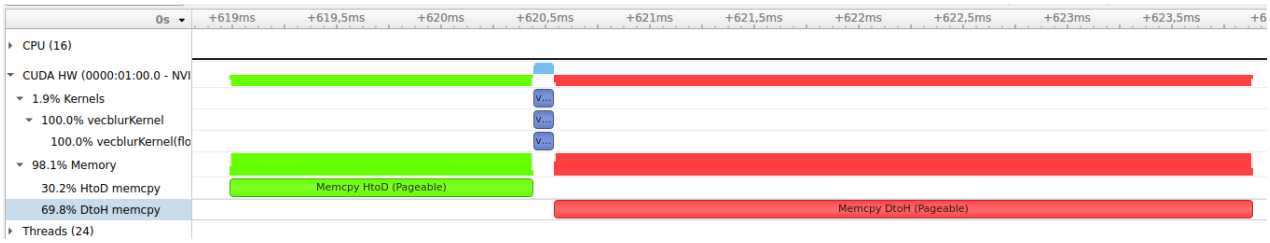
Chapitre 12 – Exercice 2

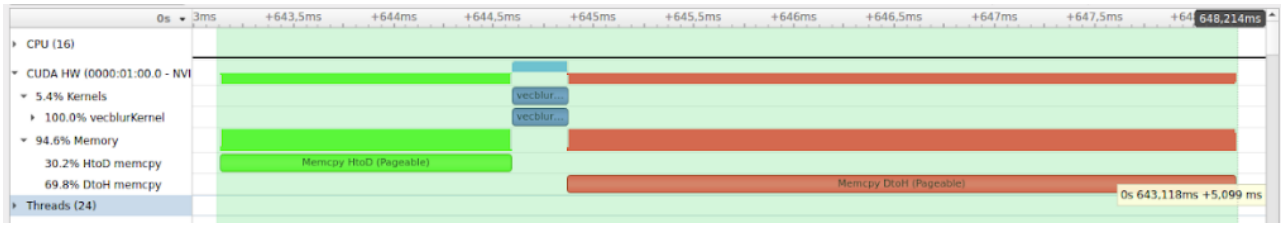
Profile your code from Chapter 3 with convolution kernel using a kernel with an average of the 7×7 neighbor pixels (instead of 3×3 in chapter 3) on the 4k image.

Avec un flou de 3x3 :



Malheureusement, j'ai mal pris ma capture.

Et avec un de 7x7 :



Le kernel pour un flou autour de 3x3 pixels, mets environ 23 microsecondes à s'exécuter. Or, quand nous travaillons avec 7x7 pixels, le temps est d'environ 62 microsecondes. On remarque donc que le temps est presque multiplié par 3.

Adapt the algorithm to handle the entire image with streams and pinned memory. Explain your strategy.

Dans mon kernel, je prends en compte si l'image est en RGB ou en gris. J'applique le kernel du chapitre 3 en rajoutant une condition au tout début (ligne 20).

if (y < 3 || x < 3 || y > height - 4 || x > width - 4) {

Celle-ci prends les conditions de l'ajout de stream. En effet, notre image se retrouve coupé à chaque fin stream lorsque nous faisons en sorte d'ignorer cette condition :

if (y < 3 || x < 3 || y > height - 4 || x > width - 4) {

return ;

}

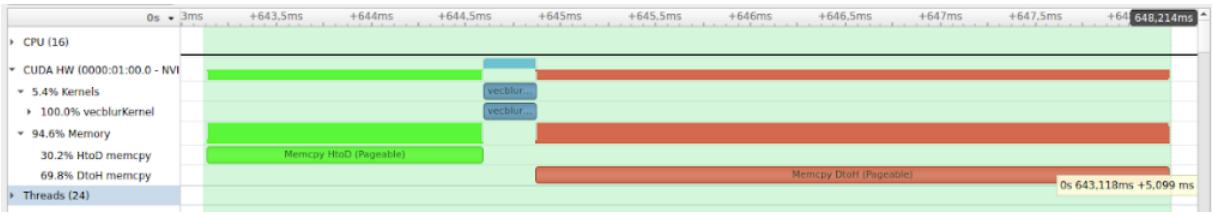
Pour corriger ce problème, je fais la moyenne des pixels sur cette ligne mais uniquement avec les pixels présents dans les colonnes autour. Le pixel de la ligne ne change pas car il ne peut pas prendre les pixels autour car ils sont alloués dans différents tableaux.

Dans ma fonction main, j'épingle ma mémoire avec *cudaHostAlloc* et je la libère à la fin avec *cudaFreeHost*. Et je crée différents streams : 4. Ainsi, je peux allouer dans 4 tableaux différents mes pixels sur le device. Enfin, j'appelle ma fonction kernel et obtient une image floue.

Voici le rendu avec le tigre :

Profile the new version and report the total execution time from start of the first copy to end of last copy. Compare with the initial version.

Initial :



Nouvelle :

PAS DE PROFILAGE POSSIBLE EN DISTANCIEL.