

# Bachelor- & Masterarbeiten

offene Themen

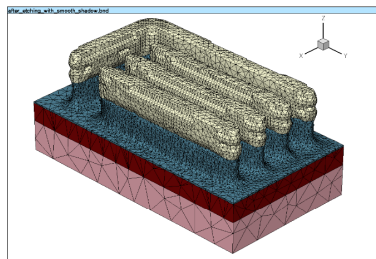
Friedrich-Alexander-Universität Erlangen-Nürnberg

Lehrstuhl für Informatik 3 (Rechnerarchitektur)

# Parallelisierung von Algorithmen für TCAD-Simulationen

Das Technology Computer-Aided Design (TCAD) beschäftigt sich mit Herstellungsprozessen in der Halbleiterindustrie, z.B. der Simulation des Einflusses der Photomaskentopographie auf die Lithographie. Die Algorithmen an sich sind gut parallelisierbar und teilweise bereits (seriell) implementiert.

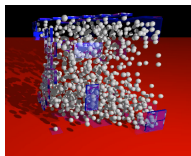
- Betreuung durch Fraunhofer IISB und Lehrstuhl Informatik 3
- Parallelisierung z.B. auf Multicores, GPGPUs oder Clustern
- Sprache: C/C++ bzw. Python



Kontakt: Dietmar Fey ([dietmar.fey@informatik.uni-erlangen.de](mailto:dietmar.fey@informatik.uni-erlangen.de))  
Eberhard Bär ([eberhard.baer@iisb.fraunhofer.de](mailto:eberhard.baer@iisb.fraunhofer.de))

Smartphones sind heute mit erstaunlich leistungsfähigen Prozessoren ausgestattet. Auch wenn die Prozessoren sich nicht wirklich mit den Multicores in Desktops messen können, so wäre doch interessant, wie groß der Unterschied wirklich ist und wie es sich mit der Performance pro Watt verhält.

- 1 Portierung der LibGeoDecomp in eine native Android-Anwendung.
- 2 Ausmessen einiger Smartphones mit Hilfe der in der Bibliothek verfügbaren Beispiel-Simulationen.



Kontakt: Andreas Schäfer ([andreas.schaefer@informatik.uni-erlangen.de](mailto:andreas.schaefer@informatik.uni-erlangen.de))

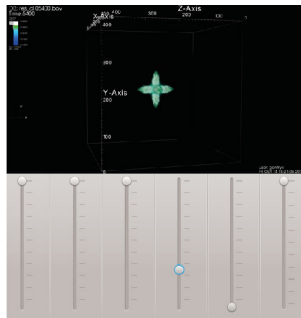
# Live Steering von Computer Simulationen

Große Simulationsläufe benötigen oft viele Tage, selbst auf Großrechnern. Ärgerlich, wenn ein Lauf wiederholt werden muss, nur zu Beginn ein Output-Parameter falsch gewählt wurde.

- Live-Visualisierung mit LibGeoDecomp und libsim
- ändern von Output-Params durch User (remote)
- (dito für Modell-Parameter?)

Voraussetzungen: C++

Ausrichtung: Programmieren (90 %)  
Recherche (10 %)



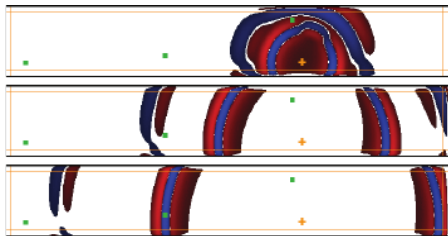
Kontakt: Andreas Schäfer ([andreas.schaefer@informatik.uni-erlangen.de](mailto:andreas.schaefer@informatik.uni-erlangen.de))

# Seismic Wave Propagation

Die Simulation der Wellenausbreitung im Untergrund ist ein ebenso wichtiges wie rechenintensives Problem, welches z.B. bei der Suche nach Ölvorkommen auftritt. Ziel der Arbeit ist es, einen existierenden Kernel (Seismic\_CPML) mit der LibGeoDecomp zu verbinden.

**Voraussetzungen:** C++, etwas Fortran

**Ausrichtung:** Programmieren (70 %),  
Leistungsmessung (30 %)



Kontakt: Andreas Schäfer ([andreas.schaefer@informatik.uni-erlangen.de](mailto:andreas.schaefer@informatik.uni-erlangen.de))

# Auto-tuning für Stencil Codes

In unserer Simulationssoftware müssen viele Parameter in Abhängigkeit vom Rechner gewählt werden. Beispiele hierfür sind:

- Blockgrößen der Daten vs. Caches
- Pinning der Threads vs. CPU-Architektur
- MPI-Kommunikation vs. PCIe-Anbindung

All diese Parameter lassen sich in ein abstraktes Modell verpacken und die Optimierung zur Laufzeit lässt sich via lokale Suche bewerkstelligen. Dies soll in der Arbeit umgesetzt werden. Modell:

- $V = \{v_1, v_2, \dots, v_n\}$  Suchraum,  $v_i$  entspricht einem Parametersatz
- $f(v) : V \rightarrow \mathbb{R}$  Kostenfunktion (Ausführungszeit)
- $G = (V, E)$  Graph für lokale Suche

**Voraussetzungen:** C++  
Mathe

**Ausrichtung:** Programmieren (70 %),  
Recherche (30 %)

Kontakt: Andreas Schäfer ([andreas.schaefer@informatik.uni-erlangen.de](mailto:andreas.schaefer@informatik.uni-erlangen.de))

# Automatische Optimierung von Compiler-Flags

Heutige Compiler bieten eine unüberschaubare Fülle an Flags um die Codeerzeugung zu steuern. Hinzu kommt, dass im High Performance Computing die Leistung der Programme stark von Parametern wie der Blockgröße oder dem Alignment der Daten abhängt. Oft lassen sich diese Parameter nur zur Compile-Zeit (z.B. via `#define`) festlegen. Projekte wie BLAS lassen daher schon während des Builds Benchmarks laufen, um optimale Parameter für den gegebenen Prozessor zu bestimmen.

Ziel der Arbeit ist es, auf Basis von genetischen Algorithmen ein Programm zur Parameter-Optimierung von Builds zu schreiben. Das fertige Programm soll dann direkt in das Buildsystem unserer Bibliothek LibGeoDecomp integriert werden.

**Voraussetzungen:** Skripting-Sprache (z.B. Ruby, Python)  
mathematische Optimierung

**Ausrichtung:** Programmierung (60 %)  
mathematische Modellierung (40 %)

Kontakt: Andreas Schäfer ([andreas.schaefer@informatik.uni-erlangen.de](mailto:andreas.schaefer@informatik.uni-erlangen.de))

# Parallelisierung von Wavefront-Algorithmen auf GPUs

In einigen Industrieprojekten, aber auch in der Forschung, sind wir auf Verfahren gestoßen, die man als Wavefront-Algorithmen formulieren lassen kann. Hierbei läuft die Berechnung in Form einer Wellenfront über eine 2D-Matrix (Beispiele: Pfadplanung in Robotersystemen, optische Oberflächenvermessung).

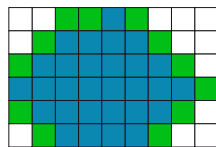
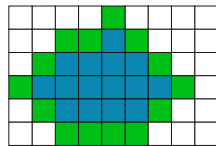
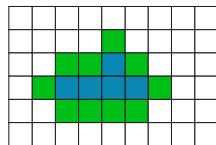
Ziel der Arbeit ist es, eine generische Parallelisierung für solche Probleme auf Grafikkarten zu erstellen. Die Schwierigkeit dabei ist, dass die Form der Wellenfront unregelmäßig ist. Die Ergebnisse dieser Arbeit können an unserem Robosoccer-System erprobt werden.

**Voraussetzungen:** C++, CUDA oder OpenCL

**Ausrichtung:** Programmierung (60 %)  
Code-Optimierung (40 %)

Kontakt: Ralf Seidler ([ralf.seidler@informatik.uni-erlangen.de](mailto:ralf.seidler@informatik.uni-erlangen.de))

Andreas Schäfer ([andreas.schaefer@informatik.uni-erlangen.de](mailto:andreas.schaefer@informatik.uni-erlangen.de))



# Cactus/LibGeoDecomp Integration

Cactus ist ein weit verbreitetes Toolkit für physikalische Simulationen. Eine Vielzahl von Plugins kann direkt für die verschiedensten Modelle eingesetzt werden. Eine besondere Klasse sind dabei die Driver. Diese Plugins sorgen für die Parallelisierung. Diese Arbeit soll untersuchen, inwieweit sich unsere Bibliothek LibGeoDecomp als Driver in Cactus einbringen ließe, damit Cactus-Nutzer von LibGeoDecomps flexibler Parallelisierung profitieren können. Das Ergebnis soll ein prototypischer Driver, der als Interface zwischen beiden dient, sein.



**Voraussetzungen:** C, C++

**Ausrichtung:** Recherche, Code analysieren (70 %)  
Programmieren (30 %)

**Kontakt:** Andreas Schäfer ([andreas.schaefer@informatik.uni-erlangen.de](mailto:andreas.schaefer@informatik.uni-erlangen.de))

Der Shack-Hartmann Sensor ist ein Messapparat zur Bestimmung von optischen Wellenfronten. Dabei ist die Rekonstruktion der Welle aus den Messdaten ein komplexer und rechenaufwändiger Schritt.

In dieser Arbeit soll untersucht werden, wie sich eine solche Rekonstruktion auf einem FPGA mittels eines Soft-IP Prozessor-Arrays umsetzen lässt und wie gut die Performance dafür ist.

**Voraussetzungen:** C, VHDL, Grundlegendes Verständnis von Hardwarebeschreibung

**Ausrichtung:** Programmierung (50 %)  
Synthese/Optimierung (50 %)

Kontakt: Ralf Seidler( [ralf.seidler@informatik.uni-erlangen.de](mailto:ralf.seidler@informatik.uni-erlangen.de))

Marc Reichenbach( [marc.reichenbach@informatik.uni-erlangen.de](mailto:marc.reichenbach@informatik.uni-erlangen.de))

# Marching Pixels Algorithmen auf GPUs

Marching Pixels Algorithmen basieren auf zellulären Automaten und wurden entwickelt, um voll parallel die Mittelpunkte, sowie die Orientierung einer beliebig hohen Anzahl von Objekten innerhalb eines Bildes zu bestimmen. Ziel der Arbeit ist es, einige der Marching Pixels Algorithmen mittels CUDA oder OpenCL auf Grafikkarten zu portieren. Dabei soll im speziellen auf die Möglichkeiten aktueller Hardware Bezug genommen und Vergleiche über die Leistungsfähigkeit bei der Verwendung von mehreren Grafikkarten erstellt werden.

**Voraussetzungen:** C/C++, OpenMP, CUDA oder OpenCL

**Ausrichtung:** Programmierung (50 %)

Profiling/Code-Optimierung (50 %)

Kontakt: Ralf Seidler( [ralf.seidler@informatik.uni-erlangen.de](mailto:ralf.seidler@informatik.uni-erlangen.de))

