UNIVERSIDAD TECNICA FEDERICO SANTA MARIA | DEPARTAMENTO DE INFORMÁTICA

Identificación

Libre: Computación en GPU		Modalidad y Campus: Presencial San Joaquín
Profesor: Jorge Díaz Matte	Mail: jorge.diazma@usm.cl	
Créditos SCT: 2; UTFSM: 1	Asignaturas Prerrequisitos: Arquitectura y Organización de Computadores (INF245), Lenguajes de Programación (INF253); haberlas cursado al menos.	
Hrs. Cat. Sem.: 1:10 (hora reloj)		Horario: Lunes 11-12
Cant. Sesiones: 12	Fecha Inicio: 01 - 09 - 2025	Fecha Término: 01 - 12 - 2025

Descripción

Este taller tiene como objetivo enseñar los conceptos básicos de la computación en GPU, mediante el uso de CUDA y OpenCL. Además, se busca que el estudiante pueda comprender el procesamiento de rendering mediante el uso de OpenGL, para luego adquirir la capacidad de implementar algoritmos interoperables entre GPU y OpenGL.

Requisitos de entrada

- Programación en C++.
- Conocimientos de arquitectura de computadores.
- Comprensión básica de inglés escrito.

Competencias Específicas del Perfil de Egreso a las que contribuye

- Fundamentos de Informática: Aplica los fundamentos teóricos para identificar niveles de complejidad que un problema algorítmico puede tener.
- Fundamentos de Informática: Modela y diseña algoritmos de solución para problemas de informática aplicando los fundamentos de las matemáticas discretas.

Competencias Transversales del Perfil de Egreso a las que contribuye

- Actuar con autonomía, flexibilidad, iniciativa, y pensamiento crítico al enfrentar problemáticas de la profesión.
- Incorporar una dinámica de actualización permanente de sus competencias fortaleciendo su espíritu innovador y emprendedor.

Objetivos (Resultados del aprendizaje): Al aprobar la asignatura, el estudiante será capaz de:

1. Conocer y entender las arquitecturas de las GPUs y su evolución en el tiempo.
2. Programar soluciones paralelas, para aplicaciones gráficas o de propósito general.
3. Evaluar el desempeño de soluciones paralelas.
4. Conocer aplicaciones paralelizables en GPU en distintos ámbitos.

Metodología de enseñanza y de aprendizaje

El taller consta de doce clases, de las cuales las once primeras consisten en exposición de contenidos. En cada una de estas se explicarán definiciones, modelos o tecnologías que permiten hacer uso y acelerar la ejecución de algoritmos con la GPU. En seis clases se realizarán ejercicios de programación para fortalecer el aprendizaje.

Además, se seleccionarán dos lecturas que le permitirán al estudiante complementar los contenidos vistos durante las clases. Estas lecturas serán evaluadas en la séptima sesión del taller

Para asegurar que el estudiante pueda poner en práctica y consolidar los conocimientos adquiridos en las sesiones, cada estudiante deberá desarrollar un proyecto personal que ocupe las técnicas de programación paralela vistas en clases, junto con el uso de visualización por computador. Este proyecto será presentado en la última sesión del taller, lo que permitirá a los estudiantes compartir el conocimiento adquirido durante el taller

Bibliografía

- Tolga Soyata. GPU parallel Program Development using CUDA. Computational Science Series. CRC Press. 2018.
- Hubert Nguyen. NVIDIA corporation. GPUGems 3. Addison Wesley. 2008
- Hearn, Baker, Carithers. Computer graphics with OpenGl. Fourth edition. 2011
- Cristobal Navarrro, Nancy Hitschfeld-Kahler, Luis Mateu, A Survey on Parallel Computing and its Applications in Data-Parallel Problems Using GPU Architectures, Communications in Computational Physics, 15:285-329, 2014
- Jaja, Joseph, An introduction to Parallel Algorithms, 1992. Pearson

Evaluación

- Un control de lectura, que corresponde al 30% de la nota final.
- Desarrollo de un proyecto, que corresponde al 40% de la nota final.
- La presentación del proyecto en clases, que corresponde al 30% de la nota final.

Programación del semestre

Sesión Nº	Nombre	Tipo Actividad
1	Motivación y modelos de computación paralela	- Presentación del taller - Paralelismo de tareas - Paralelismo de datos
2	Programación en CUDA I	- Presentación infraestructura CUDA - Programación en clases
3	Modelos GPU y diferencias con CPU	- Revisión de arquitecturas - Revisión de modelos paralelos
4	Programación en OpenCL	- Presentación OpenCL - Programación en clases
5	Casos de Estudio	- Mallas poligonales a partir de Delaunay - Mallas Quadtree

6	Programación en CUDA II	- Tipos de memoria - Programación en clases
7	Estrategias para diseñar algoritmos paralelos	- Problemas de Mapeo - Control de lectura
8	Shaders y OpenGL	- Proceso de rendering - Vertex Shader - Fragment Shader - Programación en clases
9	Interoperabilidad CUDA/OpenGL	- Computación interoperativa entre CUDA y OpenGL - Programación en clases
10	Interoperabilidad OpenCL/OpenGL	- Computación interoperativa entre Opencl y OpenGL - Programación en clases
11	Tensor Cores	- GPU para Deep Learning
12	Presentación Proyectos	- Actividad en clases

Elaborado por	Jorge Díaz Matte	Observaciones:
Aprobado por	José Luis Martí	
Fecha de aprobación	Agosto 2025	