

# Trabajo practico N° 3 – Ejercicio 1

Masi Juan, Perez Damián, Vildozola Mariano  
37981647, 35375255, 32254403  
Martes 19 a 23 hs, 6

<sup>1</sup>Universidad Nacional de La Matanza,  
Departamento de Ingeniería e Investigaciones Tecnológicas,  
Florencio Varela 1903 - San Justo, Argentina

**Resumen.** Se realizó la experiencia de Mean Filter (Filtro medio), realizado en Colab. Utilizando la ejecución de programa CPU-CPU y CPU-GPU. En base a esta experiencia se analizó y se obtuvieron las desventajas, ventajas y métricas de la programación secuencial contra la programación paralela.

**Palabras claves:** Python, GPU, mean filter, Cuda, Colab

## 1 Introducción

Google Colaboratory, también llamado "Colab", permite ejecutar y programar en Python directamente en el navegador. Posee ciertas ventajas: no requiere configuración, se pueden correr en la nube, es posible elegir correr en una CPU o GPU de forma gratuita, y permite compartir contenido fácilmente.

CUDA es una plataforma de computación paralela general y un modelo de programación. Puede administrar GPU, recursos de hardware y utilizar GPU para realizar computación compleja.

El programa CUDA consta de dos partes: una es el código de host que se ejecuta en la CPU y otra es el código del dispositivo ejecutándose en GPU. El programa paralelo que se ejecuta en la GPU también se conoce como kernel.

Mean filtering: El filtrado medio es un método de filtrado lineal, que reemplaza el valor de gris de píxel del centro punto de la ventana por medio de un píxel de ventana suave.

Dada una imagen  $f(i,j)$ , el procedimiento consiste en generar una nueva imagen  $g(i,j)$  cuya intensidad para cada píxel se obtiene promediando los valores de intensidad de los píxeles  $f(i,j)$  incluidos en un entorno de vecindad predefinido.

En pocas palabras, es tomar el promedio de los valores de píxeles en un área determinada en lugar del valor de píxeles original.

La máscara de filtro 3 \* 3 que se usa comúnmente es:

$$R = \frac{1}{9} \sum_{i=1}^9 z_i$$

$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

El proceso de filtrado de una imagen  $M \times N$  a través de un filtro promedio ponderado  $m \times n$  se puede dar mediante la siguiente fórmula:

$$g(x, y) = \frac{\sum_{s=-a}^a \sum_{t=-b}^b w(s, t) f(x + s, y + t)}{\sum_{s=-a}^a \sum_{t=-b}^b w(s, t)} \quad (3.5-1)$$

## 2 Desarrollo

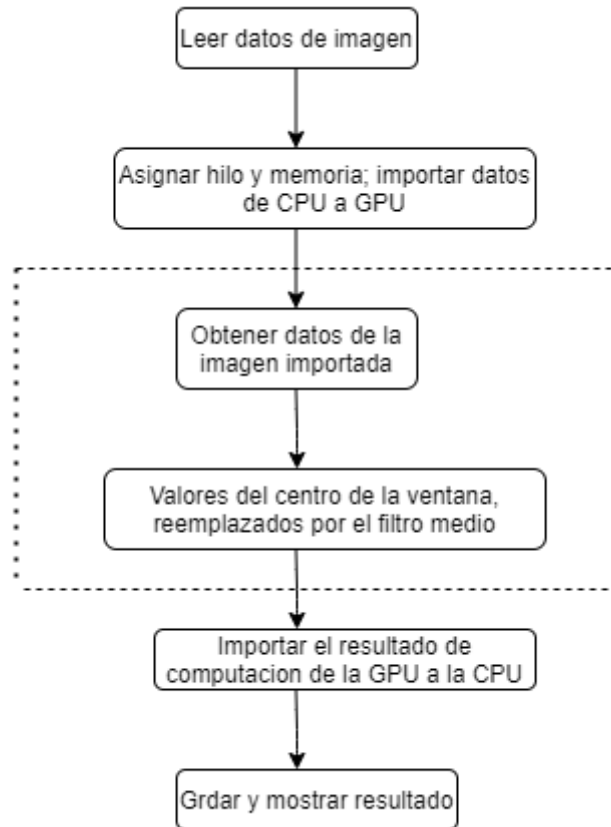
- Manual de usuario: Primero se debe ejecutar armado del ambiente y luego lectura de imagen de dominio público.

Luego se tiene que ejecutar procesamiento y filtrado de imagen utilizando CPU. Con esta ejecución se va a visualizar la imagen original y la imagen aplicando Mean Filter, utilizando CPU.

Después se tiene que ejecutar (Desarrollo GPU) Mean Filter utilizando CUDA. Con esta ejecución se va a visualizar la imagen original y la imagen aplicando Mean Filter, utilizando GPU.

- <https://github.com/juanagustinmasi/VacunarTech>
- El funcionamiento desde el punto de vista de la programación en CPU es el siguiente: se realiza la apertura de la imagen, se recorre la misma, se le aplica el filtro correspondiente y se muestra el resultado. Para realizar el recorrido de la imagen, se realiza de forma secuencial.
- El funcionamiento desde el punto de vista de la programación en GPU es el siguiente: se realiza la apertura de la imagen, luego se reserva en la memoria de la CPU y dicha porción de memoria se copia a la memoria de la GPU. Luego de acuerdo a las dimensiones de la imagen se recorre con los hilos los píxeles de la imagen, se aplica el filtro y se obtiene el resultado. Para realizar el recorrido de la imagen, se realiza con los hilos, ingresando a cada píxel de forma paralela.

- Diagrama de flujo del algoritmo de filtrado medio



Los pasos en la línea de puntos se ejecutan en el extremo del dispositivo (GPU) y otros pasos ejecutar en el extremo del host (CPU).

- Resultados de experimentos:
  - Al planificar con la mitad de los hilos-gpu, aumenta un poco el tiempo de procesamiento de la imagen en comparacion a utlizando los hilos-gpu que se tenian.
  - Al planificar con el doble de hilos-gpu, el tiempo de procesamiento baja un poco. Pero no es un tiempo de gran importancia como para decidir utlizar el doble de hilos.
  - Al planificar con la cantidad maxima de hilos por bloques soportadas por el GPGPU, se obtiene un menor tiempo que el original, pero tampoco varia mucho el tiempo de procesamiento.

### 3 Conclusiones

Comparación del tiempo de ejecución de filtro medio de un programa por CPU y programa por CUDA.

Programa	TIPO DE DEFECTO			
	Punto negro	Inclusión	Raya	Punto blanco
Tiempo de ejecución del programa CPU (ms)	4.36	4.04	4.20	4.34
Tiempo de ejecución del programa CUDA (ms)	1.28	1.09	1.26	1.11
Relación de aceleración	3.41	3.71	3.33	3.91

Analizando los resultados de la tabla podemos conocer que el programa por CUDA ejecuta a nivel mas rapido que los programas por CPU.

### 4 Referencias

<https://towardsdatascience.com/image-filters-in-python-26ee938e57d2>

<https://www.programmersought.com/article/49584514760/>

<https://www.programmersought.com/article/72128196687/>

<http://www.librow.com/articles/article-5>

<https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/iet-ipr.2013.0521>

<https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/iet-ipr.2017.0825>