Figura 1. Escudo de la Universidad.

Practica 1 - Paralela

Omar Eduardo Roa Quintero
Fernando Vargas Montero

Universidad Nacional de Colombia
Facultad de Ingeniería, Departamento de Ingeniería de Sistemas e Industrial
Bogotá, Colombia
2017

ÍNDICE

I. Antes de empezar

I-A. *Que se necesita*

Usar Python 3.*.
Instalar paquetes de python.

I-B. *OpenCV para python*

Para la instalación de OpenCV se puede seguir la guía en la pagina oficial
https://docs.opencv.org/2.4/doc/tutorials/introduction/linux_install/linux_install.html

I-C. *Pymp*

```
pip3 install pymp-pypi
```

https://github.com/classner/pymp

I-D. *Como ejecutar el programa*

Comando de ejecución:

```
python3 blur_OpenMp.py -i 720p.jpg
-bx 3 -by 3 -t 3
```

-i: Seleccionan la imagen a tratar
-bx: Cantidad de kernel en el eje X
-by: Cantidad de kernel en el eje Y
-t: Numero de hilos

II. Introducción

II-A. *Efecto borroso de una imagen en Python*

En este documento se vera la implementación del efecto borroso en una imagen utilizando como lenguaje python y el uso de hilos. Para el tratado de la imagen, se usara la librería OpenCV. Para el trato de los hilos se usara OpenMP.
Se diseñó y realizó un programa con unos metodos para aplicar el efecto borroso en una imagen dada. Este programa trabaja con varias formas de kernel (cuadrado, o en forma de çruz").

II-B. *PARALELIZACIÓN DEL ALGORITMO.*

El método de paralelización utilizado en este programa es Block Wise en 2 dimensiones.

Si por ejemplo se utilizan 4 hilos para aplicar el efecto borroso en una imagen, la imagen se divide como en la imagen anterior, es decir, a cada hilo le corresponde una sección horizontal, partida equitativamente entre el numero de hilos, y en esa parte correspondiente trata las 3 matrices de la imagen (RGB).

II-C. *EXPERIMENTOS Y RESULTADOS.*

II-C1. Gráficas con kernel igual a 3: En esta sección están las gráficas con kernel=3, ejecutado hasta 4 hilos para cada una de las 3 imágenes.

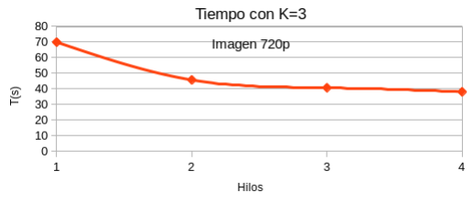

Figura 2. Tiempo para k=3 en imagen de 720p.

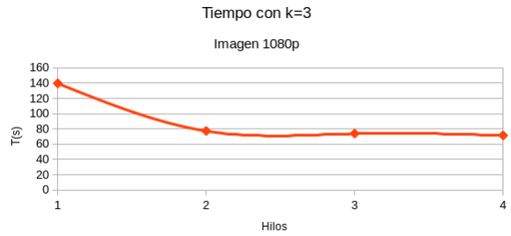

Figura 3. Tiempo para k=3 en imagen de 1080p.

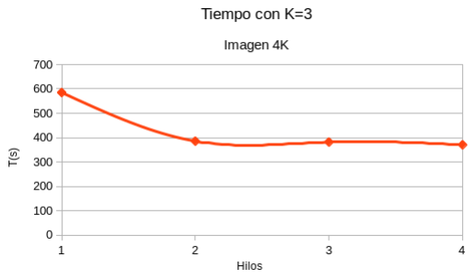

Figura 4. Tiempo para k=3 en imagen de 4K.

II-C2. Gráficas misma imagen con diferente kernel: Para esta prueba se realizaron con valores de kernel k=3 y k=5. Como se esperaba, el tiempo para procesar con kernel igual a 5 es mayor que el del kernel igual a 3.

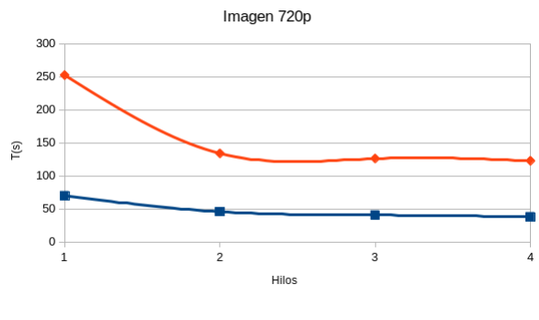

Figura 5. Linea roja: k=5,Linea azul: k=3

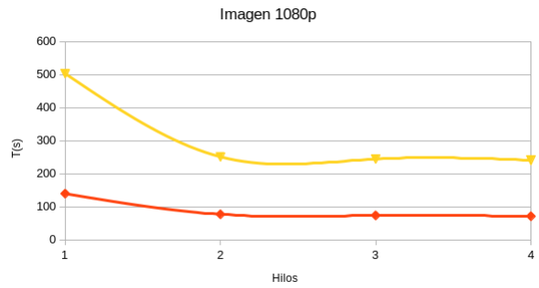

Figura 6. Linea amarilla: k=5, Linea roja: k=3

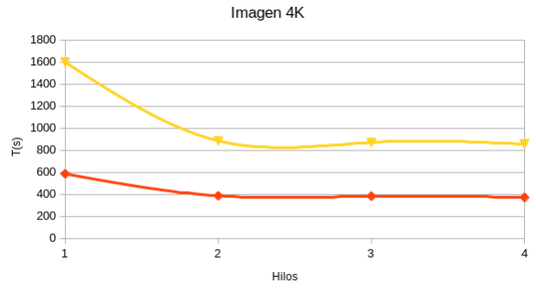

Figura 7. Linea amarilla: k=5,Linea roja: k=3

II-D. Graficas SpeedUp kernel=3

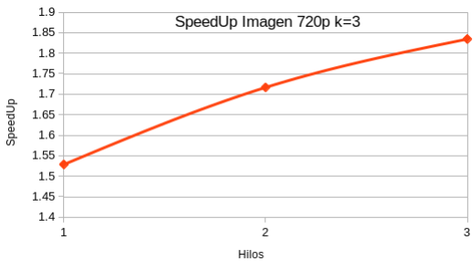

Figura 8. SpeedUp imagen 720p.

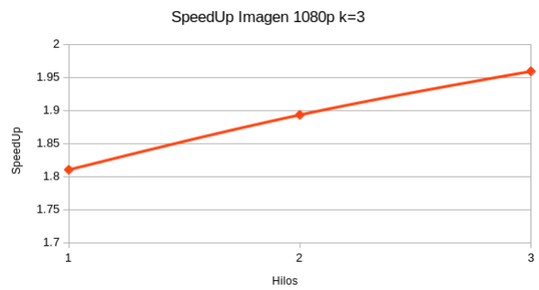

Figura 9. SpeedUp imagen 1080p.

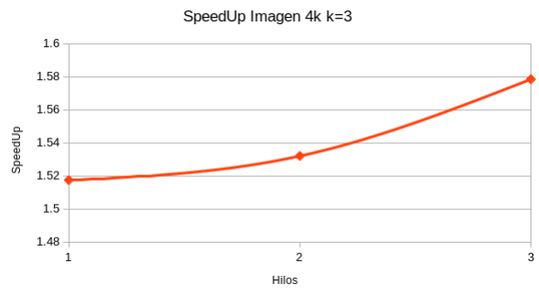

Figura 10. SpeedUp imagen 4K.

II-E. Conclusiones

Con estos resultados podemos concluir el impacto en el tiempo de ejecución al paralelizar un algoritmo o programa. Hay que tener en cuenta que al usar mas hilos mas comunicaciónn debe haber entre ellos y esto resulta en más tiempo.
La mejora del tiempo de un programa al paralelizarlo es limitado, en las gráficas podemos ver que la mejora ya no es notable al aumentar mucho la cantidad de hilos. Para verificar esto, el SpeedUp también nos da un resultado en el que su aumento, al ser mayor los hilos usados, no es notable.