GPGPU – Laboratorio 3

En este laboratorio usaremos la GPU para procesar una imagen.
Antes de comenzar descargar el archivo comprimido desde la plataforma EVA.

El ejecutable generado se llama “blur” y recibe como parámetro un archivo de imagen en formato .pgm
Modifique el script que utilizó en el práctico anterior para encolar el trabajo en el cluster y ejecute el programa usando alguna de las imágenes de ejemplo.

El programa debería producir la imagen de salida `output_brillo.ppm` la cual debe transferir a su estación de trabajo para poder visualizarla.

Ejercicio 1

La función `ajustar_brillo_cpu(...)` recorre la imagen sumando un coeficiente entre -255 y 255 a cada píxel, aumentando o reduciendo su brillo.

a) Construir la función `ajustar_brillo_gpu(...)` y los kernels correspondientes para realizar esta tarea en la GPU.

Configure adecuadamente la grilla (bidimensional) de threads para aceptar matrices de cualquier tamaño. En el kernel, utilice las variables blockIdx y threadIdx adecuadamente para acceder a la estructura bidimensional.

El procesamiento de la imagen debe realizarse en dos versiones: una donde el acceso a memoria sea coalesced y otra donde sea completamente no coalesced (cada thread genera una transacción de 32B al acceder a memoria).

Registre el tiempo de ejecución de cada kernel y analice el ancho de banda (datos procesados por unidad de tiempo) de cada uno.

b) Genere un efecto en la imagen donde a los píxels de posición x par se les suma un valor fijo recibido por parámetro, y a los píxels de posición x impar se les resta dicho valor. Realice dos versiones de este programa: una donde existan divergencias y otra donde no existan.

Registre el tiempo de ejecución de cada kernel y analice el ancho de banda (datos procesados por unidad de tiempo) de cada uno.

Ejercicio 2

Construiremos un filtro box para difuminar una imagen en escala de grises. Consiste en sustituir el valor de intensidad de cada pixel por un promedio de los píxels vecinos. El vecindario a considerar para cada píxel es una ventana cuadrada de tamaño (2k+1)x(2k+1) centrada en el píxel, donde k es un parámetro que controlará cuánto se difumina la imagen.

a) Construya el kernel que aplica el filtro en la GPU y la función que invoca dicho kernel y transfiere los datos correspondientes hacia y desde la GPU.

b) Construya la función análoga en CPU y compare el tiempo de ejecución las variantes de CPU y GPU para distintos tamaños de imagen (para ejecutar con imágenes pequeñas puede aplicar las funciones sobre una zona de la imagen de ejemplo) y valores de k.

Entregar

El código correspondiente a cada parte y un informe en formato pdf con el análisis de los resultados experimentales.