M2 - Architecture et Programmation d'accélérateurs Matériels.

(APM 2021-2022)

Mini Projet

CUDA

julien.jaeger@cea.fr
adrien.roussel@cea.fr

Les objectifs de ce TP sont :

- Utilisation de CUDA pour faire du traitement d'images
  - Effet noir et blanc,
  - Détection de contours,
  - Saturation des couleurs,
  - ...

Nous allons modifier une image afin de réaliser plusieurs transformations sur tout ou une partie de celle-ci. Une image est un tableau de pixels, encodé selon la norme RGB. Plus précisément, chaque pixel dispose de trois composantes : R (rouge), G (vert) et B (bleu).

I Modalités de RENDU

- Le projet se fera en binome (monome autorisé)
- La liste des binomes doit être connue pour le 12 Janvier 2022
- Le projet est à envoyer par mail aux intervenants du module
  - Adrien Roussel : adrien.roussel@cea.fr
  - Julien Jaeger : julien.jaeger@cea.fr
- UNIQUEMENT LES SOURCES (pas les images, pas de binaires, pas la bibliothèque FreeImage)
- DEADLINE
  - **16 Février 2022 23h59 (dernier délai)**

II Rappel : utilisation de la commande `scp`

La commande `scp` permet de transférer des fichiers et dossiers depuis et vers une machine distante. N'oubliez pas de remplacer la valeur de *<login>* par votre login dans chacune des commandes suivantes, ainsi que chacune des valeurs entre chevrons "<...>".

Q.1: Depuis votre machine personnelle, pour transférer des fichiers vers une machine distante (Ruche dans cet exemple), utiliser la commande suivante :

```
scp -r <fichier>
<login>@ruche.mesocentre.universite-paris-saclay.fr:/home/<login>/<dossier
destination>
```

Q.2: Depuis votre machine personnelle, pour récupérer des fichiers stockés sur une machine distante (Ruche dans cet exemple), utiliser la commande suivante :
`scp -r <login>@ruche.mesocentre.universite-paris-saclay.fr:`
`/home/<login>/<dossier>/<fichier> <destination>`

III Installation

Afin de convertir une image compressée (format JPG, PNG, BMP, ...) en un tableau de bits, nous avons besoin d'une bibliothèque capable de faire ce traitement. Nous allons installer la bibliothèque **FreeImage**, disponible dans le répertoire *CODE/FreeImage/FreeImage3180.tar.gz*
Cette opération n'est à réaliser qu'une seule fois. Dès que la bibliothèque est installée, inutile de la réinstaller plusieurs fois (elle ne bougera pas du dossier où vous l'avez installé). Voici les étapes nécessaires à l'installation de la bibliothèque.

Q.3: Depuis Ruche, décompresser le dossier contenant la bibliothèque, grâce à la commande suivante :
`tar xvzf FreeImage3180.tar.gz`

Q.4: Une fois dans le dossier *FreeImage*, compiler la bibliothèque grâce à la commande :
`make ; make install`
Par défaut, la bibliothèque va être installée dans le dossier : `$(HOME)/softs/FreeImage`

Q.5: Ajouter la bibliothèque à votre environnement, en modifiant la variable d'environnement `LD_LIBRARY_PATH`, grâce à la commande suivante :
export LD_LIBRARY_PATH=$HOME/softs/FreeImage/lib :$LD_LIBRARY_PATH
Ajoutez cette ligne dans le chargement de votre environnement à chaque connexion sur la machine (pour ne pas avoir à refaire cette étape à la prochaine connexion) :
echo "export LD_LIBRARY_PATH=$HOME/softs/FreeImage/lib :$LD_LIBRARY_PATH" » /.bashrc

IV Modification d'image : Saturation de pixel

Le programme *modif_img.cpp* présent dans le répertoire *CODE* lit une image, la traduit sous forme de tableau de pixels (*i.e.* char), et recopie ce tableau dans le fichier *new_img.png*. Vous pouvez compiler et exécuter le programme, en utilisant l'image test fournie *img.jpg*, pour voir l'effet du programme sur l'image générée *new_img.png*.

Le programme n'attend plus que vos modifications !

Q.6: Pour chaque pixel de l'image, écrivez un programme CUDA (transferts mémoire, configuration d'une grille 2D, noyau de calcul GPU) qui sature l'une des composante de chaque pixel (parmi R, G ou B).

Q.7: A partir du programme précédent, écrivez le noyau CUDA qui permette de faire une symétrie horizontale de l'image.

V Flou et Flou itératif

Q.8: Dans ce TP, nous allons implémenter une nouvelle modification d'image : le flou. Le flou est une opération de stencil : pour mettre à jour la valeur de la case, nous avons besoin des donnés des cases voisines. Pour effectuer un flou, il suffit de mettre à jour un pixel avec la moyenne des valeurs de ce pixel et des pixels voisins, pour chaque composante. Dans notre cas, nous allons uniquement considérer les voisin directs. Pour un pixel[i][j], il s'agit des pixels [i-1][j], [i+1][j], [i][j-1] et [i][j+1].

Implémenter le kernel permettant de faire un flou sur l'image.

Q.9: Nous allons à présent implémenter une méthode permettant de faire une image en niveau de gris. Pour cela, les trois composantes d'un pixel ont la même valeur. L'oeil étant plus sensible à certaines couleurs qu'à d'autres, une moyenne arithmétique n'est pas adaptée à la transformation désirée. On passe alors par une moyenne pondérée, via la formule suivante :
$p_{i,j}^{RGB} = 0.299 \times p_{i,j}^{red} + 0.587 \times p_{i,j}^{green} + 0.114 \times p_{i,j}^{blue}$.

Ecrire un kernel CUDA qui permette d'appliquer cette transformation sur l'image.

Q.10: La méthode de *SOBEL* fournie (dossier *docs*) permet de détecter les contours sur une image quelconque. Ces contours sont alors affichés en blanc sur fond noir. Comme la question précédente, ecrivez le code CUDA permettant de réaliser la mèthode de SOBEL sur le GPU.

Q.11: Quelques suggestions supplémentaires de transformations que vous pouvez effectuer sur l'image via un kernel CUDA pour vous amuser :

- Redimensionnement (interpolation bilinéaire),
- Rotation,
- Ne garder qu'une composante par pixel, et mettre les autres à 0,
- Effet diapositive : à chaque composante c d'un pixel, appliquer la formule $c = 255 - c$.
- Filtre de Canny

VI Utilisation de Streams

Q.12: Effet Pop-art – Warhol. Découpez votre frame originelle en 4 parties identiques. Sur chacune des parties, affichez une frame réduite à la taille appropriée. Dans chaque cadre, saturez une couleur pour avoir quatre images identiques, avec des couleurs différentes. Implémentez cette fonctionnalité à l'aide d'un kernel CUDA.

Q.13: Est-ce que ce programme est efficace ? Pourquoi ? Quelle modification proposez-vous ?

Q.14: Modifiez votre code pour utiliser 4 streams CUDA pour effectuer une modification de cadre par Stream.

Exemple que vous devriez avoir

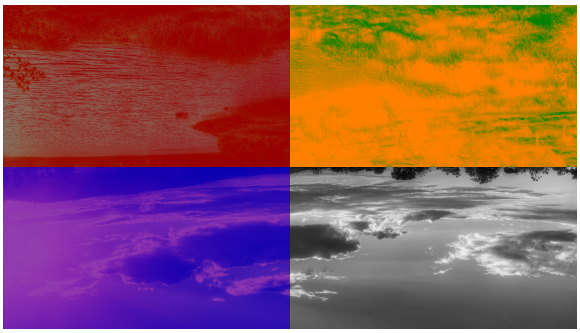

FIGURE 1 – Exemple effet Pop-art