Aplicacion de filtro de Sobel en imagenes en escala de grises, utilizando herramientas de paralismo (CUDA)

Juan Pablo Flórez Caicedo
Estudent
Computer Engineering
Pereira, Universidad Tecnologica de Pereira
Email: junpaflorez@gmail.com

Christhian Julián Gómez Castaño
Estudent
Computer Engineering
Pereira, Universidad Tecnologica de Pereira
Email: chjugomez@utp.edu.co

***Resumen*—El procesamiento de imagenes, es una de las mejores tecnicas para la implementacion de computación paralela, este tipo de implementacion tiene unas caracteristicas según los diferentes objetivos que se pretendan alcanzar, en este documento se encuentra los resultados obtenidos aplicando paralelismo en Cuda, con diferentes tipos de memoria, cabe resaltar que el proceso de filtrado de imagenes fue hecho con el filtro de Sobel. En el proceso de analisis de los diferentes tipos de memoria, se encontro la eficiencia del uso de la memoria constante, y de como el paralelismo ayuda en gran medida el procesamiento de los datos, acomparacion del uso de algoritmos secuenciales empleados en la CPU.**

1. Introduction

Las tarjetas gráficas o GPU de Nvidia que fueron las utilizadas en esta investigacin, contienen cuatro tipos de memoria denominadas como, memoria global, constante, compartida y registro.

Para esta investigación se decidió hacer uso de estas memorias mediante tres implementaciones del algorítmo de la convolución y el filtro de sobel sobre una imagen llevada a escala de grises. Es importante resaltar que en las tres implementaciones se está haciendo uso de la memoria de registro que contiene cada hilo de un stream multiprocessor, estos registros son usados cuando se declara variables o arreglos dentro de una kernel que no contengan un qualifier que indique la declaracin de la variable fuera de ese contexto, como por ejemplo el identificador $\_\ shared\_$.

1.1. Procesamiento digital de imágenes

El procesamiento digital de imágenes es el conjunto de técnicas que se aplican a las imágenes digitales con el objetivo de mejorar la calidad o facilitar la búsqueda de informacin, para realizar dicho procesamiento se realizan diversos filtrados en las imágenes tales como: Filtrado en el dominio de la frecuencia, Filtrado en el dominio del espacio, Filtro paso alto, entre otros, en este caso se dio uso al filtro de sobel, junto con comparaciones en computación paralela y secuencial.

Figura 1. Filtro de sobel

1.1.1. Filtro de Sobel. El filtro Sobel detecta los bordes horizontales y verticales separadamente sobre una imagen en escala de grises. Las imgenes en color se convierten en RGB en niveles de grises. Como con el filtro Laplace, el resultado es una imagen transparente con líneas negras y algunos restos de color.

1.1.2. Computación Paralela. La computación paralela es una forma de cómputo en la que muchas instrucciones se ejecutan simultáneamente, operando sobre el principio de que problemas grandes, a menudo se pueden dividir en unos más pequeños, que luego son resueltos simultáneamente (en paralelo). Para la realización de este documento el paralelismo es realizado en la GPU, en la cual el manejo de los hilos y de los diferentes nodos, ofrecen unos excelentes resultados a comparacion de la computación Secuencial.

1.1.3. Computación Secuencial. La computación secuencial es aquella en la que una acción (instrucción) sigue a otra en secuencia. Las tareas se suceden de tal modo que la salida de una es la entrada de la siguiente y así sucesivamente hasta el fin del proceso. La CPU es la encargada de hacer el procesamiento robusto de los algoritmos, pero hay casos que los procesos son mas livianos o sencillos, por lo que

Img name	SpeedUp Global	SpeedUp Share	SpeedUp Const	Total Megapixels processed
				Higth x width
Butterfly	20x	20x	27x	6.529.180
Car	17x	17x	23x	8.294.400
Cat	17x	17x	22x	8.294.400
City	10x	10x	14x	23.358.000
Control	15x	15x	20x	9.216.000
Lizard	52x	53x	70x	1.713.600
Paisaje	15x	16x	21x	9.437.184
Planet	71x	72x	92x	921.600
Thunder	15x	15x	20x	9.216.000
Wood	26x	26x	35x	4.474.613

Figura 2. Tabla de SpeedUp, las acelaraciones son correspondientes con la memoria secuencial

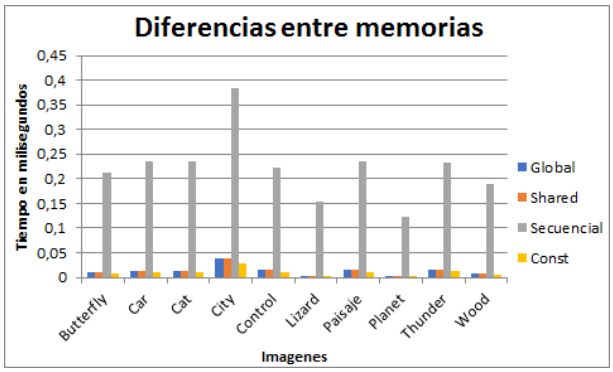

Figura 3. Grafico de tiempo entre las diferentes memorias

este trato de uno a uno, no es eficiente y para estos casos es donde se usa la computación paralela.

1.2. Speedup

El Speedup, determina la aceleracion que se obtiene al realizar una misma accion pero con diferente estrategia de diseño, para este ejercicio se hara la comparacion entre computacion secuencial y 3 tipos de computacion paralela (diferenciandosen en sus manejos de memoria), ademas se utilizaron 10 imagenes y a cada imagen se les realizo 20 pruebas, tomando sus tiempos segun cada uno de los tipos de memoria, en esta obtencion de los datos, se determino que la computacion secuencial consume mucho tiempo para realizar una accion, por lo que acontinuacion se detallara los diferentes tipos de memoria, es decir su manera de uso, su implementacion y su evaluación con respecto al tiempo empleado para ejecutarse.

1.2.1. Memoria global. Para el algorítmo que utiliza memoria global, los datos de la imagen en escala de grises reside en la memoria global así como las dos máscaras del filtro de sobel, con las que se realizará la convolucin, y es aquí donde se evidencia que existirá un problema de cuello de botella, ya que habrá que acceder a la información de las máscaras varias veces por cada pixel de la imágen, así como el acceso a los pixeles hallo al pixel que está siendo procesado, además la situación empeora si consideramos los otros hilos de los demás streaming multiprocessor que están accediendo a las máscaras.

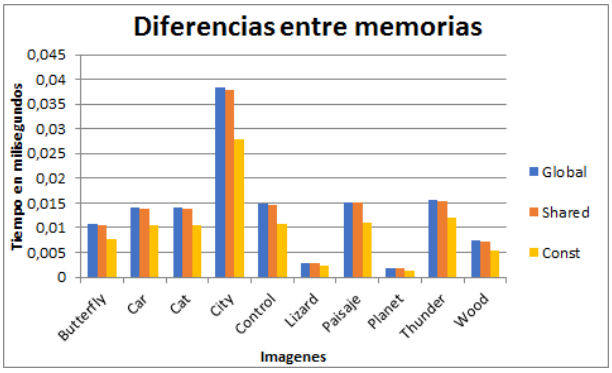

Figura 4. Grafico en donde se ven los 3 tipos de memoria representadas

1.2.2. Memoria constante. Para el algoritmo que utiliza la memoria constante, las máscaras son trasladadas a la memoria constante, se sigue usando la memoria compartida para trasladar las partes de la imagen desde la memoria global, aquí se puede visualizar que todavía persiste el problema del cuello de botella, pero al tratarse de una memoria que posee un canal de transmisin de datos más ancho de alta velocidad, y como es memoria de solo lectura el delay en el acceso se reduce notablemente en comparación con el acceso en la memoria global.

1.2.3. Memoria compartida. Para el algorítmo que utiliza memoria compartida, el problema del cuello de botella en acceso a los datos se solventa en gran medida, debido a que parte de los datos de la imagen en escala de grises se llevan primero a la memoria compartida (separada previamente dentro de la kernel), y luego las operaciones de acceso serán con respecto a ese espacio de memoria, reduciendo el acceso a la memoria global considerablemente para cada pixel y los pixeles hallo cuando estemos haciendo la convolución, las máscaras todavía residen en la memoria global, por lo que el cuello de botella en esta parte todavía existe.

2. Conclusion

El algorítmo donde se utilizó memoria global, presentó una inmensa mejoría en velocidad en comparación con la velocidad obtenida por el algorítmo secuencial(en donde se hizo uso de la herramienta opencv para su implementación). Podemos notar además que el algorítmo donde se utilizó memoria compartida, tubo aún una mayor mejoria en velocidad en comparación con el de la memoria global. Y el algorítmo donde se utilizó memoria constante obtuvo la mayor velocidad como lo muestra la grafica que el de los otros dos tipos de memoria.

Entre las razones por las cuales se presentan estas mejorías en velocidad, más hayá de que estamos trabajando con un algorítmo altamente paralelizable y así haciendo uso de many-cores, se debe a la velocidad de acceso a los datos en cada tipo de memoria, de los cuatro tipos de memoria, la global es la más lenta, y es la memoria que posee mayor probabilidad de presentar un cuello de botella,

ya que es la memoria que es accesada por cualquier hilo de cualquier stream multprocessor de la tarjeta, el segundo tipo de memoria vendría siendo la memoria compartida, esta es una memoria pequeña que está dentro de cada stream multiprocessor, y es solo accedida por lo hilos pertenecientes a ese stream multiprocessor, la velocidad de acceso a la memoria compartida es más rápida comparada con la memoria global, debido a que es una memoria que está más cerca a los hilos a nivel de hardware, y además solventa en gran medida el cuello de botella que se presenta en la memoria global, al llevar trozos de la imagen a esa memoria para un respectivo procesamiento, la siguiente memoria es la memoria constante, esta memoria es accedida por cualquier hilo de cualquier stream multiprocessor como la memoria global, pero es una memoria de mayor velocidad de acceso al tratarse de una memoria de un gran canal de transmisión de datos de alta velocidad y de solo lectura y por último tenemos la memoria de mayor velocidad de acceso que es el registro de cada hilo de un stream multiprocesor, y es la de mayor velocidad porque ese la más cercana en hardware al hilo.

Referencias

[1] David B.Kirk, Wen-Mei W Hwu. *Programming Massively Parallel Procesors*, second Edition. 255 Wyman Street, Whaltam, MA, 2013.

[2] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.