

Tabla de Contenido

Índice de Tablas	v
Índice de Ilustraciones	vii
1. Introducción	1
1.1. Motivación	2
1.2. Objetivos	6
1.2.1. Objetivo general	6
1.2.2. Objetivos específicos	6
1.3. Metodología	6
1.3.1. Investigación	6
1.3.2. Análisis	7
1.3.3. Desarrollo	7
1.4. Contenido de la Memoria	7
2. Antecedentes	9
2.1. Paralelización: CPU vs GPU	9
2.2. Elección de tecnología para la paralelización	10
2.3. Arquitectura OpenCL	11
2.3.1. Modelo de plataforma	11
2.3.2. Modelo de ejecución	11
2.3.3. Modelo de la memoria	13
2.3.4. OpenCL vs CUDA	13
3. Análisis del Software	15
3.1. Arquitectura del Software	15
3.2. Representación del Terreno: Grilla y Triangulación	16
3.3. Algoritmos a paralelizar	17
3.3.1. Peucker	18
3.3.2. Callaghan	18
3.3.3. RWFlooding	19
3.3.4. Ángulo Diedro	21
3.4. Profiling de algoritmos a paralelizar	23
4. Diseño	27
4.1. Paralelización	27
4.1.1. Peucker	27

4.1.2.	Callaghan	28
4.1.3.	RWFlood	29
4.1.4.	Ángulo Diedro	30
4.2.	Triangulación Simplificada	31
5.	Implementación	34
5.1.	Paralelización	34
5.2.	Triangulación	38
6.	Resultados	42
6.1.	Validación de la implementación	43
6.2.	Paralelización	43
6.2.1.	Peucker	44
6.2.2.	Callaghan	45
6.2.3.	RWFlood	46
6.2.4.	Ángulo Diedro	48
6.3.	Triangulación Simplificada	49
7.	Conclusión y Trabajo Futuro	59
8.	Anexo	61
9.	Bibliografía	65

Índice de Tablas

2.1.	Diferencias de acceso entre el host y el device en las distintas regiones de memoria. . .	13
2.2.	Conceptos claves de OpenCL y su análogo en CUDA.	14
3.1.	Porcentaje de instrucciones usado en los distintos procedimientos del algoritmo de Peucker.	24
3.2.	Porcentaje de fallos de cache del algoritmo de Peucker.	24
3.3.	Porcentaje de instrucciones usado en los distintos procedimientos del algoritmo de Callaghan.	25
3.4.	Porcentaje de instrucciones usado en los distintos procedimientos de la elección de vecino y cálculo de agua acumulada del algoritmo de Callaghan.	25
3.5.	Porcentaje de fallos de cache del algoritmo de Callaghan.	25
3.6.	Porcentaje de instrucciones usado en los distintos procedimientos del algoritmo RWFlood.	25
3.7.	Insutrcciones RWFlood Determinar dirección de flujo.	25
3.8.	Porcentaje de fallos de cache del algoritmo RWFlood.	26
3.9.	Porcentaje de instrucciones usado en los distintos procedimientos del algoritmo Ángulo Diedro.	26
3.10.	Porcentaje de fallos de cache del algoritmo Ángulo Diedro.	26
4.1.	Ejemplo de flujo del algoritmo de Callaghan.	28
6.1.	Características de las unidades de procesamiento.	43
6.2.	Resultados del algoritmo de Peucker (Pro: Tiempo de ejecución promedio, Med: Mediana del tiempo de ejecución, Spdu: Speed-up, Var: Variación del tiempo de ejecución).	45
6.3.	Resultados del algoritmo de Callaghan (Pro: Tiempo de ejecución promedio, Med: Mediana del tiempo de ejecución, Spdu: Speed-up, Var: Variación del tiempo de ejecución).	46
6.4.	Resultados del algoritmo RWFlood (Pro: Tiempo de ejecución promedio, Med: Mediana del tiempo de ejecución, Spdu: Speed-up, Var: Variación del tiempo de ejecución).	48
6.5.	Resultados del algoritmo Ángulo Diedro (Pro: Tiempo de ejecución promedio, Med: Mediana del tiempo de ejecución, Spdu: Speed-up, Var: Variación del tiempo de ejecución).	49
6.6.	Resultados del algoritmo Ángulo Diedro usando distintas triangulaciones (Pro: Tiempo de ejecución promedio, Med: Mediana del tiempo de ejecución, Spdu: Speed-up, Var: Variación del tiempo de ejecución).	52

6.7. Resultados de la cantidad de triángulos en las distintas triangulaciones (# Triángulos: Cantidad de triángulos, Proporción: Proporción de triángulos respecto a la triangulación regular).	52
6.8. Resultados del tiempo de ejecución de las distintas triangulaciones simplificadas (Promedio: Tiempo de ejecución promedio, Mediana: Mediana del tiempo de ejecución).	52

Índice de Ilustraciones

1.1.	Visualización de un terreno en Runnel visto desde el eje Z.	3
1.2.	Visualización lateral de un terreno en Runnel.	3
1.3.	Visualización del resultado de la ejecución del algoritmo de Peucker en Runnel. . . .	4
1.4.	Visualización de la ejecución del algoritmo de Callaghan en Runnel.	4
1.5.	Visualización de la ejecución del algoritmo Normal Vector Similarity.	5
1.6.	Ventana de configuración de Runnel.	5
2.1.	Modelo de plataforma de OpenCL.	11
2.2.	Representación de un NDRange.	12
2.3.	Modelo de la memoria en OpenCL.	14
3.1.	Arquitectura de Runnel.	16
3.2.	Generación de la triangulación a partir de una grilla.	17
3.3.	Ejemplo de ejecución del algoritmo de Peucker.	19
3.4.	Ejemplo de inundación en el algoritmo RWFlood.	22
4.1.	Vertex-Deletion.	31
4.2.	Variables usadas en Vertex-Deletion.	32
5.1.	Configuración inicial para la ejecución de un kernel.	34
5.2.	Cálculo de memoria de entrada y salida de los kernels.	35
5.3.	Creación de buffers en la memoria del device.	35
5.4.	Paso de datos desde el host al device.	35
5.5.	Creación de kernels y asignación de sus argumentos.	36
5.6.	Configuración del tamaño del work-group.	36
5.7.	Ejecución de kernels y lectura de resultados.	36
5.8.	Liberación de la memoria de los buffers del device.	37
5.9.	Primer kernel del algoritmo de Callaghan.	37
5.10.	Segundo kernel del algoritmo de Callaghan.	38
5.11.	Triangulación simplificada.	38
5.12.	Eliminar un triángulo deshaciéndose del arco más corto.	39
5.13.	Antes de eliminar el arco, verifica que no se vaya a eliminar un punto del borde. . .	39
5.14.	Elimina un punto siempre y cuando cumpla ciertas condiciones.	40
5.15.	Primera parte de Vertex-Deletion.	40
5.16.	Segunda parte de Vertex-Deletion.	41
6.1.	Visualización del terreno de prueba.	42

6.2.	Ejecución del algoritmo de Peucker secuencial.	45
6.3.	Ejecución del algoritmo de Peucker paralelo.	46
6.4.	Ejecución del algoritmo de Callaghan secuencial.	47
6.5.	Ejecución del algoritmo de Callaghan paralelo.	47
6.6.	Ejecución del algoritmo RWFlood secuencial.	48
6.7.	Ejecución del algoritmo RWFlood paralelo.	49
6.8.	Ejecución del algoritmo Ángulo Diedro secuencial.	50
6.9.	Ejecución del algoritmo Ángulo Diedro paralelo.	50
6.10.	Zoom del resultado de la ejecución de la triangulación simplificada en un terreno.	52
6.11.	Zoom del resultado de la ejecución de la triangulación simplificada en un terreno	53
6.12.	Terreno utilizando la Triangulación Simplificada - Área Triángulo Mínima.	53
6.13.	Terreno utilizando la Triangulación Simplificada - Área Triángulo Promedio.	53
6.14.	Terreno utilizando la Triangulación Regular.	54
6.15.	Ejecución del algoritmo Ángulo Diedro usando la Triangulación Regular.	54
6.16.	Ejecución del algoritmo Ángulo Diedro usando la Triangulación Simplificada - Área Triángulo Mínima.	55
6.17.	Ejecución del algoritmo Ángulo Diedro usando la Triangulación Simplificada - Área Triángulo Promedio.	55
6.18.	Ejecución del algoritmo Normal Vector Similarity usando la Triangulación Regular.	56
6.19.	Ejecución del algoritmo Normal Vector Similarity usando la Triangulación Simplificada - Área Triángulo Mínima.	56
6.20.	Ejecución del algoritmo Normal Vector Similarity usando la Triangulación Simplificada - Área Triángulo Promedio.	57
6.21.	Ejecución del algoritmo de Peucker usando la Triangulación Regular.	57
6.22.	Ejecución del algoritmo de Peucker usando la Triangulación Simplificada - Área Triángulo Mínima.	58
6.23.	Ejecución del algoritmo de Peucker usando la Triangulación Simplificada - Área Triángulo Promedio.	58
8.1.	Kernel del algoritmo de Peucker que marca los puntos de mayor altura en una ventana.	61
8.2.	Parte 1 del kernel del algoritmo RWFlood que calcula la dirección de flujo de un punto.	62
8.3.	Parte 2 del kernel del algoritmo RWFlood que calcula la dirección de flujo de un punto.	62
8.4.	Parte 3 del kernel del algoritmo RWFlood que calcula la dirección de flujo de un punto.	63
8.5.	Parte 1 del kernel del algoritmo Ángulo Diedro que calcula el ángulo diedro entre un triángulo y sus vecinos.	63
8.6.	Parte 2 del kernel del algoritmo Ángulo Diedro que calcula el ángulo diedro entre un triángulo y sus vecinos.	64