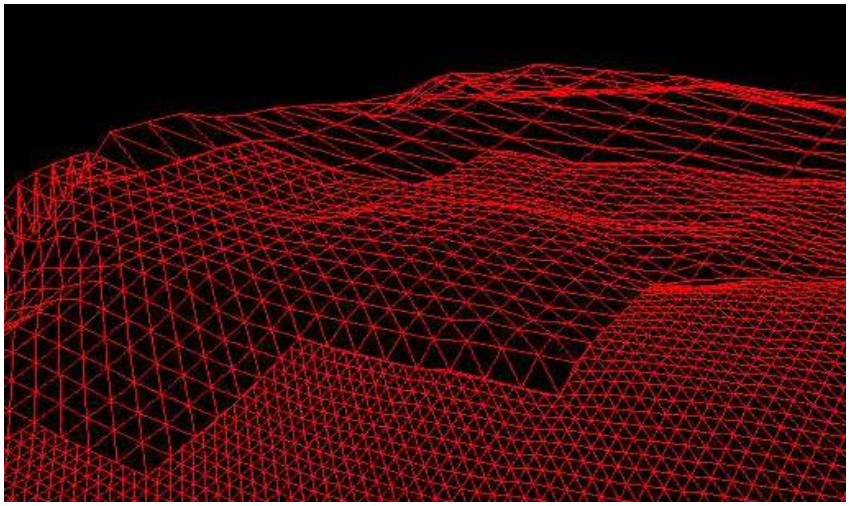
Nous avons donc cherché à comprendre son fonctionnement, puis avons fait en sorte qu'elle puisse lire des MNT (et non des images comme c'était le cas). Nous avons éliminé une limitation importante de cette application : celle-ci ne permettait d'afficher des MNT que de taille carrée et puissance de 2 (4x4, 8x8, 16x16,…). La méthode a donc été généralisée afin qu'elle puisse afficher n'importe quelle taille de MNT (ce qui a entraîné de nombreuses modifications dans le mécanisme de fractionnement du MNT et du L.O.D.)

3.5.3. Utilisation des LOD

Comme précisé dans le cahier des charges, l'application doit pouvoir fonctionner sur des ordinateurs de milieu de gamme. La vitesse d'affichage est donc primordiale dans KartoMNT v2. Sachant qu'un simple MNT (Modèle Numérique de Terrain) de 101x101 est composé de 20000 polygones (2 par carré), et que l'application doit être capable d'en afficher de bien plus grand, on comprend qu'afficher rapidement un MNT n'est pas chose aisée. C'est pour cela qu'il faut se servir d'un L.O.D. (Level of Detail).

Le L.O.D. permet de définir des niveaux de détail d'un objet en fonction de la distance qui les sépare de l'utilisateur. En effet, pour un objet lointain, il n'est pas forcément utile d'afficher tout ses détails. Cette technique permet de réduire le nombre de faces totales affichées dans une grande scène en 3D et donc accélère son affichage.

Remarquons un détail important. Pour appliquer un L.O.D. à un terrain en 3D, il faut préalablement le découper en plusieurs sous-parties et appliquer un L.O.D. différent pour chacune d'elles. En effet, si nous appliquons un L.O.D. au terrain en entier, c'est en son ensemble qu'il va se modifier, ce qui n'est pas le but recherché. Ce que l'on désire, c'est que les parties les plus lointaines du MNT s'affichent avec un niveau de qualité inférieure.



Mise en évidence de l'utilisation des LOD avec la représentation "fil de fer d'un terrain" : Les parties éloignées sont représentées avec une densité moins importante de facettes triangulaire

3.5.4. Texturage du MNT

Pour permettre un rendu 3D réaliste, il est nécessaire d'appliquer une texture sur le MNT. On peut choisir d'appliquer une texture quelconque (roche, terre, etc..), ou bien la carte qui a servi à générer le MNT, ou pourquoi pas une photo aérienne du terrain correspondant.

Etant donné les coordonnées des points du MNT, il suffit d'associer à chacun le point correspondant de la texture. Il faut donc affecter des coordonnées à la texture, et arranger les vecteurs normaux des polygones afin de tenir compte des modifications de la triangulation due aux L.O.D.

Cela devient plus compliqué lorsque l'on veut appliquer comme texture une carte ou une photo aérienne. En effet, pour que celle-ci corresponde à la topologie du MNT, elle doit être calibrée, c'est à dire contenir des informations sur sa localisation et son orientation. En utilisant les données de calibration du MNT, on peut alors appliquer la texture correctement.

KartoMNT permet, pour la génération de MNT à partir de cartes, de calibrer celles-ci dans un document "*Karto*". Nous avons donc utilisé ce format *karto* pour permettre à l'utilisateur de texturer son MNT à l'aide d'une image (photo ou carte) calibrée avec KartoMNT (en particulier, il peut utiliser le fichier utilisé pour la création du MNT tel quel). Nous avons prévu le cas où la texture serait plus petite que le MNT visualisé (le cas inverse ne posant pas de problème puisqu'il suffit de découper l'image) en proposant, le cas échéant, à l'utilisateur de choisir une texture "bouche-trous" qui sera appliqué là où cela s'avèrera nécessaire.

Exemple de MNT :le plateau de Cavillor,
sur lequel nous avons appliqué la carte calibrée de ce même plateau

3.5.5. Captures d'image et de vidéos

Nous avons également recherché un moyen de réaliser des captures d'écran en Java, afin d'extraire des prises de vue, voire des petites séquences animées, de la visualisation. Nous l'avons trouvé en faisant appel à des classes spécifiques de Java3D qui détourne l'affichage à l'écran pour écrire dans un *BufferedImag,. BufferedImage* qu'il fallait encoder en un format d'images. Nous avons donc permis la sauvegarde au format JPEG et, de plus, nous avons trouvé une classe permettant d'encoder des Gifs Animés [8]. Ce qui nous a donc permis, en plus de pouvoir réaliser des captures d'écran, de réaliser des captures de séquences d'images et de "film" de gifs animés. Nous avons également ajouté la possibilité de réaliser des captures d'écran haute résolution (en vue d'impressions au format A4, par exemple).

3.5.6. Aperçu du résultat

Lorsque l'application démarre, une fenêtre de dialogue apparaît permettant à l'utilisateur de choisir différents paramètres : le MNT à afficher bien sûr, mais aussi s'il veut appliquer une texture au MNT, et si oui, laquelle , s'il veut appliquer une texture de background et si oui, laquelle, et si il souhaite laisser KartoMNT choisir les paramètres de visualisation de manière automatique ou les choisir manuellement (mise à l'échelle, niveaux de L.O.D, …).

Le visualisateur crée alors le modèle et l'affiche.

Le visualisateur 3D KartoMNT

Sur l'image ci-dessus, on voit le panel 3D sur lequel est affiché le MNT en 3D. On peut remarquer le petit repère en bas à gauche qui permet de se repérer dans l'espace.

Sur la droite se trouvent les différents boutons correspondant aux possibilités qui s'offrent à l'utilisateur. On trouve en bas les commandes principales qui , complétées par l'utilisation de la souris, permettent d'effectuer les déplacement dans le monde 3D. Parmi celles-ci, on notera la possibilité de se remettre à l'horizontale, de se recentrer sur le MNT, ou de tourner autour de lui. Au dessus, des champs de textes permettent de saisir manuellement une position et une direction d'observation.

Les autres onglets, que l'on ne voit pas sur l'image, permettent de modifier les vitesses de déplacement (onglet *Vitesses)* et de choisir des caractéristiques de représentations du MNT, comme l'affichage texturé ou fil de fer, et la résolution du modèle (onglet *MNT*).

Enfin, en haut, les différents outils de saisie sont disponibles (capture d'écran, ou de séquences d'images).

Le visualisateur a été réalisé indépendamment du logiciel. Il a donc du être intégré par la suite au reste du programme.

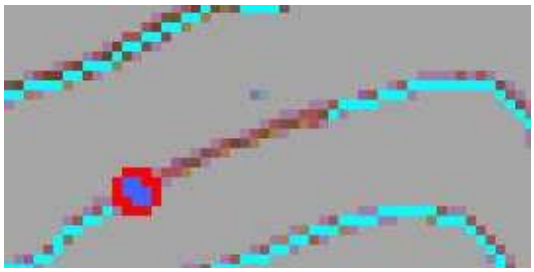
3.6. Amélioration des méthodes de détection des lignes de niveaux

La détection des lignes de niveaux est une partie délicate du travail en aval de la génération du MNT. Elle ne fait pas exactement l'objet de notre projet, puisqu'elle a été réalisée par nos prédécesseurs sur le projet KartoMNTv1, mais nous en avons amélioré certains aspects afin de rendre cette tache plus agréable à l'utilisateur.

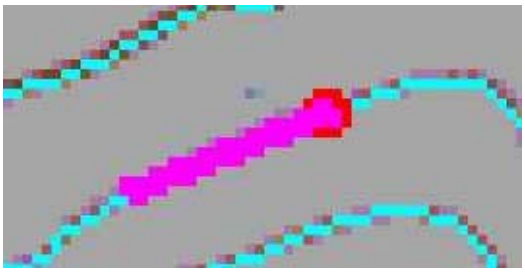
Il faut savoir que la détection des lignes sur la carte se fait à partir d'une plage de couleur définie par l'utilisateur. Cette méthode ne peut évidemment pas détecter parfaitement les lignes dès le premier jet (la carte a été numérisée, les lignes peuvent être plus ou moins foncées, voire de différentes couleurs, etc..) Nous avons donc dans un premier temps permis à l'utilisateur de réaliser plusieurs détections superposées avec différentes couleurs de référence, et de réinitialiser le tout si il le souhaite (auparavant, on ne pouvait effectuer qu'une détection puisque le précédente était écrasée à chaque fois).

Les lignes détectées doivent ensuite être travaillées afin de pouvoir être utilisées dans le calcul des altitudes. Il faut corriger les erreurs de détection à l'aide de différents outils : gomme, crayon, etc.. C'est un travail minutieux et relativement fastidieux. A ces outils, nous avons ajouté un utilitaire permettant le raboutage semi-automatique des lignes. Lors de la correction des lignes, la majorité du travail consiste à recoller les partie des lignes oubliées par la détection. Ce nouvel outil permet de sélectionner une ligne, dont il détecte l'extrémité, et de se diriger avec la souris vers une autre ligne, dont il détecte également l'extrémité pour la joindre à la première. Ceci facilite donc grandement le travail de l'utilisateur.

Voici un exemple d'utilisation (les lignes de niveau détectées sont en bleu clair) :



Détection de l'extrémité d'une ligne pointée par le curseur.

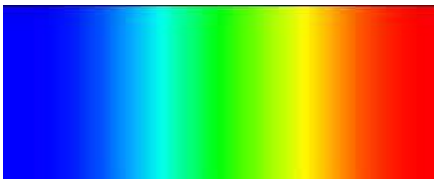


Détection de l'extrémité de l'autre ligne et raboutage.

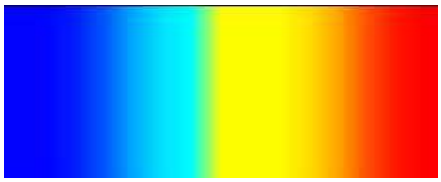
3.7. Correction de l'algorithme de génération de MNT

Nous avons pu, après la création des méthodes de fusion de MNT, mettre en évidence un défaut important dans l'algorithme d'interpolation qui génère le MNT à partir des lignes de niveaux du document Karto. En effet, il apparaissait nettement sur la carte altimétrique générée, ainsi que sur le modèle 3D du MNT, une marche au niveau de la jonction entre deux MNT.

Voici le type de résultat obtenu, en recollant deux morceaux issus d'une même carte d'origine (nous avons créé un MNT artificiel de pente régulière pour mettre en évidence ce défaut) :



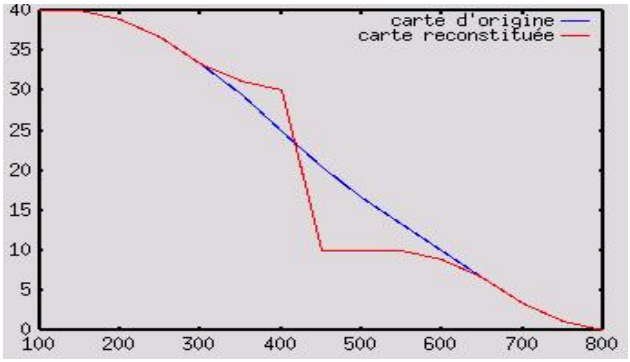
MNT obtenu à partir de la carte d'origine



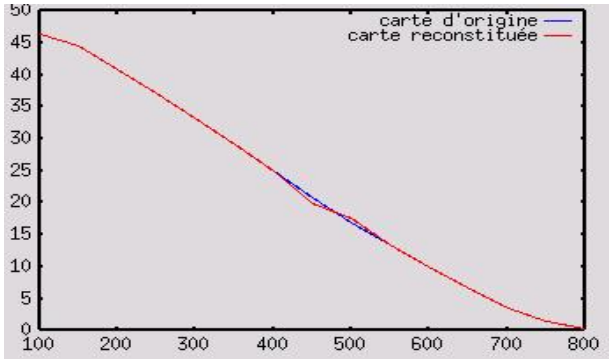
MNT reconstitué par fusion de deux parties générées sur la même carte d'origine

On voit bien la démarcation entre les deux parties sur la carte de droite. Une variation brutale de couleur correspond en effet à une variation brutale d'altitude. Ce problème est observé sur les bords des MNT. Il est du au fait que le calcul de l'altitude d'un point à partir des lignes de niveaux se fait à partir des points alentours de celui-ci, et ne prend pas toujours en compte les valeurs à l'extérieur de la région sélectionnée pour le MNT. Ainsi, sur le bas d'une pente descendante, par exemple, seules les valeurs supérieures des points compris dans la région sont prises en compte pour l'interpolation, ce qui crée une diminution de la pente. Ce phénomène est valable pour tous les bords mais n'avais pas été mis à jour auparavant.

Nous avons donc corrigé cela en reprenant la méthode d'interpolation des lignes de niveaux. Voici une courbe montrant le profil de notre pente régulière avant et après notre intervention :



Profil de la pente reconstituée avec l'ancienne methode



Profil de la pente reconstituée avec la nouvelle methode

On voit que ce défaut est présent au niveau de la jonction mais également au niveau des autres bords (la pente est censée être régulière, même aux extrémités). Toutefois, il n'est à maintenant localisé que sur le dernier point, et très faible (certainement encore corrigible), ce qui le rend invisible sur la visualisation en couleurs ou en 3D.

Nouveau MNT reconstitué :

La rupture entre les deux parties est à présent invisible. On voit aussi que la pente est moins aplatie sur les bords.

Il est conseillé de laisser une marge autour du MNT généré afin de limiter cet effet, inévitable lorsque l'on arrive sur les bords réels de la carte.

3.8. Internationalisation

Pour internationaliser le logiciel, nous avons utilisé les outils proposés par Java, à savoir l'utilisation des classes *Locale*, et *ResourceBundle*. Dans le code, nous avons géré tous les messages, labels, et autres informations textuelles présentées, à l'aide d'une classe *ResourceManager*, qui utilise ces outils en se référant à un fichier de type *properties*, sorte de dictionnaire contenant la traduction de ces textes dans une langue donnée. Ce ficher doit avoir un nom particulier : *kartoMNTresources_XX.properties*, où *XX* est un identifiant de la *Locale* (le langage) souhaitée. Dans un fichier d'initialisation du logiciel, le langage est spécifié par ce même identifiant *XX*. Il est donc possible très facilement de rajouter une langue quelconque en copiant, traduisant et renommant le fichier *properties* comme il se doit, puis en spécifiant la langue dans le fichier d'initialisation.

Nous avons dores et déjà traduit le logiciel en deux langues (anglais et espagnol), et il a été fait en sorte dans le kit d'installation de permettre aux utilisateurs de rajouter les langues qu'il souhaitent à moindre coût (autre que celui de traduire le fichier en question…)

3.9. Rendu du logiciel

La finalisation du logiciel est un aspect qui est ressorti en priorité dans les spécifications du projet. C'est donc une chose à laquelle nous avons essayé de porter beaucoup d'attention. En particulier, deux aspects sont à retenir :

3.9.1. Du point de vue de l'utilisateur

- Il faut faciliter au maximum le **lancement du programme**, en considérant que nos utilisateurs ne sont pas forcement experts en informatique.

Il doit être possible, par un simple clic sur une icône (sous *Windows* en tous cas), de démarrer le programme. Dans cet objectif, nous avons préparé deux "kits" d'installation pour chacune des plates formes *Linux* et *Windows*. L'un d'eux nécessite l'installation préalable de Java sur la machine, et éventuellement une petite configuration pour spécifier sa localisation. L'autre en revanche est totalement indépendant puisqu'il contient un *Java Runtime Environment* integré (mais il est beaucoup plus lourd). Nous avons étudié la possibilité de convertir notre programme java en un exécutable *Windows* (ce qui permettrait de lui associer un icône KartoMNT, que nous avons dessinée), à l'aide du programme *JavaExe*. Mais celui-ci ne nous permet pas de spécifier toutes les options que nous souhaiterions dans la commande java, en particulier pour l'utilisation des pilotes Java3D. Ainsi cela nécessiterait une configuration préalable à activer par l'utilisateur, ce qui est ce que nous cherchons à éviter. Pour l'instant donc, le programme démarre à partir d'un fichier *.bat* sous *Windows*, et d'un *.sh* sous *Linux*.

- Rendre un **programme installable**, c'est à dire un seul exécutable qui installe ce qu'il faut sur le machine de l'utilisateur.

Nous avons utilisé le logiciel *InstallMaker* [14], qui permet d'offrir à l'utilisateur un guide d'installation classique (informations, choix du lieu d'installation,…)

- Fournir une **documentation d'utilisation** claire et complète.

Nous nous sommes appuyé sur la documentation fournie avec KartoMNT version 1, que nous avons complétée, et à laquelle nous avons ajouté l'aide concernant les fonctionnalités nouvelles de la version 2. Cette aide est fournie dans le kit d'installation et est consultable directement à partir du programme, dans lequel nous avons mis un petit menu d'aide.

- Fournir une interface claire pour **l'ajout de nouvelles classes de formats**.

Comme nous l'avons expliqué dans ce rapport, l'utilisateur peut ajouter dans un répertoire les classes qu'il a créé. Nous lui proposons un petit exécutable de compilation (mais il peut avoir compilé lui-même ses classes), et un répertoire dans lequel il doit les déposer. A partir de là, sans aucune manipulation, ces classes sont reconnues en tant que format supporté par le logiciel. Afin de simplifier cette opération, une petite aide spécialement dédiée est également présente.

- Permettre d'obtenir les informations nécessaires et le logiciel en téléchargement à partir d'un **site web**.

Nous avons créé un nouveau site web KartoMNT, dans lequel nous présentons le logiciel, ses fonctionnalités, son historique, et mettons à disposition les différentes documentations, et bien sur le logiciel en téléchargement.

Une page du site web KartoMNT

3.9.2. Point de vue des développeurs qui prendront notre succession

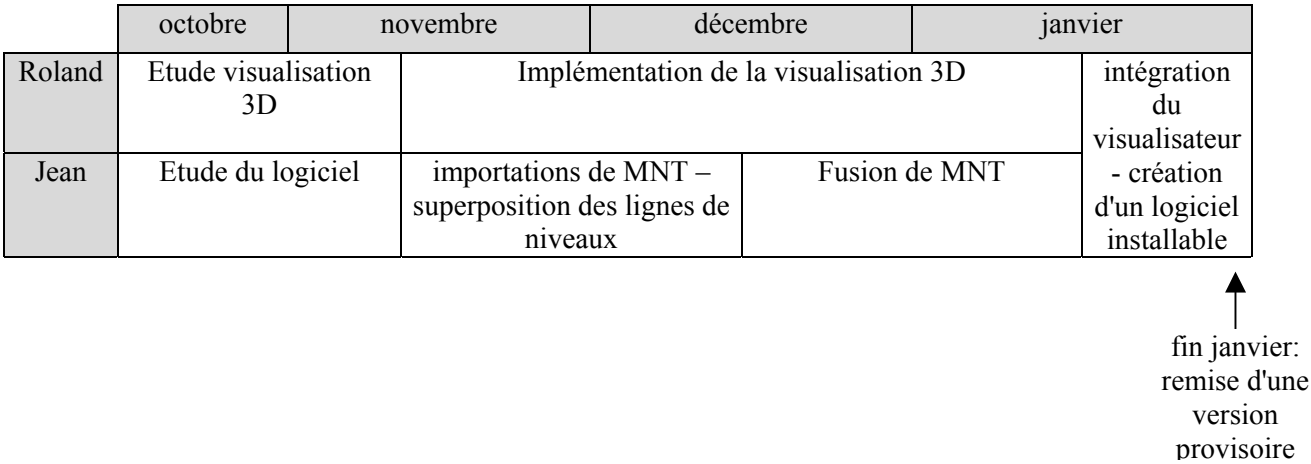
Afin de faciliter la tâche de nos futurs successeurs, nous avons bien sûr essayé de fournir un code java le plus clair possible et commenté. De plus nous avons réalisé une documentation technique, essayant de couvrir au mieux les différentes fonctionnalités du logiciel (y compris, dans la mesure du possible, celles qui avaient été réalisées auparavant), ainsi que sa structure générale, afin de permettre à toute personne désirant améliorer, compléter, ou corriger le code source de l'application puisse s'y retrouver rapidement (ce qui ne dispense pas une étude attentive du code…)

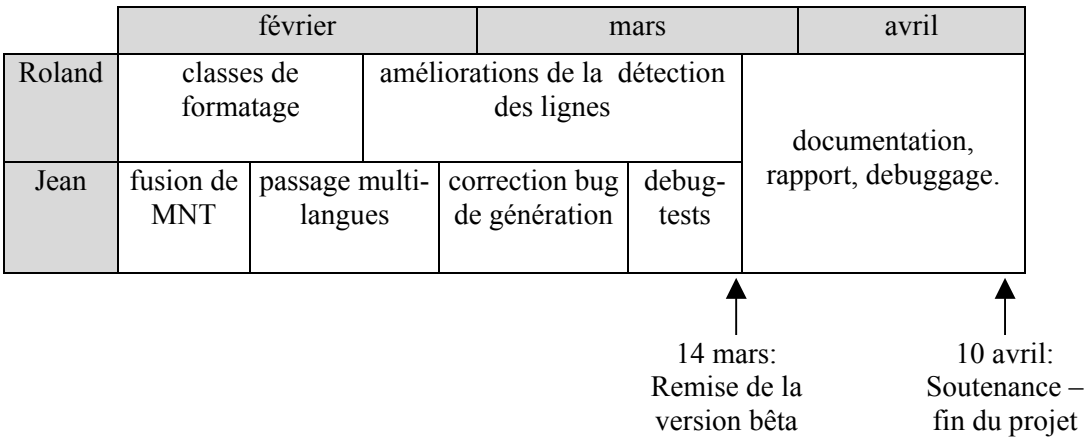
Cette documentation sera disponible également, avec les sources de l'application, sur le site web de KartoMNT.

4. Planning du projet

Comme nous l'avons vu, la plupart des objectifs spécifiés dans le cahier des charges ont été en majorité atteints. Nous avons pu respecter un planning convenable, même si parfois certaines parties se sont avérées plus ou moins longues que prévues, voire se sont ajoutées au planning prévisionnel (par exemple avec la correction de bugs de la précédente version).

Voici donc, dans ses grandes lignes, la manière dont a été réparti le travail :





En règle générale, nous avons su nous conformer aux limites fixées, par nos encadreurs pour la présentation des avancements successifs ou par nous mêmes dans les plannings prévisionnels.

Le travail en binôme a été efficace puisque nous nous sommes réparti les taches de manière à pouvoir travailler indépendamment sur différentes parties tout en maintenant une cohésion dans le travail en nous concertant régulièrement.

5. Conclusion

Ce projet nous a semblé intéressant par les techniques auxquelles il a fait appel. Celles-ci ont pu être liées à l'image, pour ce qui est de la manipulation des lignes de niveaux, à la synthèse particulièrement puisqu'une partie importante du travail a été centré sur la visualisation 3D, ou d'ordre plus général, avec l'utilisation de Java pour la manipulation de données et de formats sur les MNT par exemple.

Il a été également passionnant par son domaine d'application, le type d'utilisateurs qu'il concerne, et par les différentes étapes de la création d'un logiciel dans lesquelles nous sommes intervenus. En plus de ce que nous aurons pu apprendre en terme technique, de programmation, de synthèse 3D, d'interface, de traitement d'image, cet aspect important d'ergonomie et d'utilisabilité d'un produit, s'adressant à un public à priori non averti en informatique, nous a été très enrichissant. Enfin, le fait de remettre un produit fini, avec aide et installation, nous sera certainement tout aussi utile pour des projets à venir.

L'évolution par rapport à la version 1 de KartoMNT est aujourd'hui bien nette. Toutefois, ce logiciel n'a pas la prétention d'être abouti et il serait envisageable d'ajouter encore de nouvelles fonctionnalité, dans la création de MNT, avec des techniques de détection et de correction de lignes de niveaux plus avancées et plus précises, limitant au maximum les interventions de l'utilisateur, ou dans le visualisation des MNT, où l'on pourrait envisager de tracer et de suivre des "promenades" sur le terrain virtuel modélisé, ou bien des insertions de modèles de cavités pour la navigation souterraine.

6. Références

[1]http://karto.free.fr	Le site web de KartoMNT version 1, et des autres produits de la famille *Karto*
[2]http://cds06.free.fr/commissions/com-info/com-info.html	Commission informatique du Comité Départemental de Spéléologie des Alpes Maritimes
[3]http://vtopo.free.fr	Visual Topo, logiciel de topographie spéléologique
[4]http://craterlake.wr.usgs.gov/dem3d.html	Dem3D, visualisateur de données cartographiques 3D
[5]http://www2.active.ch/~proxima/idx3d/idx3d.html	idx3D : un moteur 3d en java
[6]http://nis-lab.is.s.u-tokyo.ac.jp/~robin/jGL/	JGL, interface pour l'utilisation d'OpenGL en passant par Java.
[7]http://www.hut.fi/~hsalonen/java3d/	L'application en Java3D que nous avons utilisé à l'origine du visualisateur3D
[8]http://www.fmsware.com/stuff/gif.zip	Des classes permettant l'encodage des gifs animés
[9]http://java.sun.com	Le portail web de Sun pour la distribution de Java.
[10]http://java.sun.com/products/java-media/3D/	L'API java3D pour Windows distribuée par Sun.
[11]ftp://ftp.oleane.net/pub/java-linux/java3d/1.3/i386/fcs/	Une adresse pour le téléchargement de Java3D pour Linux.
[12]http://www.sccs.chukyo-u.ac.jp/~miyasaki/manual/java3d-1_3-doc/html/index.html	La javadoc complète de l'API Java3D 1.3.
[13]http://jamaica.ee.pitt.edu/Eric/java3d/rotation.htm	Pour l'obtention des angles d'Euler à partir d'un *Transform3D* en Java3D.
[14]http://www.clickteam.com/French/installmaker.php	Install Maker, Outil de distribution et d'installation utilisé par KartoMNT