Computación paralela y distribuida
Prácticas 1 y 2

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

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 y OpenMP, 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 el 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: los archivos visuales de gran tamaño deben soportados por el hardware que se disponible actualmente en el modelo de computación clásica.

Mayo 14 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. 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 las 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. Los valores se manipularon en hojas de cálculo para obtener las gráficas de tiempo de respuesta y de speedup. Se usó un procesador Intel(R) Core(TM) i7-6500U CPU 2.50GHz que tiene dos núcleos.

V. Conclusiones

- En cada una de las gráficas de tiempo de respuesta, se evidencia claramente el uso de los dos núcleos, con una curva azul para un solo hilo muy separado del conjunto de otros hilos denotado por los demás colores.
- El uso de varios hilos reduce la pendiente del tiempo de respuesta.
- El aumento de tiempo solo depende en mayor medida del tamaño del kernel.
- El speedup fue mejor con OpenMP para las imágenes de resoluciones 4K y 1080p.
- El speedup fue mejor con POSIX para la imágen de 720p.s

VI. Bibliografía

- https://www.w3.org/Talks/2012/0125-HTML-Tehran/Gaussian.xhtml
- http://www.jhlabs.com/ip/blurring.html
- https://bit.ly/2Hror6D
- https://intel.ly/2VOCM5K

VII. Anexos

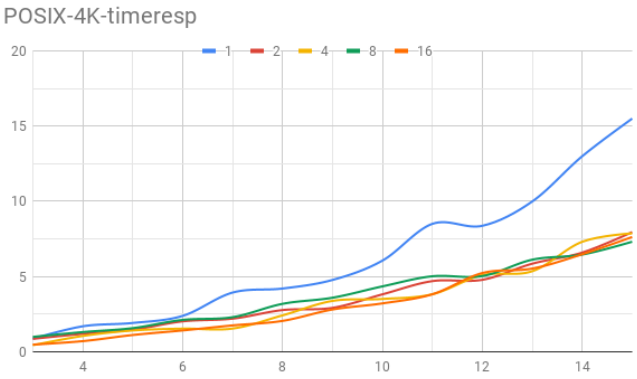

Figura 2: POSIX - 4K - Kernel vs Tiempo de respuesta

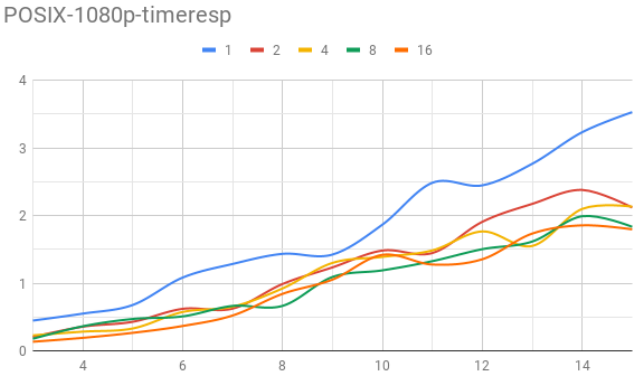

Figura 3: POSIX - 1080p - Kernel vs Tiempo de respuesta

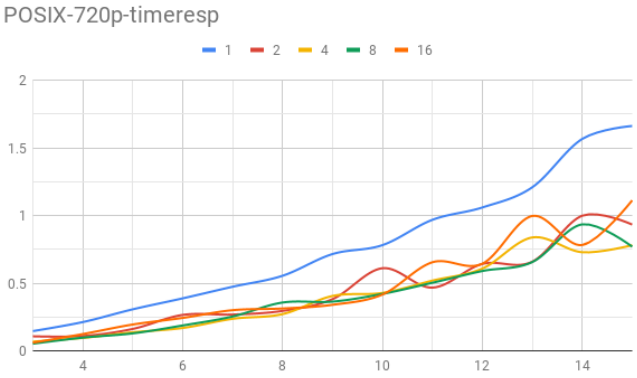

Figura 4: POSIX - 720p - Kernel vs Tiempo de respuesta

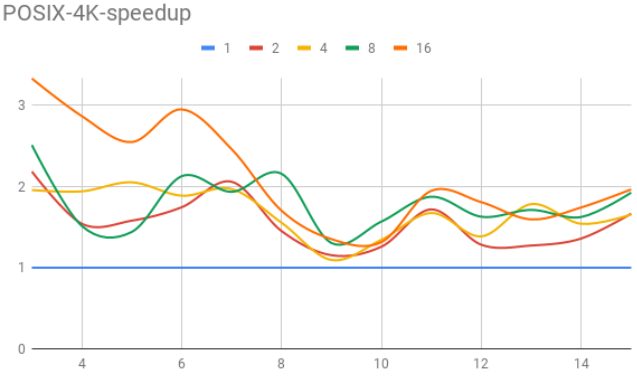

Figura 5: POSIX - 4K - Kernel vs Speedup

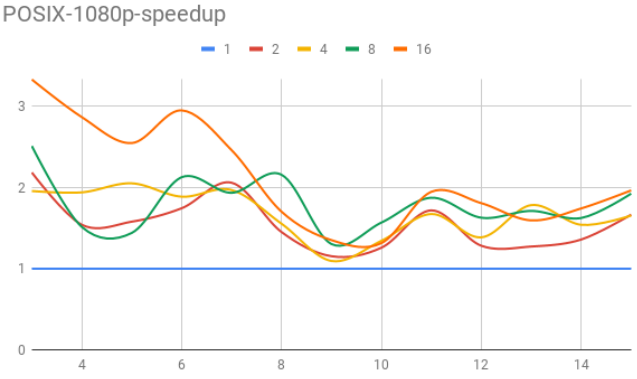

Figura 6: POSIX - 1080p - Kernel vs Speedup

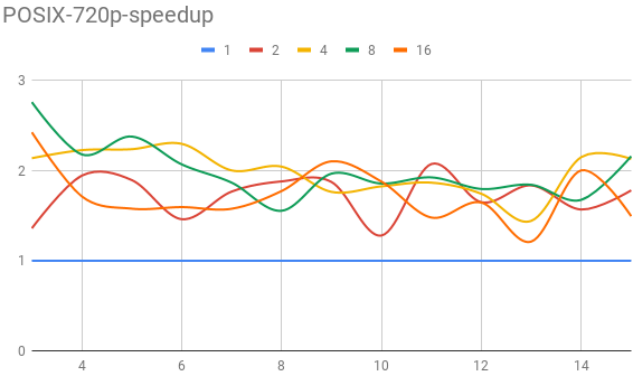

Figura 7: POSIX - 720p - Kernel vs Speedup

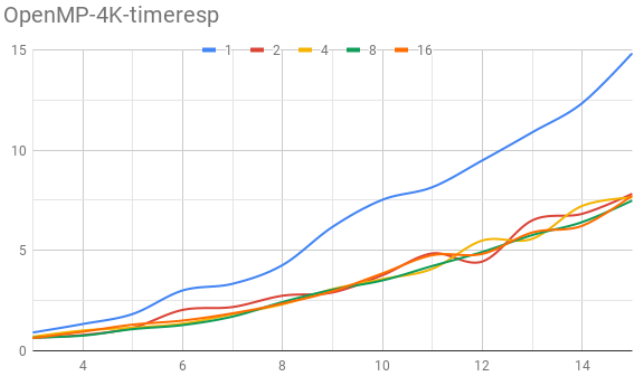

Figura 8: OpenMP - 4K - Kernel vs Tiempo de respuesta

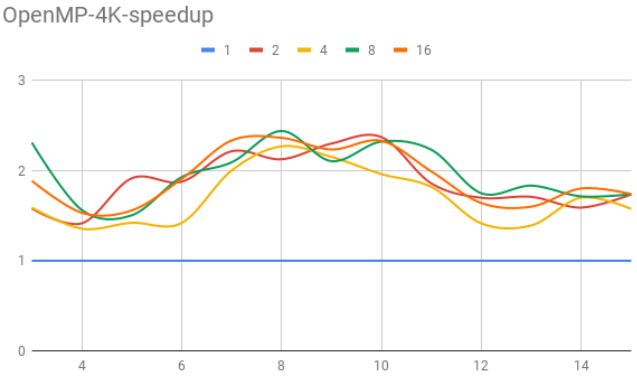

Figura 11: OpenMP - 4K - Kernel vs Speedup

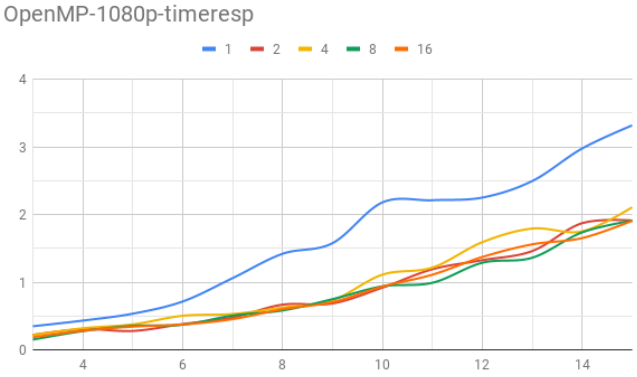

Figura 9: OpenMP - 1080p - Kernel vs Tiempo de respuesta

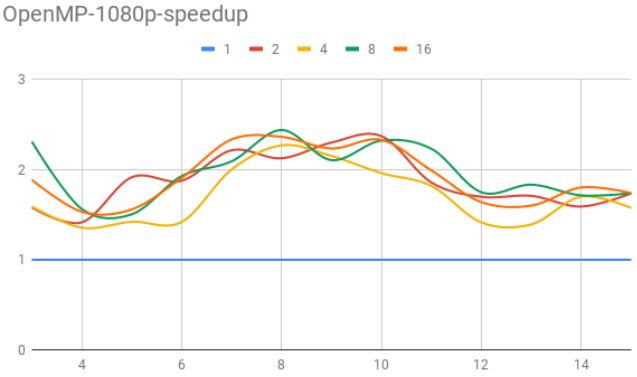

Figura 12: OpenMP - 1080p - Kernel vs Speedup

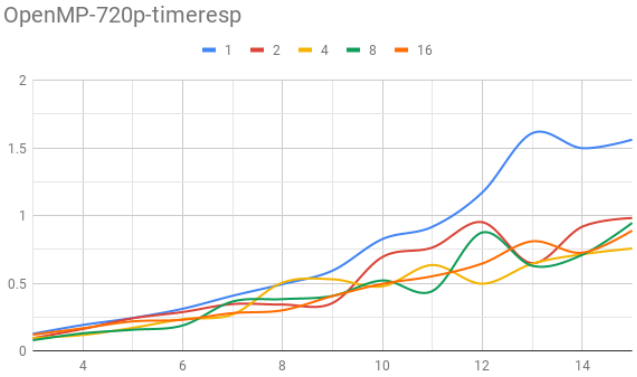

Figura 10: OpenMP - 720p - Kernel vs Tiempo de respuesta

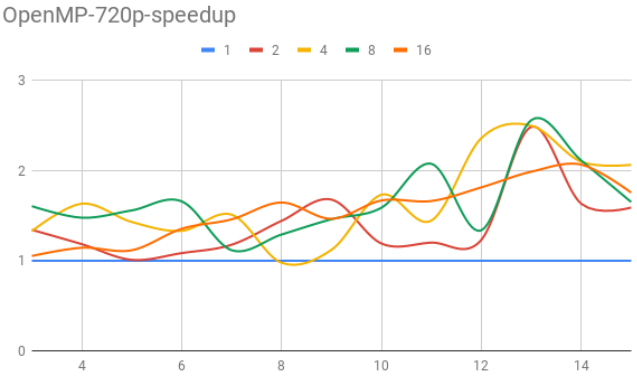

Figura 13: OpenMP - 720p - Kernel vs Speedup