Auxiliar 2

Stencil en OpenCL

Vicente González y Diego García

CC7515-1 — Computación en GPU

Introducción

Operaciones comunes

Map Aplicar una función a cada elemento de una estructura de datos.

Operaciones comunes

Map Aplicar una función a cada elemento de una estructura de datos.

Reduction Reducción a un solo valor mediante la aplicación de una operación binaria.

Operaciones comunes

Map Aplicar una función a cada elemento de una estructura de datos.

Reduction Reducción a un solo valor mediante la aplicación de una operación binaria.

Scan Construcción de una estructura de datos acumulativa mediante la aplicación de una operación binaria.

Operaciones comunes

Map Aplicar una función a cada elemento de una estructura de datos.

Reduction Reducción a un solo valor mediante la aplicación de una operación binaria.

Scan Construcción de una estructura de datos acumulativa mediante la aplicación de una operación binaria.

Search Búsqueda de elementos en una estructura de datos

Operaciones comunes

Map Aplicar una función a cada elemento de una estructura de datos.

Reduction Reducción a un solo valor mediante la aplicación de una operación binaria.

Scan Construcción de una estructura de datos acumulativa mediante la aplicación de una operación binaria.

Search Búsqueda de elementos en una estructura de datos

Sort Ordenación de elementos de una estructura de datos

Operaciones comunes

Map Aplicar una función a cada elemento de una estructura de datos.

Reduction Reducción a un solo valor mediante la aplicación de una operación binaria.

Scan Construcción de una estructura de datos acumulativa mediante la aplicación de una operación binaria.

Search Búsqueda de elementos en una estructura de datos

Sort Ordenación de elementos de una estructura de datos

Stencil Aplicación de un operador local a cada elemento de una estructura de datos en función de su vecindario

Operaciones comunes

Map Aplicar una función a cada elemento de una estructura de datos.

Reduction Reducción a un solo valor mediante la aplicación de una operación binaria.

Scan Construcción de una estructura de datos acumulativa mediante la aplicación de una operación binaria.

Search Búsqueda de elementos en una estructura de datos

Sort Ordenación de elementos de una estructura de datos

Stencil Aplicación de un operador local a cada elemento de una estructura de datos en función de su vecindario

Stencil

La técnica Stencil

- Es una técnica de programación que se utiliza para procesamiento de imágenes, simulaciones de fluidos, entre otras aplicaciones.

La técnica Stencil

- Es una técnica de programación que se utiliza para procesamiento de imágenes, simulaciones de fluidos, entre otras aplicaciones.
- Se basa en actualizar cada punto de una imagen o conjunto de datos en función de los valores de sus vecinos cercanos.

La técnica Stencil

- Es una técnica de programación que se utiliza para procesamiento de imágenes, simulaciones de fluidos, entre otras aplicaciones.
- Se basa en actualizar cada punto de una imagen o conjunto de datos en función de los valores de sus vecinos cercanos.
- CUDA y OpenCL son frameworks que permiten implementar esta técnica de manera paralela.

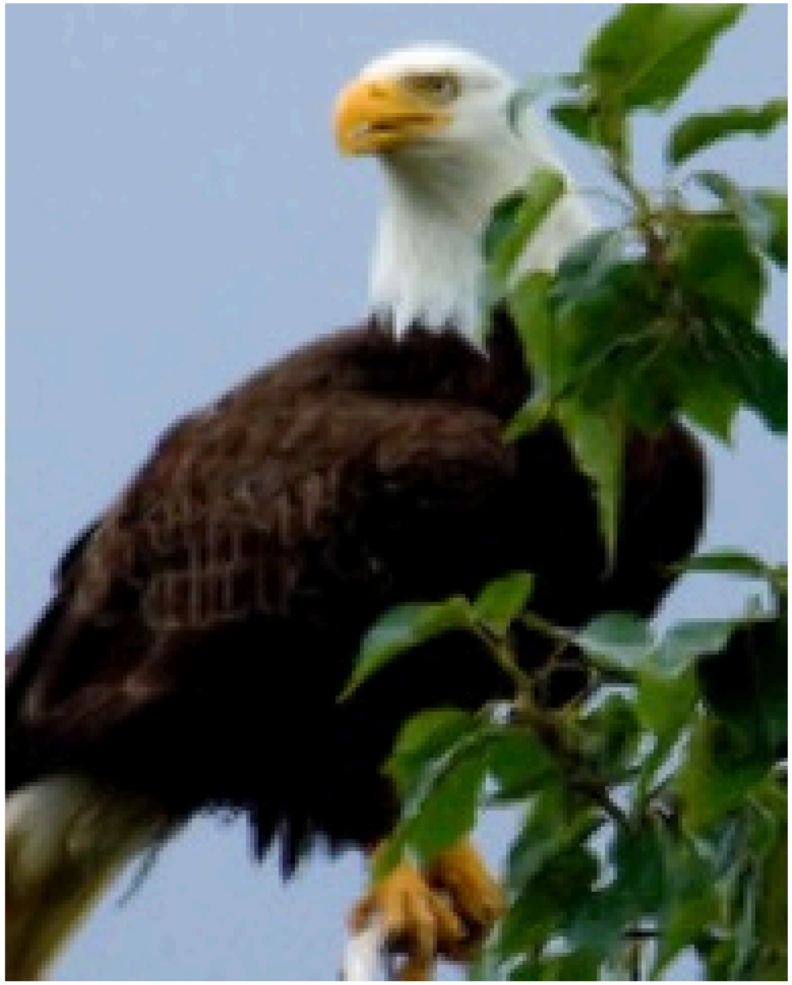
La técnica Stencil

- Es una técnica de programación que se utiliza para procesamiento de imágenes, simulaciones de fluidos, entre otras aplicaciones.
- Se basa en actualizar cada punto de una imagen o conjunto de datos en función de los valores de sus vecinos cercanos.
- CUDA y OpenCL son frameworks que permiten implementar esta técnica de manera paralela.
- Al utilizar la técnica de stencil en paralelo, se pueden obtener mejoras significativas en el rendimiento y tiempo de procesamiento de las aplicaciones que la utilizan.

Ejemplo

Input

Contour Stencil Interpolation



Ejemplo

Una convolución es un buen ejemplo de un Stencil.

Ejemplo

Una convolución es un buen ejemplo de un Stencil.

- Aplicaremos la técnica de stencil para suavizar una imagen.

Ejemplo

Una convolución es un buen ejemplo de un Stencil.

- Aplicaremos la técnica de stencil para suavizar una imagen.
- El stencil se define de la siguiente manera:

$$\begin{pmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{pmatrix}$$

Ejemplo

Una convolución es un buen ejemplo de un Stencil.

- Aplicaremos la técnica de stencil para suavizar una imagen.
- El stencil se define de la siguiente manera:

$$\begin{pmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{pmatrix}$$

- Se aplica iterativamente en cada punto de la imagen.

Ejemplo

Una convolución es un buen ejemplo de un Stencil.

- Aplicaremos la técnica de stencil para suavizar una imagen.
- El stencil se define de la siguiente manera:

$$\begin{pmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{pmatrix}$$

- Se aplica iterativamente en cada punto de la imagen.
- Al utilizar CUDA u OpenCL para implementar esta técnica de manera paralela, se pueden obtener mejoras significativas en el rendimiento y tiempo de procesamiento de la imagen.

Ejemplo

- Por ejemplo, si el stencil se coloca en el siguiente píxel:

$$\begin{pmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{pmatrix} \qquad \begin{matrix} 1 & 2 & 3 \\ 4 & X & 5 \\ 6 & 7 & 8 \end{matrix}$$

- La operación de suavizado para ese píxel sería:

$$\begin{aligned} &(1/9 \cdot 1) + (1/9 \cdot 2) + (1/9 \cdot 3) + (1/9 \cdot 4) + (1/9 \cdot X) + (1/9 \cdot 6) + (1/9 \cdot 7) + (1/9 \cdot 8) + (1/9 \cdot 9) \\ &= (X + 1 + 2 + 3 + 4 + 6 + 7 + 8 + 9)/9 \end{aligned}$$

- El resultado de esta operación es el nuevo valor del píxel X actual después de aplicar la operación de suavizado.
- Este proceso se repite para cada píxel de la imagen, utilizando el stencil para calcular el nuevo valor de cada píxel en función de los valores de sus vecinos cercanos.

Implementación

`https://github.com/Seivier/StencilCL`

(pauta en rama reference)

Experimentos

Recomendaciones

1. Calcular el tiempo de las distintas etapas por separado

Recomendaciones

1. Calcular el tiempo de las distintas etapas por separado
   - Generación de datos aleatorios

Recomendaciones

1. Calcular el tiempo de las distintas etapas por separado
   - Generación de datos aleatorios
   - Copia de datos de host a device

Recomendaciones

1. Calcular el tiempo de las distintas etapas por separado
   - Generación de datos aleatorios
   - Copia de datos de host a device
   - Ejecución del kernel

Recomendaciones

1. Calcular el tiempo de las distintas etapas por separado
   - Generación de datos aleatorios
   - Copia de datos de host a device
   - Ejecución del kernel
   - Copia de datos de device a host

Recomendaciones

1. Calcular el tiempo de las distintas etapas por separado
   - Generación de datos aleatorios
   - Copia de datos de host a device
   - Ejecución del kernel
   - Copia de datos de device a host
   - Tiempo total de toda la operación

Recomendaciones

1. Calcular el tiempo de las distintas etapas por separado
   - Generación de datos aleatorios
   - Copia de datos de host a device
   - Ejecución del kernel
   - Copia de datos de device a host
   - Tiempo total de toda la operación
2. Correr experimento más de una vez (5-10)

Recomendaciones

1. Calcular el tiempo de las distintas etapas por separado
   - Generación de datos aleatorios
   - Copia de datos de host a device
   - Ejecución del kernel
   - Copia de datos de device a host
   - Tiempo total de toda la operación
2. Correr experimento más de una vez (5-10)
3. Para calcular el speed-up:

$$s = \frac{\text{tGPU}}{\text{tCPU}}$$

Recomendaciones

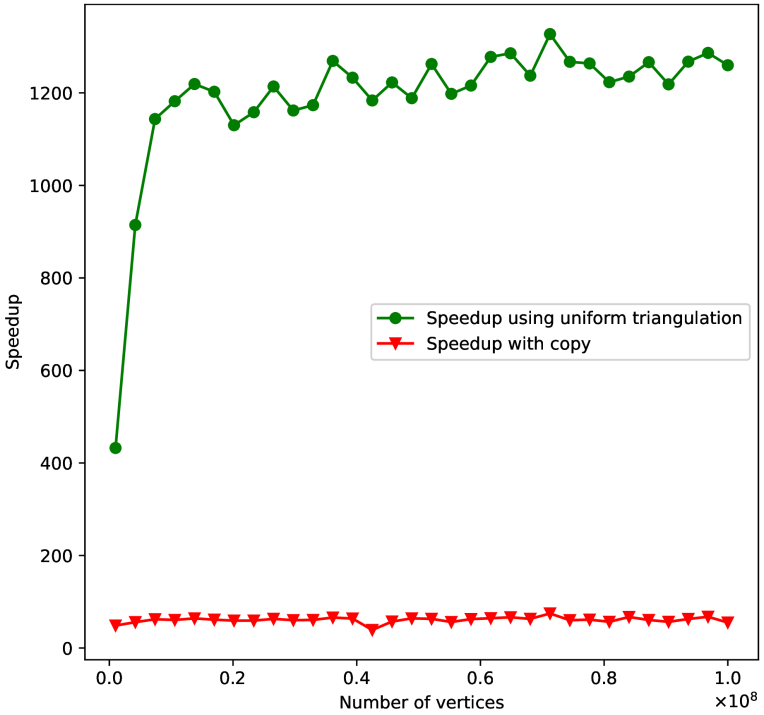


Figure 1: Un solo experimento

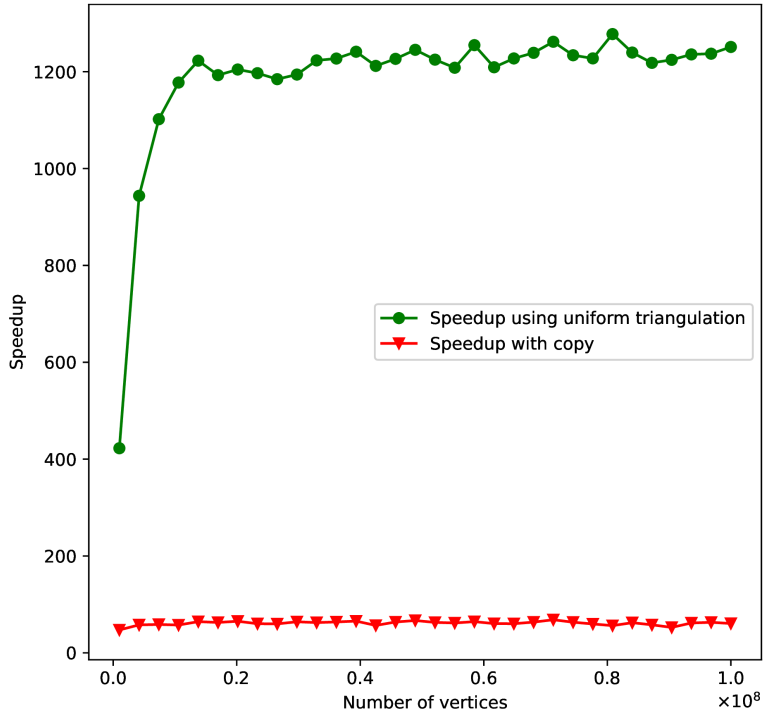


Figure 2: Promedio de 5 experimentos

Referencias

- Blog de Nvidia

  https://developer.nvidia.com/blog/using-shared-memory-cuda-cc/
- Presentación de Nvidia

  https://www.nvidia.com/docs/IO/116711/sc11-cuda-c-basics.pdf
- Experimentos GPolylla:

  https://github.com/ssalinasfe/GPolylla/tree/main/experiments
- Presentación de Sergio:

  https://www.u-cursos.cl/ingenieria/2023/1/CC7515/1/material_docente/bajar?bajar=1&id=6539157