# Computación paralela y distribuida
# Práctica 1

Angel Rendón, Andrés Forero
Universidad Nacional Bogotá, Colombia
Email:amrendonsa@unal.edu.co, afforeroc@unal.edu.co


Figura 1: Escudo de la Universidad.

***Resumen*—Informe de práctica que muestra la influencia del uso de hilos de paralelización de cómputo y el tamaño de kernel de procesamiento de imágen sobre el tiempo de respuesta y el speedup en una implementación del efecto borroso en tres archivos *.bmp con resolución: 720p, 1080p y 4k. El código fuente implementa hilos POSIX, memoria dinámica y estructuras de datos para el cálculo de tiempo.**

## I. Introducción

Partiendo del concepto de computación paralela se dispuso a implementar un algoritmo de efecto borroso para imágenes de alta calidad y sobre este, estudiar el efecto combinado de hilos y kernel sobre el tiempo de respuesta. Lo anterior se hace de suma importancia debido a la eficiencia en el procesamiento de imágenes que tienen como condiciones: soportar los archivos de visual de gran tamaño soportado por el hardware que disponible actualmente en el modelo de computación clásica.

Mayo 7 de 2019

## II. Efecto borroso

Cuando desenfocamos una imágen, hacemos que la transición entre colores de los pixeles que la componen sea más suave.

- Efecto borroso promedio: Toma un pixel referente y sobre un conjunto de pixeles alrededor promedia cada unos de los canales de colores de RGB para el pixel referente. El conjunto de pixeles alrededor puede varian en tamaño; esto se llama kernel. Y el promedio se realiza sobre todos los pixeles de la imágen.
- Efecto borroso gaussiano: Es similar al efecto borroso promedio pero el promedio de los canales de colores se calcula con distintos pesos en función de la cercanía del pixel referencia.

## III. Paralelización del algoritmo

La paralelización se logró primero construyendo el algoritmo de efecto borroso de forma secuencial y después aplicando el número de hilos dentro de su forma encapsulada. La forma encapsulada usa como argumentos de entradas los mismos que la función main y tiene como función nuclear void llamada parallel que es la que hace el promedio de los canales de colores para cada pixel.

## IV. Experimentos y resultados

El script itera entre los diferentes valores del kernel de procesamiento de imágen y el número de hilos para los 3 resoluciones. El programa toma estos valores como argumento paraleliza y finalmente arroja un valor de tiempo que es anexado a un documento *.csv. El archivo contiene todos los valores de tiempo.

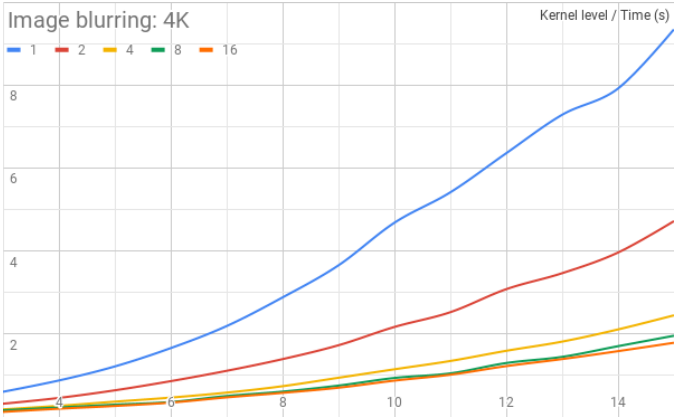


Figura 2: 4K - Tiempo de respuesta: Kernel vs Hilos.

## V. Conclusiones

- El uso de varios hilos reduce la pendiente del tiempo de respuesta. El aumento de tiempo solo depende en mayor medida es del tamaño del kernel.
- Hay una mejora en el tiempo de ejecución que depende solamente de la cantidad de hilos utilizado. Esto se nota más en imágenes de alta calidad.


Figura 3: 4K - Speedup: Kernel vs Hilos


Figura 5: 1080p - Speedup: Kernel vs Hilos

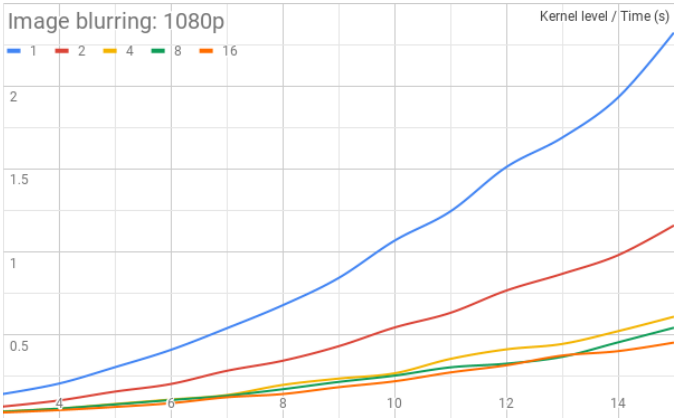


Figura 4: 1080p - Tiempo de respuesta: Kernel vs Hilos

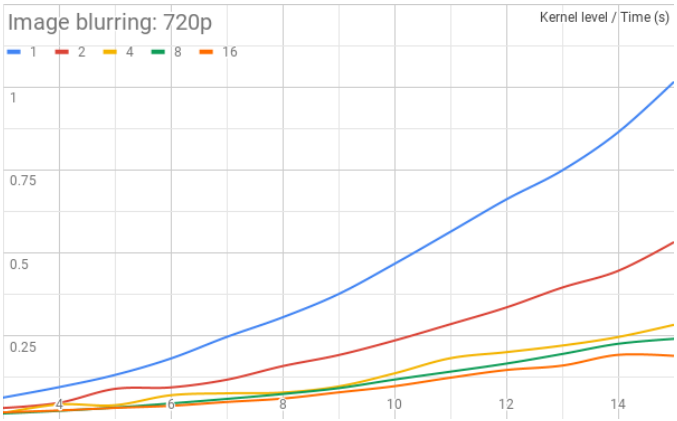


Figura 6: 720p - Tiempo de respuesta: Kernel vs Hilos

## VI. Bibliografía


- https://www.w3.org/Talks/2012/0125-HTML-Tehran/Gaussian.xhtml
- http://www.jhlabs.com/ip/blurring.html
- https://computergraphics.stackexchange.com/questions/39/how-is-gaussian-blur-implemented



Figura 7: 720p - Speedup: Kernel vs Hilos