TD GPGPU n°3

Cours de GPGPU
MII 2 option SIS / IMAC 3ème année

GPGPU grâce à Cuda

Premier pas pour la programmation efficace : mémoire partagée, opérations atomiques, coalescence des accès mémoire.

Le but de ce TD est d'appliquer les concepts de programmation efficace quand on utilise et exploite le GPGPU en Cuda, notamment via la mémoire partagée.

Exercice 1 : SAT via la mémoire partagée

L'algorithme SAT, acronyme de Summed Area Table, permet de créer à partir d'un tableau de donnée 2D (comme une image par exemple), un tableau dont les cases possèdent la somme de toutes les cases des lignes et colonnes précédentes.

$$s_{mn} = \sum_{i=1}^{m} \sum_{j=1}^{n} t_{ij}$$

	1	2	3	4
1	2	3	2	1
2	3	0	1	2
3	1	3	1	0
4	1	4	2	2

Original

2	5	7	8
5	8	11	14
6	12	16	19
7	17	23	28

Summed-area table

Un tel tableau a de nombreux avantages (voir question 5).

Du code source vous est donné (sur mon site), qui charge un tel tableau de donnée à partir d'une image.

1. Transformer les données issues de l'image en un tableau de unsigned int (afin de pouvoir faire la somme correctement)
2. Déterminez la taille maximal du tableau (carré) 2D afin de ne pouvoir utiliser qu'un seul bloc de thread.
3. Implémentez le calcul sur GPU du tableau SAT résultat (sans utiliser de mémoire partagée). Comparez les temps de calcul avec le CPU.
4. Implémentez le calcul sur GPU avec utilisation de mémoire partagée. Comparer avec les autres temps de calcul. Peut on utiliser plusieurs autres blocs ?
5. Utilisez le tableau SAT pour réaliser une application de flou moyen sur l'image...

Exercice 2 : Histogramme

Reprendre le code précédent et chargez cette fois une image plus « conséquente » (fichier *halloween.png* dans le répertoire data). Le but de cet exercice est de calculer l'histogramme de cette image, passée en niveau de gris.

1. Calculer en CPU un histogramme du niveau de gris de cette image. Le niveau de gris s'obtient par le calcul de la luminance vu aux précédents TD.
2. Calculer grâce à des opérations atomiques, un histogramme sur le GPU. Pour ce faire, vous devrez d'abord appeler un kernel (déjà vu précédemment) permettant de transformer l'image en luminance, puis appliquer un nouveau kernel sur l'image résultante.
3. Sachez qu'il est possible de calculer l'histogramme de l'image en utilisant la mémoire partagée et en simulant des opérations atomiques. C'est expliqué ici : http://users.cecs.anu.edu.au/~ramtin/papers/2007/ICSPCS_2007.pdf

Exercice 3: Image blur

A partir d'une image en niveau de gris, réalisez le kernel permettant de faire un flou gaussien de l'image (avec un masque gaussien 5x5) en utilisant le plus efficacement possible la mémoire partagée pour gagner du temps de calcul.