Modulbezeichnung	**Massively Parallel Algorithms**
Modulverantwortliche(r)	Prof. Dr. G. Zachmann
Modulart	Pflicht/Wahl ☐ Wahlpflicht ☒
Spezialisierungsbereich	Automatisierung und Robotik, Systemsoftware / Eingebettete Systeme, Raumfahrt-Systemtechnik
Dauer des Moduls	1 Semester
Kreditpunkte	6 CP
Arbeitsaufwand	Berechnung des Workloads Präsenz 56 h Übungsbetrieb/Prüfungsvorbereitung 124 h Summe 180 h
Turnus des Moduls	i. d. R. angeboten alle 2 Semester
Voraussetzung für die Teilnahme	Keine ☐ Folgende: Inhaltliche Voraussetzungen: Algorithmisches Denken. Gewisse Programmierfähigkeiten in C (empfohlen wird das "Propädeutikum C/C++")
Lehr- und Lernformen	Seminar ☐ Vorlesung ☒ Tutorium ☒ Praktikum ☐ Projekt ☐

Lernziele	Die Ära der single-core Prozessoren ist zu Ende. Inzwischen gibt es neue, massiv-parallele Prozessoren (GPUs), die hunderte bis tausende von Threads parallel abarbeiten können. Diese entwickeln sich zur Zeit als Co-Prozessoren, die große Teile der Berechnung den (multi-core) CPUs abnehmen. Möglicherweise werden sich GPUs als neue Architektur für die Haupt-Prozessoren – gerade auch auf mobilen Geräten – etablieren, da diese mehr Compute-Power pro Energieeinheit bieten. Die große Zahl von parallelen Cores stellt das Design von Algorithmen und Software allerdings vor neue Herausforderungen, damit diese von der großen Parallelität profitieren können. Das Hauptziel dieser Vorlesung ist es, Studenten in die Lage zu versetzen, für solch massiv-parallele Hardware Algorithmen zu entwerfen. Simulation wird inzwischen gemeinhin als die dritte Säule der Wissenschaft angesehen (neben den Experimenten und der Theorie). In der Simulation wird ein ständig wachsender Bedarf an Rechenleistung benötigt; gerade diese wird aber durch die Verfügbarkeit von GPUs fast schon zu einer Commodity auf dem Desktop. Daher gibt es viele wissenschaftliche Bereiche, in denen Studenten das Wissen, das sie in dieser Vorlesung erwerben, gewinnbringend einsetzen können, wie z.B.: • Computer science (e.g., visual computing, database search) • Computational material science (e.g., molecular dynamics simulation) • Bio-informatics (e.g., alignment, sequencing, . . .) • Economics (e.g., simulation of financial models) • Mathematics (e.g., solving large PDEs) • Mechanical engineering (e.g., CFD and FEM) • Physics (e.g., ab initio simulations) • Logistics (e.g. simulation of traffic, assembly lines, or supply chains) Am Ende dieser Vorlesung werden Studenten • aktive Erfahrungen bei der Entwicklung von Software und Algorithmen für massiv-parallele Architekturen gesammelt haben; • eine Anzahl von massiv-parallelen Algorithmen-Patterns kennen; • in der Lage sein, eigene massiv-parallele Algorithmen zu entwickeln; • CUDA kennen. In der ersten Hälfte der Vorlesung werden Studenten sich anhand von kleinen und mittelgroßen Übungen und Frameworks mit der parallelen Programmier-Umgebung CUDA vertraut machen. In der zweiten Hälfte werden Studenten an einem eigenen Projekt arbeiten.
Lerninhalte	Diese Vorlesung führt Studenten in die grundlegenden und einige fortgeschrittene Methoden und Techniken der massiv-parallelen Algorithmen ein. Einige der vorgesehenen Themen sind: • die Programmierumgebung CUDA C; • die Speicher-Hierarchie und verschiedene Speicher-Charakteristiken; • die GPU Architektur • parallele Reduktion; • coalesced memory access; • massiv-parallele Matrix-Algorithmen; • Prefix-Sum und deren Anwendungen in der Bildverarbeitung; • Textur-Filterung; • Paralleles Sortieren (odd-even, bitonic, adaptive bitonic); • Bildverarbeitung; • Thrust;
Prüfungsformen	i.d.R. Bearbeitung von Übungsaufgaben und Fachgespräch oder mündliche Prüfung

Literatur	• Jason Sanders, Edward Kandort: CUDA by Example. Addison-Wesley, Pearson Education. • Wen-Mei W. Hwu: GPU Computing Gems Jade Edition. Morgan Kaufmann. • David B. Kirk, Wen-Mei W. Hwu: Programming Massively Parallel Processors. Morgan Kaufmann. • NVidia: CUDA C Programming Guide.