Projet 2 : Pré-rapport

Récupération des arguments

Dans ce projet l'utilisateur aura la possibilité d'introduire des arguments via la console. Comme les filtres à appliquer aux images et les differents threads correspondant à chaque filtre. Il est donc indispensable de pouvoir récupérer ces donnée qui seront stockées dans les paramètres de la fonction main(). Et grâce à getopt(), nous allons pouvoir les récupérer afin de pouvoir les utiliser.

On stockera ainsi les filtres à utiliser dans un premier tableau qu'on aura alloué et que l'on réallouera si besoin. On stockera ensuite le nombre de thread correspondant à chaque filtre dans un autre tableau et les paths du dossier "in" et "out" dans dans des varibles.

Les threads

Un des objectifs de ce projet sera d'alléger une tache en la décomposant en plusieurs parties (threads), qui vont s'exécuter simultanément en partageant les même données en mémoire.

La difficulté sera donc de pouvoir synchroniser les différentes parties. Et pour éviter les problèmes de syncronisation, nous allons utiliser les mutex, qui empêcheront notamment à plusieurs threads d'accéder à une même variable au même moment. Et le tout se fera grâce à la bibliothèqe pthread qui permet de manipuler les threads, les processus et les mutex assez facilement.

Un des objectifs de ce projet sera d'alléger une tache en la décomposant en plusieurs parties (threads), qui vont s'exécuter simultanément en partageant les même données en mémoire.

La difficulté sera donc de pouvoir synchroniser les différentes parties. Et pour éviter les problèmes de syncronisation, nous allons utiliser les mutex, qui empêcheront notamment à plusieurs threads d'accéder à une même variable au même moment. Et le tout se fera grâce à la bibliothèqe pthread qui permet de manipuler les threads, les processus et les mutex assez facilement.

Nous allons poser le problème ainsi :

Il y aura des**N thread producteur (selon les filtres)** qui liront les fichiers images contenu dans le dossier in et on ajoutera chaque fichier lu à l'arrière d'une **queue**. Ainsi tous les fichiers images se trouverons dans une queue.

Ensuite il y aura plusieurs **N threads consommateurs** qui traiterons les fichiers image contenus dans la queue (qui servira de buffer).

Filter functions

-Blur filter sera basé sur le produit de convolution d'une image avec un noyau de valeurs gaussiens. Dans notre cas, nous diviserons le processus en étapes : Tout d'abord, un noyau à une dimension est utilisé pour appliquer le filtre dans une direction seulement, dans le sens horizontal de l'image. Ensuite, un autre noyau unidimensionnel est utilisé pour appliquer le filtre sur le reste de l'image.

De cette façon, le résultat sera le même qu'une convolution avec un noyau en deux dimensions, mais il faudra moins de calculs.

On peut calculer le noyau unidimensionnel Gaussien à l'aide de formules suivantes :

Dans notre cas, quand x peut être la distance du point d'origine x ou de l'axe y

Enfin, nous pouvons simplifier l'opération de convolution en utilisant la transformation de Fourier comme ci-dessous :

- calculer la Transformée de Fourier rapide de l'image et du noyau unidimensionnel gaussien
- calculer le produit de deux Transformée de Fourier rapide
- calculer le contraire du transformée de

Fourier rapide du résultat.

-Sur une image de l'échelle gris, chaque pixel est un échantillon unique, qui contient des informations l'intensité. Cela signifie, que la valeur 0 correspond au noir et que la valeur 1 correspond au blanc. L'algorithme simple est basé sur le calcul de la moyenne des composants (r, g et b) dans chaque pixel, pour ensuite, définir le pixel de cette valeur

Le filtre rouge et bleu a la même structure que le filtre vert. Seulement, il prend les composants rouge/bleu de chaque pixel (un par un) et le transforme en 0. En utilisant ce processus, nous pouvons éviter d'utiliser les composants rouge et bleu de toute l'imagine

Switch filter. Il mélange les composants de couleurs différentes. L'algorithme prend la valeur de chaque composante et les définit comme ci-dessous :

- le composant ß bleu précède le composant vert
- le composant ß rouge précède le composant bleu

- le composant ß vert précède le composant rouge