CUDA N-Body Simulation

Daniel Noske, Theodor Wübker, Leonard Franke

27. Juli 2024

Übersicht

Projektidee

- N-Body Simulation
- Asteroiden beliebig in Fenster platzieren
- Gravitationskräfte und Kraftfelder wirken auf Körper
- $O(n^2)$ Aufwand $\rightarrow$ Parallele Berechnung für jeden Asteroid

Projektidee

- Cuda Kernelfunktionen ermöglichen effiziente parallele Berechnungen
- Ziel: Optimierung der Kernelrechenleistung

Umsetzung

- Programmiersprache CUDA C $\rightarrow$ C++ Dialekt
- Qt5 zum Rendern
- Buildtool CMake
- CUDA Kernels für Physiken der Asteroiden

Implementierung

- C++ Teil $\leftrightarrow$ CUDA Schnittstelle
- Aufteilung $\rightarrow$ neue Versionen des Kernels gut einbindbar

Ausführung

- Demo...

CUDA Kernel V1

- Kopieren des Asteroidenvektors auf die GPU vor jedem Durchlauf
- Einen CUDA Thread für alle Asteroiden
- Jeder Thread berechnet Kräfte die auf seinen Asteroiden wirken

CUDA Kernel V2

- Nur Asteroiden Vektor auf GPU kopieren, wenn sich etwas geändert hat
- $\rightarrow$ immense Einsparungen

CUDA Kernel V3

- Aufteilung der Interaktionen in Tabelle

- Jede Interaktion mit einzelnem Thread parallel berechnen

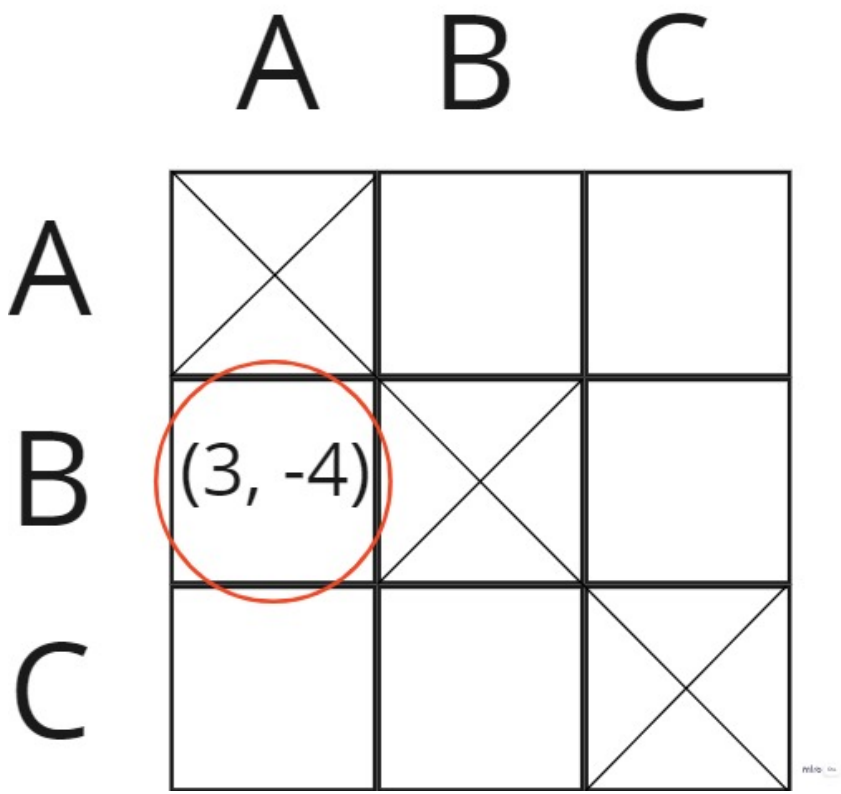

Abbildung: Paarweise Interaktionen der Asteroiden

Messvorgehen

- Version und Asteroidenanzahl vom Nutzer übergeben
- Zeitmessung vor und nach dem Kernelaufruf
- CPU-Version zum Vergleich

Endergebnis

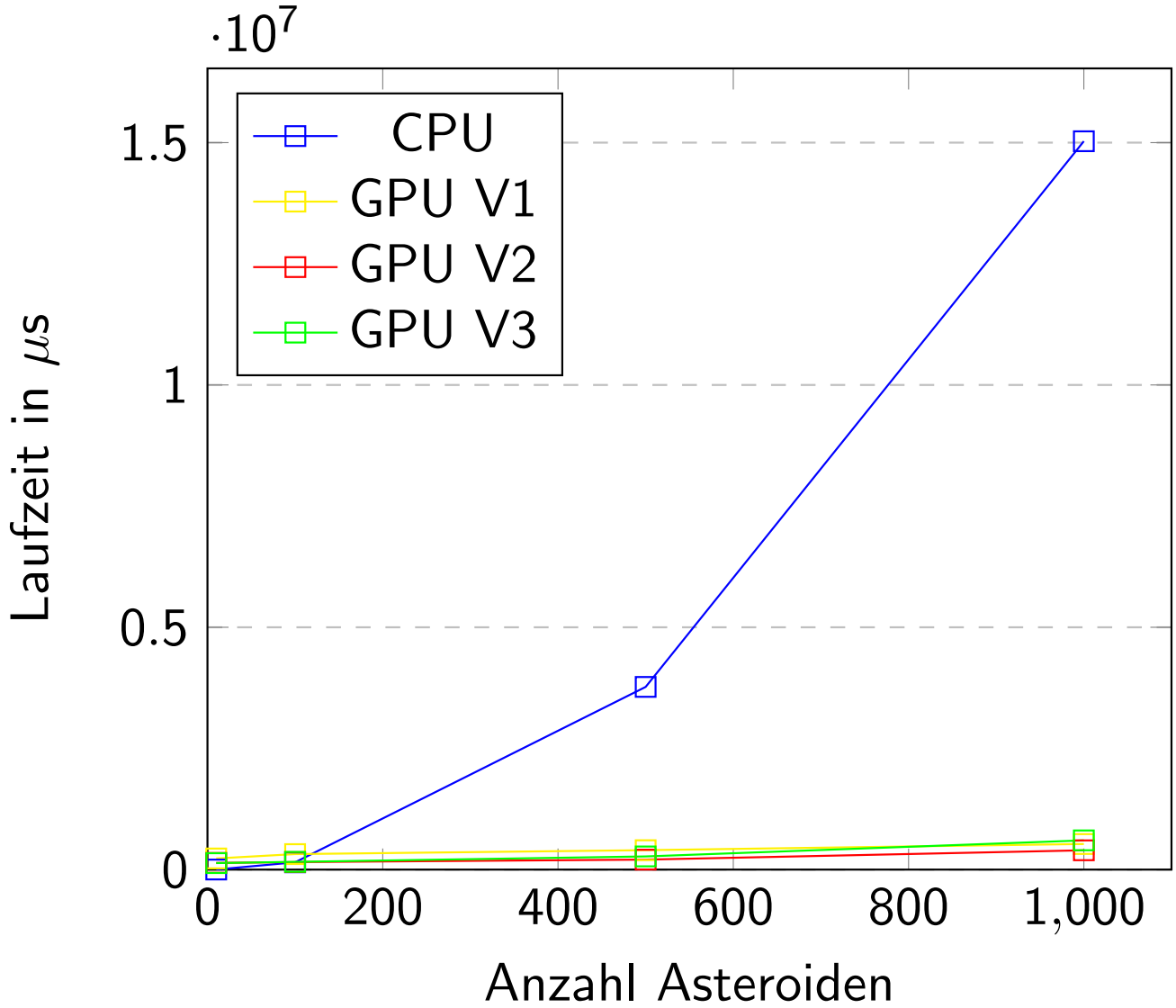

Abbildung: Laufzeit der CPU-Version für verschiedene Asteroidenanzahlen

Endergebnis

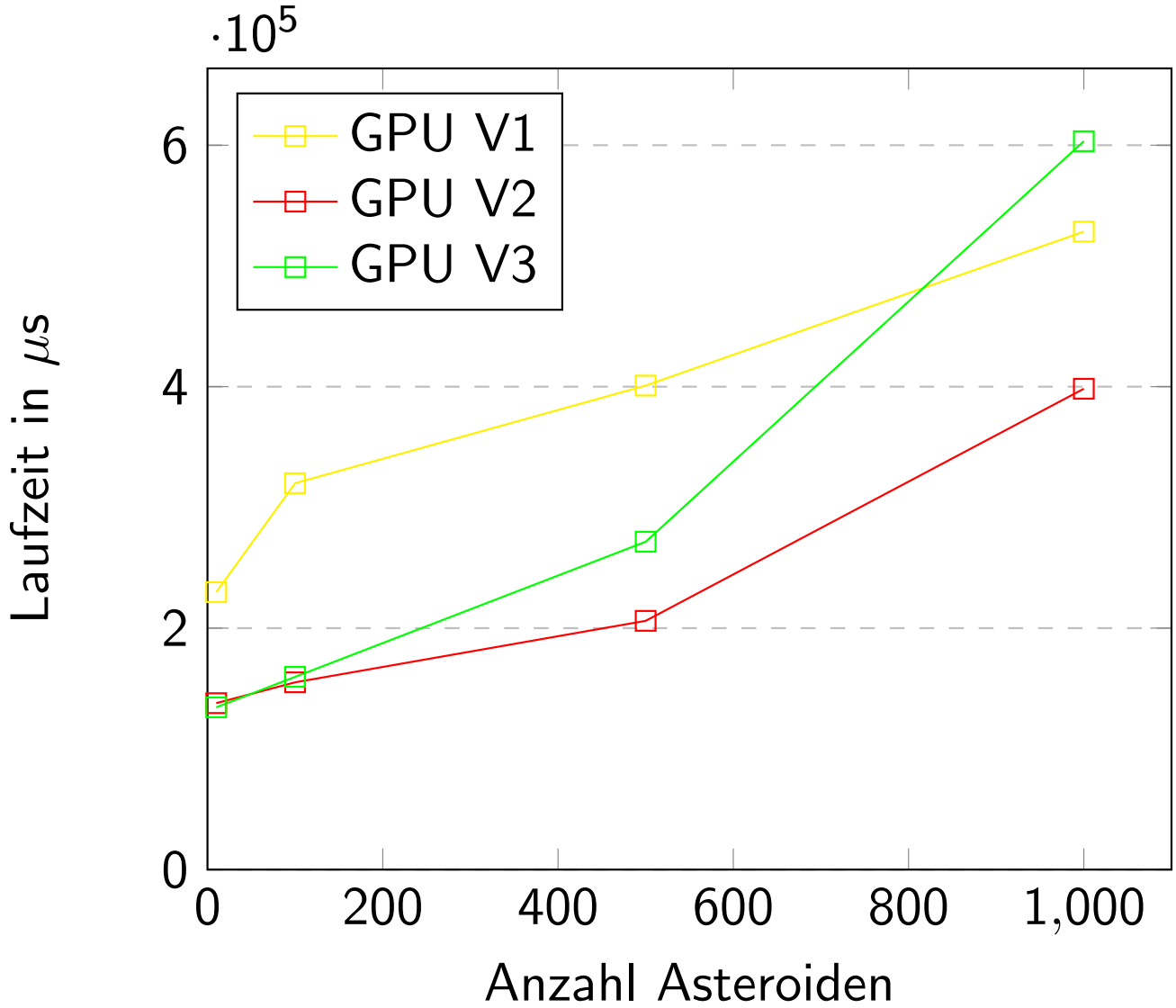

Abbildung: Laufzeit der verschiedenen Versionen für verschiedene Asteroidenanzahlen

Fazit

- Hohe Asteroidenanzahl $\rightarrow$ GPU deutlich schneller als CPU
- Kopieren der Asteroiden auf CPU kostet am meisten Zeit
- GPU Version 3 verbessert nicht
- Schwierigkeit bei hohen Asteroidenzahlen
  - teilweise serielle Ausführung mit Cuda-Grids
  - Abstimmungs- und Effizienzprobleme

Fazit

Versionen			
CPU	GPU V1	GPU V2	GPU V3
$\binom{N}{k}$ *Berechnungen*	Asteroid parallelisiert	Datentransfer reduzieren	Sämtliche Interaktionen parallelisiert
Seriell	Interaktionen seriell		Threadlimit
	Leichter Implementationaufwand		Einschränkungen beim Aufsummieren
			Hoher Implementationsaufwand

Versionen

CPU

n^2 serielle Berechnungen

GPU V1

Asteroiden parallelisiert, aber Interaktionen seriell

Leichter Implementationsaufwand

GPU V2

Datentransfer reduzieren

GPU V3

Sämtliche Interaktionen parallelisiert

höherer Mehraufwand beim Aufsummieren

Fazit

- Deutliche Beschleunigung durch die GPU
- Kopiervorgang zwischen CPU und GPU ist Flaschenhals
- GPU V3: Synchronieren der Daten aufwendig und kostet Zeit