Abou Jaoudé Yann
700324
ESILV TDB
A2 S2

Projet informatique
langage C#

Traitement d'Images Bitmap

I) Mais où est la class MyPixel ?

J'ai choisi de ne pas faire de Class Pixel pour avoir le code le plus générique possible
En effet, même si dans tous les exemples traités, le nombre de bit par Pixel est 24, et le nombre de couleurs 3 (RVB) , j'ai préféré ne pas exclure la possibilité de changer ce paramètre.
Ainsi j'ai préféré utiliser un tableau de bytes à triple entrée, là où beaucoup d'autres ont fait un tableau de pixels à double entrée.

II)Les propriétés de MyImage

Mon programme possède une Class MyImage avec ses constructeurs.

MyImage possède 2 constructeurs utilisés en fonction des paramètres donnés.
Un pour créé une image vierge à partir de ses dimensions, avec une couleur de fond.
L'autre pour charger une image déjà existante.

Voici la liste des propriétés d'un objet MyImage
typeImage_0_2;
tailleFichier_2_4;
debutDeLImage_10_4;
tailleHeader_14_4;
largeurEnPixel_18_4;
hauteurEnPixel_22_4;
nombrePlanCouleur_26_2;
nombreDeBitParPixel_28_2;
tailleImage_34_4;
et enfin tableauDePixel;

J'ai nommé mes variables avec leurs positions dans le fichier ainsi que leurs tailles en nombres de bytes afin de ne pas avoir à rechercher sur Wikipédia à chaque fois. Ce fut pratique pour débuguer.

Quand je charge une image existante, l'intégralité de ses données sont récupérées directement depuis le fichier. Même pour celles qui ne changeront à priori jamais tel que le nombre de plan de couleurs (toujours égale a 1). Je fais cela dans le souci d'être le plus générique possible.
Tout est ensuite convertie en entier, grace à ma fonction fromEndianToInt.

TypeImage est également un entier, il est en pratique égale à 66+77*256 ce qui correspond a 66;77 en endian, il s'agit du code ASCII de « BM »

Mon tableau fonctionne de la maniere suivante :
Il a pour dimension la hauteur, puis la largeur, puis le nombre de couleur.
Le nombre de couleur est obtenu par le calcul (nombre de bit par pixel)/8
En pratique, c'est égale à 3 :
0 represente le bleu
1 represente le vert
2 represente le rouge

Le tableau de lecture du fichier, n'est pas conservé. Seuls les propriétés énoncées demeurent.

Toutes les caractéristiques sont privées, seules la hauteur et la largeur sont rendues accessible, mais uniquement en lecture. L'utilisateur n'a pas besoin du reste, mais il peut être amené à avoir besoin de la hauteur et de la largeur afin de ne pas tracer de segments à l'extérieure de l'image par exemple.

III) Fonctionnement du Menu

Pour naviguer dans le menu, il faut utiliser les flèches directionnels puis appuyer sur entrer.

Dans tout le menu, pour toute les informations demandée :
Si elle n'est pas dans le bon type, un message d'erreur s'affichera indiquant le type a utilisé, l'information sera redemandée.
Si elle est dans le bon type mais que la valeur est incorrecte, impossible, ou en dehors des limites du programmes, un message d'erreur générique s'affichera indiquant intervalle de la valeur a entrer, l'information sera redemandée.
Tout ces messages d'erreurs sont généré par les fonctions DemanderInt et DemanderDouble, les seules fonction en plus du main de la Class Program.

L'utilisateur doit tout d'abord charger une image.
Pour cela, il à 2 onglets à sa disposition : Il peux soit chargé une image existante, soit créé une image vierge.
Si il veux chargé une image existante, il peux entrer soit le nom de l'image tel que « coco » soit le nom complet du fichier tel que « coco.bmp »
Un message d'erreur s'affiche si le fichier est inexistant.
Si l'utilisateur exécute une autre fonction avant de chargé une image, coco sera utilisé par défaut.

Une fois l'image chargé, il est possible d'utilisé une fonction, ou d'en cumulé plusieurs.
Ainsi il est possible de créé une fractal, la superposera avec Lena , mettre le tout en noir et blanc, retournée l'image et rajouter un bonhomme tracer a partir de segments et polygones.

L'image ne s'affiche que lorsque l'utilisateur le demande grâce a l'onglet enregistrer et afficher.
L'image s'enregistre sur le fichier par défaut « sortie.bmp », mais il est possible de modifier cela grâce a l'onglet enregistrer sous.

Le menu est fait de la manière la plus générique possible. Un tableau regroupe la liste des intitulés des onglets. Il suffit d'ajouter un intitulé pour qu'il apparaisse. Il suffit ensuite d'ajouter le « case ». Si le « case » n'est pas ajouter, un message par défaut apparaît lors de la sélection de ce nouvel onglet.

IV) Informations complémentaire sur le fonctionnement de certaines fonctions.

Les matrices de convolutions sont toutes régies par la fonction AppliquerMatriceConv
qui prend en paramètre une matrice et un diviseur. Toute les autres fonctions du TD3 tel que le flou ou le renforcement des bord ne font qu'appeler cette fonction avec certains paramètres.

L'agrandissement et le rétrécissement peuvent prendre en paramètre un double, peu importe son nombre de chiffres après la virgule. Le double donné est transcrit en écriture fractionnaire, puis simplifier si la fraction est trop complexe. Ainsi, j'ai dû ajouter quelques petites fonctions utilitaires tel que le PGCD pour déterminer ma fraction.

Le PGCD fonctionne de manière récursive.

Pour ma fonction forme, j'ai choisi de toute les faire à travers la fonction « tracer un segment » Cela est en quelques sorte une innovation également car elle permet de dessiner ce que l'on veux sur l'image. Les fonctions polygones et étoiles ne sont que des exemples d'utilisations de cette fonction.

Les fonctions polygones peuvent avoir n'importe quel nombre de cotés, dans les limites conceptuelles et visuelles. J'ai tous de même préféré limiter le nombre de cotés (à un nombre très élevé) pour éviter que les calculs ne soit trop long.
La fonction polygone trace un semblant de cercle si le nombre de cotés entré est très élevé
La fonction étoile fonctionne même si l'utilisateur inverse le petit et le grand rayon.

Le tableau de pixel de l'histogramme remplace celui de l'image chargé, il ne faut alors pas oublier d' « enregistrer sous » son image avant, si l'on souhaite sauvegarder ses modifications.

Un pourcentage de chargement s'affiche pendant l'exécution des fonctions les plus longues, il provient de la fonction EtatChargement.

V) Mes innovations

1.) Le redimensionnement.

Les fonction agrandir et rétrécir peuvent prendre tous réels positifs en paramètre.
De plus, les allongement ou affinement en largeur et hauteur sont géré indépendamment.
Ces fonctions ont été faites avec le code le plus générique possible, spécialement pour pouvoir faire la fonction redimensionnement.
Il n'est pas seulement possible d'agrandir ou rétrécir, il est également possible d'allongé en largeur en réduisant la hauteur etc. La fonction redimensionnement n'est qu'une extension de l'utilisation des méthodes agrandir et rétrécir.

2.)Tracer un segment.

Pour tracer un segment, il faut les coordonnées en pixel de ses points de départs et d'arrivés.
Je laisse aussi choisir l'épaisseur du trait, ainsi que sa couleur.

Tracer un segment calcul une équation de droite : y=ax+b ou x=constante
Une fois l'équation de droite trouvée , on parcourt tous les pixel appartenant à l'intervalle allant du point de départ au point d'arrivé.
On calcul pour chaque pixel si il vérifie l'équation, si oui, il change de couleur.

Au delà du temps de calcul, on reste sur intervalle allant du point de départ au point d'arrivée afin d'obtenir un segment, et non une droite.
L'épaisseur correspond a la tolérance lors de la résolution de l'équation. Ainsi, l'épaisseur peux être choisi très précisément, avec un double.
Le sens d'entrée du point de départ et d'arrivé n'a pas d'importance, le segment sera tracé de la même manière.

3.)Le flocon

La méthode flocon ne trace pas un flocon seul. Il décompose de manière récursive un segment, il faut trois segment décomposé pour former un flocon.

La méthode prend donc en paramètre un segment. Si le nombre d'itérations restantes est 0, alors, les coordonné du segment sont envoyé à la fonction tracer un segment. Sinon, le segment est décomposé en 4 segments 3 fois plus court et rebelote avec un nombre d'itérations réduit.

Il faut trois segment placé de manière à former un triangle équilatéral afin d'obtenir un flocon commun. Mais tout est générique, donc la méthode peux tracer un segment a partir de n'importe-quelle forme. Pour que ça soit plus simple a utilisé , je n'ai pas laissé le choix à l'utilisateur, j'ai arrangé les paramètre dans le menu afin de toujours avoir un résultat convenable.

4) La fractale de Mandelbrot

Une fractale est une figure complexe qui possède une infinité de détails, quelle que soit l'échelle à laquelle on la regarde. De plus on retrouve souvent le même motif ou un motif similaire dans le motif de la fractale en lui-même. Pour finir, il faut dire qu'une fractale est trop irrégulière pour que l'on puisse la décrire par des termes géométriques habituels .

L'ensemble de Mandelbrot est une fractale qui est définie comme l'ensemble des points c du plan complexe pour lesquels la suite récurrente définie par $Z_{n+1}=Z_n^2+c$ et la condition $Z_0=0$ ne tend pas vers l'infini (en module) (Wikipédia). Mais l'infini c'est trop grand pour moi alors je me suis limité à 2. Je teste si le module de Z_n dépasse 2 à un moment. S'il ne le dépasse pas, c'est qu'il fait partie de la fractale.

L'algorithme n'est en soit pas très compliqué :

Pour tout chaque coordonnées du plan on fait
c=x+iy ; Z=0 ;compteur=0 ;
On répète z = z*z + c le nombre de fois demander par l'utilisateur (itérations)
Si z < 2, alors il fait partie de la fractal, youpi, le pixel change de couleur.

Une partie plus compliquée fut de retranscrire les nombres complexes en C#, d'où la complexité apparente des calculs.
j'ai essayer malgré tout de tous simplifier, par exemple pour le calcul du module, je me contente de calculer le carré du module et de le comparé à 4 plutôt que de faire une racine a chaque fois.

Je calcul le zoom en fonction de la taille de l'image, pour avoir un rendu sympa peu importe l'image en entré.

La fractal est défini par sa forme , j'ai quand même rajouter de la couleur. Afin de ne pas avoir à entrer chacune des couleurs, ou même d'enregistrer un tableau de couleur, je créé une couleur à partir d'une graine que je multiplie par l'itération en cours. La graine est donnée par l'utilisateur. J'indique un exemple de graine joli dans le menu, car il est très dure de prédire les couleurs d'arrivées.

5) Les fractales de Julia

Elles fonctionnent de la même manière que la fractale de Mandelbrot. Mais les conditions initiales sont différentes. Il a fallu rendre le code plus générique. La méthode Fractal de Julia, prend en paramètre le nombre complexe c sous la forme c_i et c_r. J'ai laissé en commentaire dans le menu certaine valeur avec un bon rendu, mais encore une fois, je ne laisse pas le choix a l'utilisateur.