# 1 Problemstellung verstehen (1.-2. Woche)

## 1.1 Literatur lesen (1. Woche)

### 1.1.1 Bachelor-Arbeit

### 1.1.2 X-ray paper

### 1.1.3 FT

### 1.1.4 Gauss'sche Kernel

### 1.1.5 CUDA streams/async memory

### 1.1.6 cuFFT/cuBLAS

## 1.2 Programmaufbau festlegen (2. Woche)

### 1.2.1 SW in Abschnitte teilen

### 1.2.2 Parallelisierungspotenziale erkennen

**Datenabhängigkeiten erkennen**

**Problemaufteilung maximieren**

### 1.2.3 memory access koordinieren

### 1.2.4 Speicherformat festlegen

binäres Pixmap (Versuchsdatenformat beachten)

# 2 Implementierung (3.-7. Woche)

## 2.1 Beispiel-Daten erstellen (3. Woche)

### 2.1.1 Beispiele festlegen

**symmetrische** Kugel, Kegel

**periodische** Schachbrett

**asymmetrische** halbe Kugel, halber Kegel, grobes Rauschen

#### 2.1.2 python-Programm schreiben

### 2.2 Algo-Implementierung (4.-7. Woche)

#### 2.2.1 IO

#### 2.2.2 HIO-Implementierung

#### 2.2.3 Shrink-Wrap

#### 2.2.4 Verifizierung der Ergebnisse anhand der Beispieldaten

#### 2.2.5 Benchmarking

#### 2.2.6 falls Zeit reicht: Performance-Analye und -Optimierung

### 2.3 Doku schreiben (parallel)

### 2.4 Examples schreiben (parallel)

## 3 Abschlusspräsentation (8. Woche)

### 3.1 Bilder

#### 3.1.1 Benchmarks durchführen

Referenzdaten (z.B. aus Bachelorarbeit) sichten und vergleichen

#### 3.1.2 Ausgabedatenvisualisierung