Noé Allard
Yohann Baudet
Matysse Descamps

Rapport CUDA
2021/2022

1- Algorithme implémenté

Pour pouvoir implémenter ces différentes convolutions, nous sommes partis sur la base du greyscale du TP4. Tout d'abord, nous avons modifié le programme cpu pour que la matrice modifiée par le gpu soit du même type que la matrice de l'image d'entrée. C'est-à-dire qu'elle possède 3 canaux, un pour chaque composante du RGB. Après avoir réussi à simplement copier la matrice de l'image d'entrée dans la matrice de l'image de sortie, nous avons commencé l'implémentation de la convolution pour le flou. Après cette dernière implémentée, nous avons implémenté la détection de contour. Pour implémenter d'autre convolutions, il suffit de rajouter de nouveaux kernel possèdant une matrice filtre propre nécessaire pour la convolution voulue.
Actuellement, notre algorithme combine les convolutions du flou et de la détection du mouvement.

2- Optimisations choisies

Le nombre de blocs est déterminé en fonction de la taille de l'image d'entrée.

3- Difficultés rencontrées

Notre premier problème rencontré est que nous voulions tester avec une image en couleur cependant en rentrant une image en couleur l'algorithme, en utilisant la matrice identité pour avoir la même image, nous retournait une image en noir et blanc, la liste retournée était 3 fois moins grande, car nous retournions qu'un tic de couleur par pixel, par la suite nous avons réglé ce problème, mais rien ne changea. C'est pourquoi nous avons commencé à regarder les fonctions utilisées pour créer l'image. En regardant en profondeur, nous avons vu qu'une fonction utilisait 8UC1, ce qui veut dire que cette fonction prenait une image en 8bit avec 1 channel de couleur. Pour régler le problème, il fallait changer le C1 en C3, ce qui permettait à la fonction de prendre en compte les autres tic de couleurs de chaque pixel (RGB). Après avoir réglé ce problème nous avions notre image retournée en couleur, toujours en utilisant la matrice identité, cependant, nous n'avions que 1⁄3 de l'image. Le problème venait d'un autre endroit où il fallait traiter les 3 canaux de couleurs, la fonction précédente n'était pas la seule à devoir être modifiée, pour régler ce problème, il fallait modifier le kernel pour qu'il puisse traiter les 3 canaux de couleurs d'un pixel. Sans régler ce problème, le kernel ne traitait qu'une partie des pixels.

Un autre problème majeur a été de pouvoir coder en cuda, car nous n'avions pas de carte graphique nvidia, nous étions obligé de coder sur les machines à distance. Ce qui était très lent et pénible.

4- Résultats Obtenus

Nous avons implémenté des timer afin de mesurer le temps d'exécution de tout le programme. Cependant ce timer nous renvoyait tout le temps zéro, donc nous n'avons pas pu comparer le temps d'exécution du programme en fonction des différentes convolutions voulues.