

# CUDA Workshop

## Einführung

Daniel Tenbrinck

Computer Vision and Pattern Recognition Group

Institut für Informatik

Westfälische Wilhelms-Universität Münster

03.Juli 2009

# Worum es in diesem Workshop geht:

## Was ist CUDA?

- CUDA = Compute Unified Device Architecture
- SDK im Februar 2007 von Nvidia veröffentlicht
- **Ziel:** Beschleunigung wissenschaftlicher und technischer Berechnungen
- **Idee:** Nutzen der spezialisierten parallelen Architektur moderner GPUs  
→ Rechnen auf Grafikkarten!







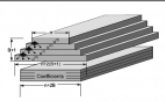
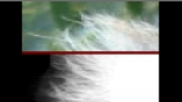


## Populäre Nutzer von CUDA:

- folding@home (Faltung von Proteinen)
- SETI@home (Suche nach außerirdischen Signalen)
- Photoshop CS4 (Bildbearbeitung)

# Gründe für einen CUDA-Workshop:

## Warum lohnt sich CUDA?

- fast alle aktuellen Rechner besitzen leistungsstarke Grafikkarte  
→ kein Neukauf von Hardware notwendig!
- viele Berechnungen lassen sich parallelisieren  
→ Beschleunigung von bis zu **Faktor 2300** in Einzelfällen
- Verteiltes Rechnen besitzt enormes Zukunftspotential

 <p>Fast and Exact Solution of Total Variation Models on the GPU</p> <p>1000 x</p>	 <p>Duality TV-L1 Flow with Fundamental Matrix Prior</p>	 <p>CUDA Fluid Simulation in NVIDIA PhysX</p> <p>10 x</p>	 <p>IMPLEMENTING ALGORITHMS FOR SIGNAL AND IMAGE RECONSTRUCTION ON GPU</p> <p>169 x</p>
 <p>Accelerating Stencil-Based Computations by Increased Temporal Locality on Modern GPUs</p>	 <p>TVSeg - Interactive Total Variation Based Image Segmentation</p>	 <p>SETI@home</p> <p>10 x</p>	 <p>Real Time Elimination of Undersampling Artifacts in CE-MRA using Variational</p> <p>2300 x</p>

## I. Grundlagen der Parallelisierung

- Warum parallelisieren?
- Wie parallelisiere ich Programme?

## II. Grafikkartenarchitektur

- Wie sind Grafikkarten aufgebaut?
- Wie kann man auf der GPU parallelisieren?

## III. Grundlagen von CUDA

- Was benötigt man um CUDA zu programmieren?
- Was sind sprachtypische Elemente von CUDA?
- Wie funktioniert ein CUDA-Programm?

PAUSE

## IV. Programmierbeispiele und Implementierung

- Was kann man mit CUDA realisieren?
- Wie sieht ein CUDA-Programm im Detail aus?

## V. OSEM-Bildrekonstruktion mit CUDA

- Wie lässt sich der LM OSEM Algorithmus parallelisieren?
- Welche Ergebnisse resultieren hieraus?

## VI. Volumenrendering mit CUDA

- Was ist GPU-based Raycasting?
- Welche Vorteile bietet die Implementierung mit CUDA?
- Wie kann man noch weiter optimieren?

## VII. Ausblick

- Was ist OpenCL?
- CUDA und Matlab
- CUDA in mobilen Endgeräten

**Viel Spaß mit CUDA!**