Projet : Manipulation d'images en ligne de commande

Le projet de cette année concerne la mise au point d'une bibliothèque de programmes pour manipuler les images depuis la ligne de commande. Cette bilbiothèque doit permettre l'application de filtres simples sur des images. Techniquement, une telle bilbiothèque existe déjà, il s'agit de ImageMagick [1], mais il faut bien un pretexte.

1) Description globale

Bien que le résultat final soit présenté comme une bibliothèque, il n'y a en fait qu'un seul programme à écrire. Ce dernier se comportera différemment selon la valeur de son premier argument. Pour rappel, l'ensemble des arguments reçus en ligne de commande sont accessibles par l'intermédiaire du tableau `argv[]`. Le premier argument sera donc `argv[1]`.

Les fonctions seront ajoutées au fur et à mesure du projet pour venir étoffer les possibilités de programme.

Deux formats de fichiers sont possibles, le format PPM que vous avez utilisé jusqu'ici dans les TPs, et le format JPG, plus complexe, mais beaucoup plus répandu. Votre programme pourra aisément déterminer quel format utiliser en fonction du nom des fichiers (sourtout de leur extension). Pour gérer le format JPG, une bibliothèque vous est fournie, à vous d'en faire bon usage. Cette dernière fait appel à `<jpeglib.h>`, qui a besoin de l'option `-ljpeg` à la compilation.

2) Description particulière des fonctions

La liste de fonctions à implémenter est la suivante (par ordre de difficulté). Un seul type de fichier (PPM ou JPG) est demandé au départ, mais à un moment, il faudra gérer les deux types de fichier. Votre programme identifiera le format grace à la valeur du premier argument.

1. `copy <fichier_origine> <fichier_destination>`
   Effectue une copie du fichier sans en modifier le contenu. Si les formats sont différents, effectue une conversion de format.

2. `negative <fichier_origine> <fichier_destination>`
   Inverse les couleurs de l'image en mode négatif photo. Pour chaque composante '*c*' de l'image d'origine, la composante équivalente doit valoir '*255-c*' sur l'image de destination.

3. `grayscale <fichier_origine> <fichier_destination>`
   Passe en mode monochrome. Ce qui signifie que seule l'information d'intensité lumineuse est gardée. Pour simplifier, nous considérerons que la valeur d'intensité lumineuse est obtenue en faisant la moyenne des trois composantes. Cette valeur d'intensité lumineuse est alors recopiée sur les trois composantes de l'image de destination pour obtenir des variations de gris

4. `color <fichier_origine> <fichier_destination> <col_r:col_g:col_b>`
   Applique un filtre d'atténuation dont la couleur de base ets donnée en troisième valeur. Poru

chaque composante '*c*' de l'image d'origine, si la composante correspondante de la couleur en argument est '*f*', alors la composante de l'image de destination sera *c*f/255*

Par exemple, `color img1.ppm img2.ppm 255:255:0` va supprimer toutes les composantes bleues de l'image.

5. `blur <fichier_origine> <fichier_destination>`
   Rend l'image plus floue. Cette technique s'effectue en calculant pour chaque pixel de destination, une moyenne pondéree des pixels autour des pixels equivalents dand l'image d'origine. Un début intéressant est d'utiliser les coefficients de la table suivante (le coefficient en rouge représente le pixel de référence) :

1/16	1/8	1/16
1/8	1/4	1/8
1/16	1/8	1/16

   La difficulté de cette partie vient des coins et des bords qui obéissent à une formule différente puisqu'il y a moins de pixels à moyenner.

Vous êtes bien entendu libres d'ajouter votre/vos propre(s) fonction(s).

3) Consignes générales

Nous sommes dans l'esprit d'un projet, vous avez donc toute liberté quant à la méthode à utiliser pour produire le résultat attendu, moyennant les restrictions suivantes :

- Le résultat que vous présenterez doit être le fruit d'un travail personnel (à toutes fins utiles, précisons que l'intérêt du projet n'est pas tant le résultat final que les compétences que vous aurez mises en œuvre pour y parvenir)
- gérez votre espace disque : veillez à ne pas garder trop de versions haute résolution de vos images de test

Quelques conseils :

- Le résultat demandé est attendu en fin de projet. Il vous appartient de fixer des objectifs intermédiaires pour aider le débogage pendant le développement et continuer sur des bases saines. Partir directement vers le résultat final est la meilleure technique pour ne pas y parvenir.
- Vos encadrants AUSSI, savent utiliser une connection internet et un moteur de recherche...

4) Evaluation

L'évaluation finale aura trois composantes :

- du contrôle continu
- la qualité technique de votre projet
- la qualité de votre rapport

Le rapport sera à remettre à votre encadrant dans les jours qui suivent la dernière séance de projet. Pensez à anticiper son écriture si vous en avez l'opportunité.

Votre rapport doit expliquer l'ensemble des démarches que vous avez dû mettre en œuvre pour le

projet. Il doit au moins contenir (liste non exhaustive) :

- vos choix d'implémentation
- les algorithmes (principal, ainsi que pour les sous-fonctions les plus complexes ou intéressantes)
- représentation des données (si nécessaire)
- technique de validation des fonctions
  - à ce propos, il existe une grosse différence entre une fonction qui a pu produire un bon résultat une ou deux fois, et une fonction qui produit un bon résultat tout le temps.
- analyse des performances
  - estimation de la complexité
  - explicitation des étapes les plus lourdes/limitantes
  - discours sur le caractère optimal et/ou sur les pistes d'amélioration
- proposition d'amélioration de votre travail (si le temps vous l'avait permis)

Toute remarque de votre encadrant est prioritaire sur ce sujet