

Computación Paralela y Distribuida - Practica #3

Angel David Corredor Morales
adcorroedorm@unal.edu.co

Sebastian Felipe Delgado Perez
sfdelgadop@unal.edu.co

I. INTRODUCCIÓN

Este documento recopila los resultados presentados al realizar el tercer laboratorio de la materia Computación paralela y distribuida del departamento de ingeniería de sistemas e industrial dictada a la carrera de ingeniería de sistemas y computación, el taller consta de desarrollar el efecto de distorsión en múltiples hilos usando GPU con cuda.

II. EFECTO DE DIFUMINADO

El efecto de difuminado se logra usando una matriz de convolución kernel, también conocido como máscara, básicamente cada píxel pasa por el kernel, el cual podríamos considerar un “filtro”, y el nuevo píxel es calculado usando los píxeles adyacentes en conjunto con el kernel como se ve en la imagen 1.

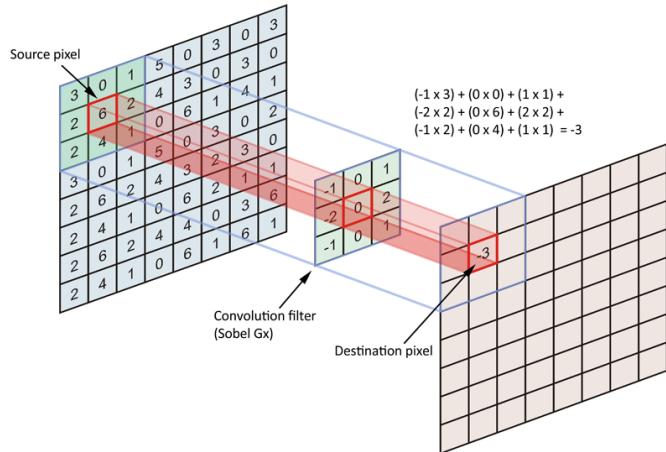


Fig. 1: Operación de Convolucion basada en Kernel
https://github.com/OKStateACM/AI_Workshop/wiki/Computer-Vision-with-Convolution-Networks

El cálculo del nuevo píxel es una convolución entre dos matrices, en este caso en particular se decidió por un promedio de los píxeles circundantes para generar el efecto de difuminado.

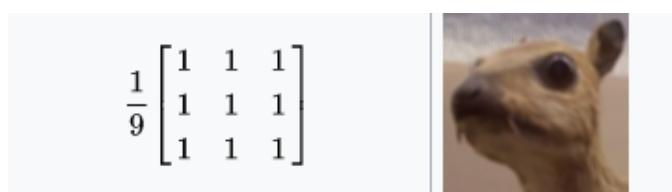
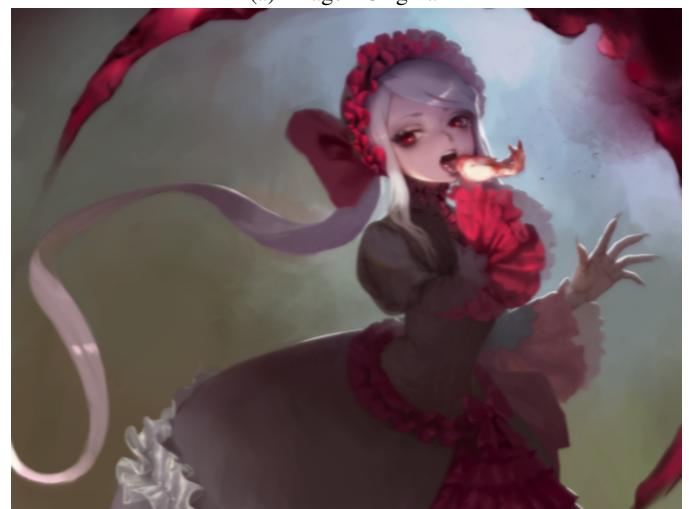


Fig. 2: Ejemplo de kernel de difuminado tamaño 3
[https://en.wikipedia.org/wiki/Kernel_\(image_processing\)](https://en.wikipedia.org/wiki/Kernel_(image_processing))

En la figura 3b se puede observar el resultado de aplicar el filtro de difuminado con un tamaño de kernel igual a 15.



III. OBJETIVO Y ENTORNO DE TRABAJO

El objetivo de la práctica es ver cómo se comporta el paralelismo realizado con diferentes hilos y kernels de tamaño distinto. Las pruebas se desarrollaron en un computador personal con una cpu ryzen 5 3600 de 6 nucleos con 12 hilos, 16 Gb de Ram ddr4 a 3000Mhz y una tarjeta gráfica nvidia gtx 1050ti de referencia Rog strix de Asus la cual cuanta con 768 cuda cores y 4Gb de memoria Gdrr5 y los resultados que se presentan a continuacion corresponden al promedio de tres

ejecuciones para reducir la aleatoriedad al momento de ejecutar los experimentos.

IV. RESULTADOS

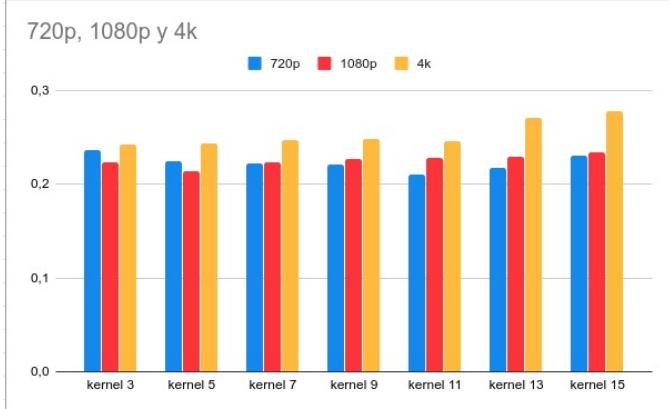


Fig. 4: Resultados obtenidos en los experimentos

Como se puede observar en la figura 4, los resultados para cada uno de los tamaños de kernel en imágenes de distintas resoluciones son altamente similares entre si, habiendo únicamente picos notables en los tamaños de kernel 13 y 15 cuando la imagen tiene una resolución de 4k.

V. CONCLUSIONES

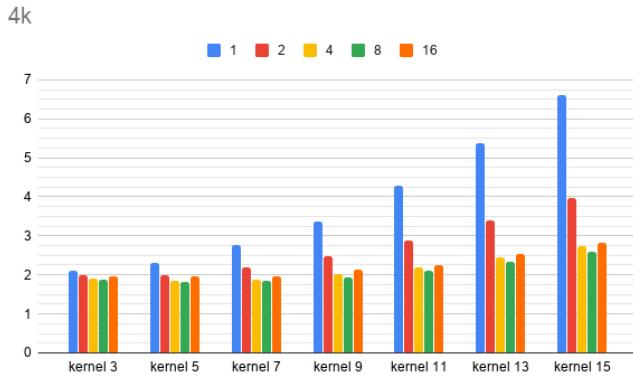


Fig. 5: Resultados previos para imagen 4k usando OMP

Se evidencia ampliamente la ventaja que supone el uso de tarjetas graficas en el procesamiento de imagenes ya que si se compara con resultados previamente obtenidos con otros metodos de paralelismo (Ver figura 5), se puede observar una mejora sustancial en los tiempos de ejecucion de cada programa, ademas teniendo en cuenta que las diferencias en el tamaño de kernel son apenas apreciables usando la GPU, se puede pensar en abordar tamaños mas grandes de kernel o procesar multiples imagenes (como un video) en un tiempo prudente haciendo uso de la tarjeta grafica, tarea que podria ser bastante limitada haciendo uso unicamente de la CPU.