GPGPU

Exercice 2 :

1. How many floating operations are being performed in your blurring kernel ?

Nous cherchons à obtenir une image en nuance gris depuis une image RGB. De ce fait, le pixel RGB (le pixel R, le pixel G et le pixel B) sont additionnées afin de créer un pixel en nuance de gris.

Ainsi, il y a qu'une opération flottante pour chaque pixel.

Ici, nous prenons une image de 512x512 pixels en entrée, soit 262 144 pixels.

Il y a donc 262 144 opérations flottantes qui sont effectuées.

2. How many global memory reads are being performed by your kernel ?

En lecture mémoire, il y aura le nombre de fois que chaque pixel est lu sur les 3 channels.

Ici, nous prenons une image de 512x512 pixels en entrée, soit 262 144 pixels. Puis, comme il y a 3 opérations sur ces 262 144 pixels, il y au total 786 432 éléments qui sont lus.

Il y a donc 786 432 lectures en mémoire.

3. How many global memory writes are being performed by your kernel ?

Comme dit précédemment, on aura en sortie une image de nuance gris. Il y aura donc le nombre de pixel d'écritures en mémoire.

Ici, cela sera 262 144 écritures en mémoire.

4. How do you choose the CUDA thread grid dimensions ?

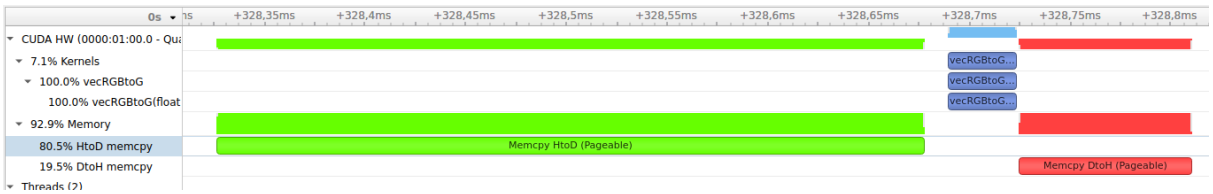
On choisit la dimension de grid et thread en fonction du nombre de pixels, soit le nombre d'opération à exécuter.

Par exemple, ici nous avons une image de 512x512. En prenant des blocks de 32x32 contenant des threads de 16x16, il y a donc 512x512(16*32) opérations qui vont se faire. Soit le nombre de pixel de l'image.

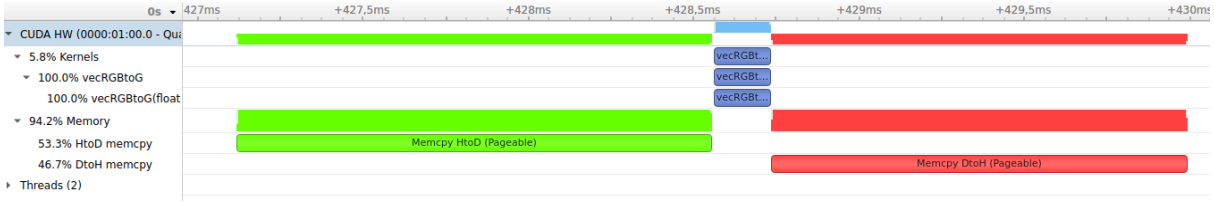
Ça rejoint donc le nombre de pixel de l'image. Chaque pixel de l'image sera analysé.

On peut s'amuser à comparer avec un autre choix de dimension de block et thread. J'ai donc testé, avec l'image 1080x1080, en changeant les tailles de blocks (34x34) et threads (32x32).

Voici le profilage de l'image de Lena1080 avec les dimensions 32x32 blocks et 16x16 threads.



Voici le profilage de l'image de Lena1080 avec les dimensions 34x34 blocks et 32x32 threads.



On remarque également une légère différence à la sortie de l'image en nuance de gris.

32x32 blocks et 16x16 threads :

34x34 blocks et 32x32 threads :

Exercice 3

1. How many floating operations are being performed in your blurring kernel ?

Ici une opération est effectuée soit sur 3 soit sur 1 channels (RGB et nuance de gris respectueusement) de chaque pixel.

Cependant, nous faisons, pour chaque pixel, une moyenne de lui avec ses voisins (ses 3x3 voisins). Pour cela, nous les additionnant tous avant de les diviser par le nombre de pixel que nous avons pris.

De ce fait, il y a :

512x512 [nombre de pixel] * channels*(9 [nombre de pixel que nous avons pris] + 1 [division])

- En RGB : 7 864 320 opérations flottantes
- En gris : 2 621 440 opérations flottantes

2. How many global memory reads are being performed by your kernel ?

En lecture mémoire, il y aura le nombre de fois que chaque pixel est lu ainsi que ses voisins, tout cela par channel également.

De ce fait, il y a :

512x512*(nombre de pixel que nous avons pris : souvent 9)*channels

- En RGB : 7 077 888
- En gris : 2 359 296

3. How many global memory writes are being performed by your kernel ?

On aura en sortie la même image (512x51é pixels) et pour chaque pixels il y a 3 (RGB) ou 1 (gris) channels.

De ce fait, il y a :

512x512*channels

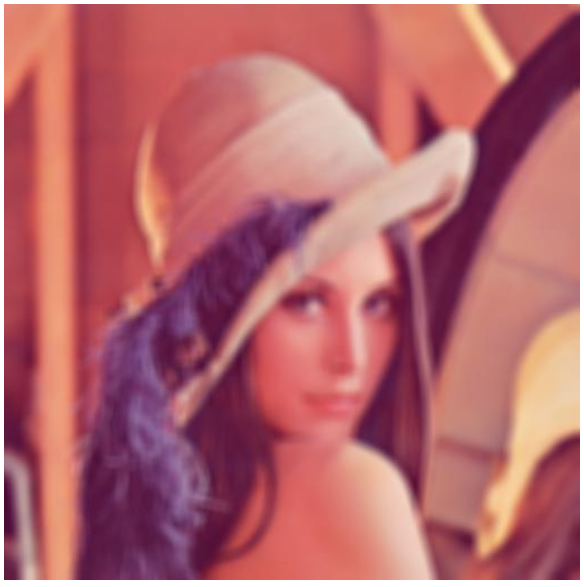
- En RGB : 786 432
- En gris : 262 144

4. What could you intent to do in order to have a stronger blur effect ? Try in your kernel, and compare the results.

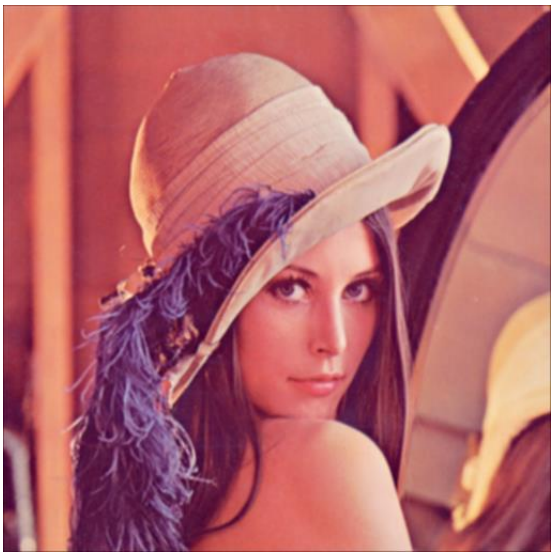
Afin de flouter davantage notre image, nous devons, pour chaque nouveau pixel, faire la moyenne d'un plus grand nombre de pixel autour de celui-ci.

```
for(int blurRow = -5;blurRow <6; blurRow++) {
    for(int blurCol = -5;blurCol <6; blurCol++) {
```

De ce fait, j'ai testé de le faire avec un intervalle de 5 au-dessus et au-dessous du pixel que l'on traite. Nous avons ce retour :



Au lieu de cela (obtenu avec le 3x3) :



On remarque bien, que plus il y a de voisin prit en compte, plus l'image sera flou.

De plus, nous remarquons que :

En 3x3 :

En 11X11 :