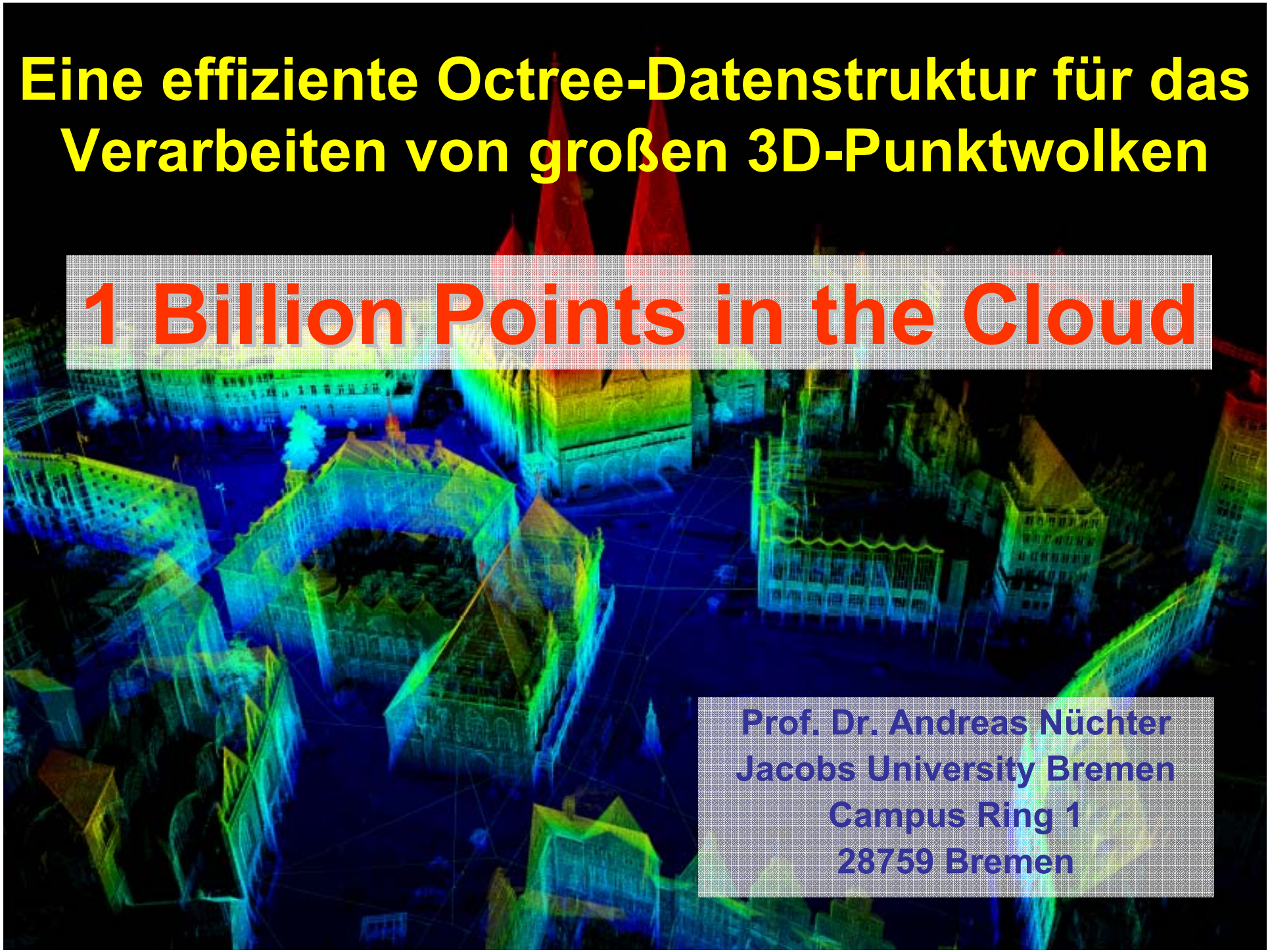
Eine effiziente Octree-Datenstruktur für das Verarbeiten von großen 3D-Punktwolken



Prof. Dr. Andreas Nüchter
Jacobs University Bremen
Campus Ring 1
28759 Bremen

Hintergrund

- Automatisierung von terrestrischen Laserscanning

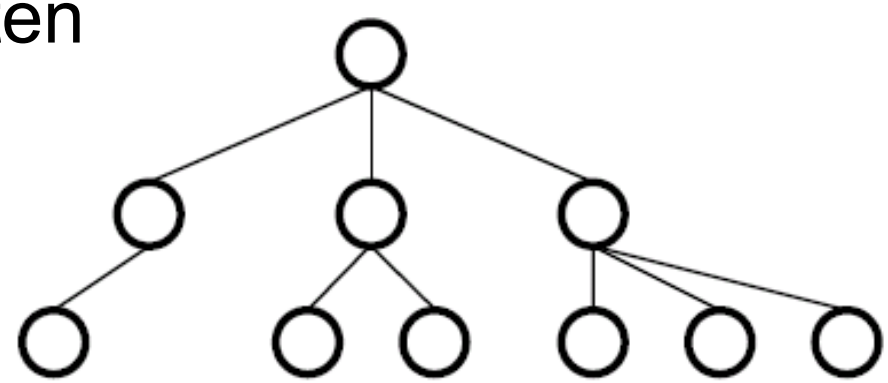
(video)

(animation)

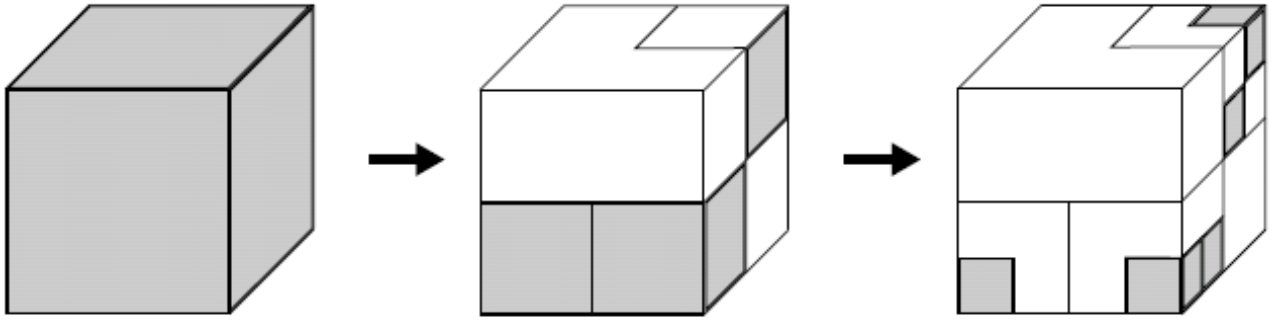
  - Kombination terrestrisches / kinematisches Laserscanning

Definition: Octree

- Datenstruktur in der Informatik
- Baum mit 8-Nachfolgeknoten



- Unterteilt einen Würfel in 8 gleich große Teilwürfel



Octree: Standard Implementation

- Octree in C/C++:

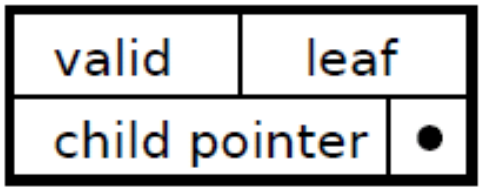
```
struct OcTree {
  float center[3];
  float size[3];
  OcTree *child[8];
  int nr_points;
  float **points;
};
```

3 x 4 byte	
3 x 4 byte	
8 x 8 byte	
8 byte	
4 byte	
= 100 byte	

- Aber: Ein großer Teil der Information, die in der **`struct`** gespeichert wird, kann während des Durchlaufens des Baumes berechnet werden!
  - ⇨ Reduktion um 24 Bytes

Octree: Effiziente Implementation (1)

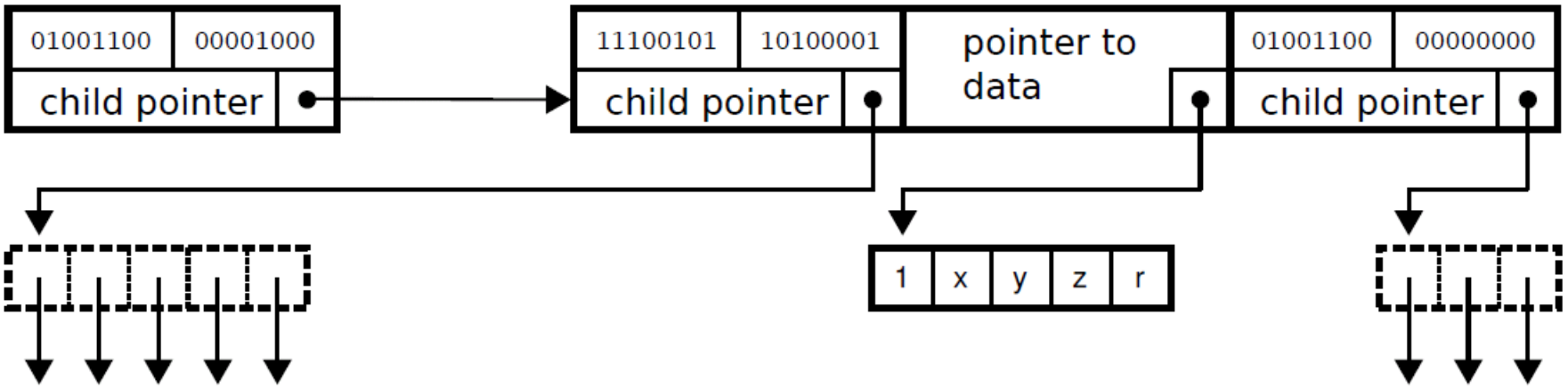
- Bitfelder für Flags für Nachfolgerknoten und Blattknoten



- Speicherung *eines* Zeigers auf die Nachfolgerknoten, mit 6 Bytes.
  - Dies reicht aus, um 256 Terrabyte zu adressieren
- Insgesamt: 8 Byte (von ehemals 100 Bytes)
- Punkte werden ausschließlich in den Blättern gespeichert und somit benötigen diese eine andere Repräsentation.

Octree: Effiziente Implementation (2)

- Punkte werden ebenfalls mit einem 6-Byte Zeiger dargestellt.
  - Der erste Eintrag ist die Anzahl der Punkte (4 Bytes)
  - Dann kommen die Daten
  - ⇨ Effizienter als `float**`-Feld
- Beispiel:



Speichern der 3D-Punkte

- Den größten Speicherplatz nehmen nun die 3D-Punkte mit ihren Attributen

- Nun muss die Punktliste selbst komprimiert werden.

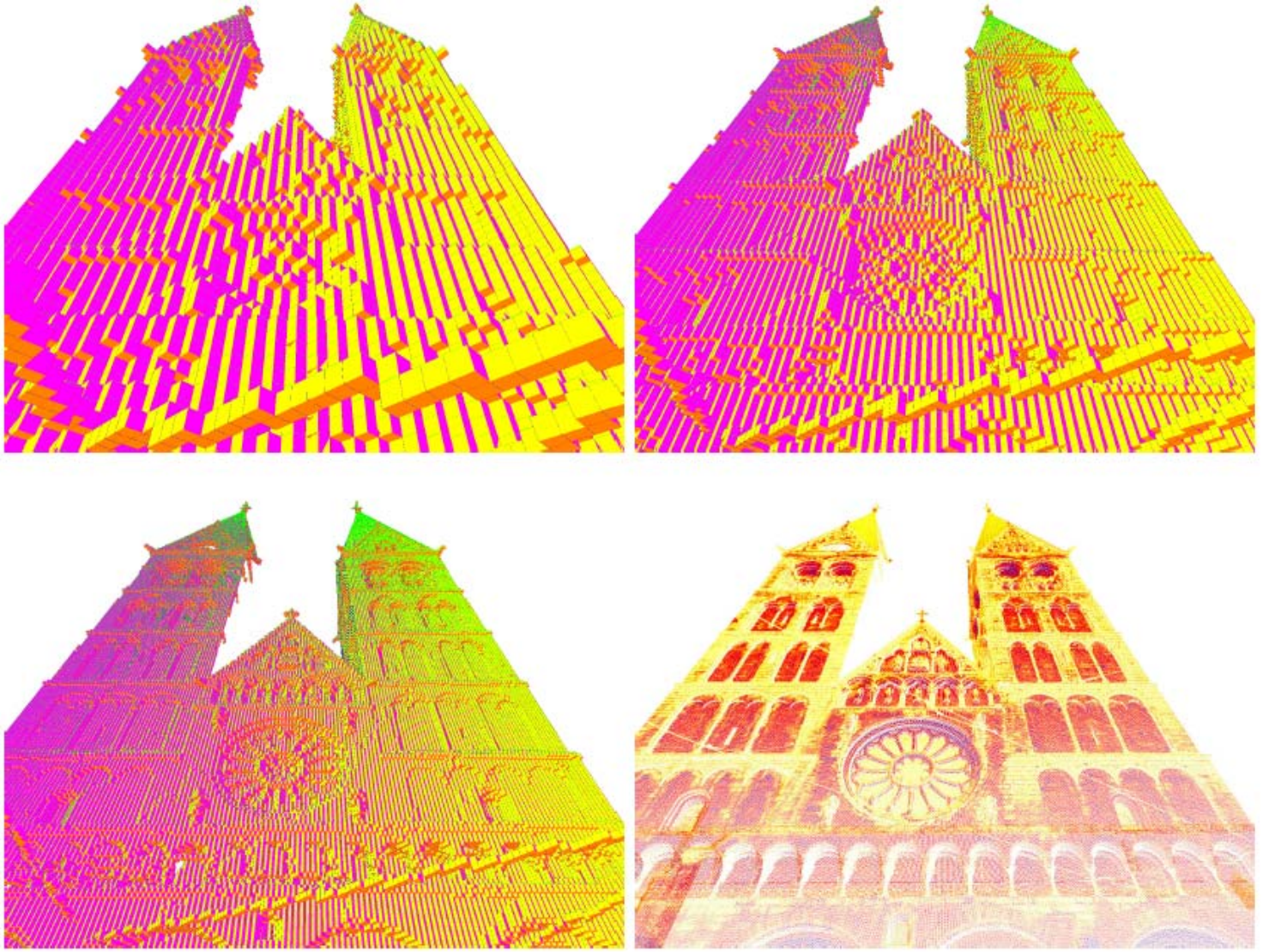
- Bisher: Ein `float` für jeden Eintrag (=4 Bytes)!
- Wir verwenden 2 Bytes für jede Koordinate
  - Aber: Die Auflösung der 3D-Daten darf sich durch 2-Byte-Koordinaten jedoch nur unmerklich ändern.
  - Jeder der 2-Byte-Koordinaten wird als $s/(2^16)$ Inkrement von der unteren, vorderen, linken Ecke eines Blattknotens angenommen, wobei s die Seitenlänge des Octree-Würfels ist.
- 2 Byte für Zusatzinformationen ist ausreichend

Genauigkeit der 3D-Punkte

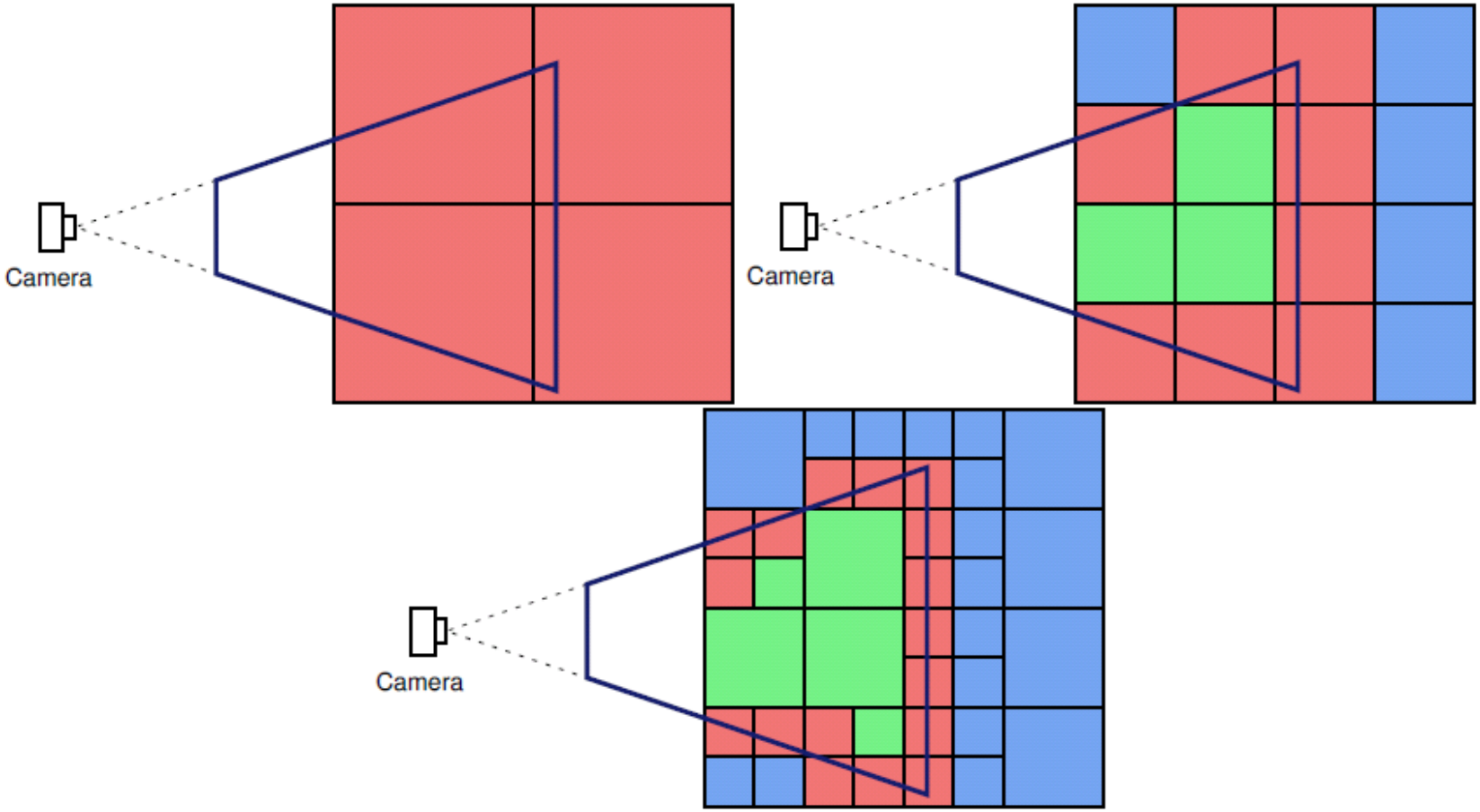
- `float` besitzt eine Genauigkeit von
  - 100 µm bei 500 m max. Entfernung
  - 1 µm bei 1.5 m max. Entfernung

- Unser Octree hat
  - 10 µm bei einer Seitenlänge eines Würfels von 65 cm
  - (bei 6.5 cm Seitenlänge hätten wir 1 µm)

Ergebnis: Detailgrade durch Octree-Darstellungen

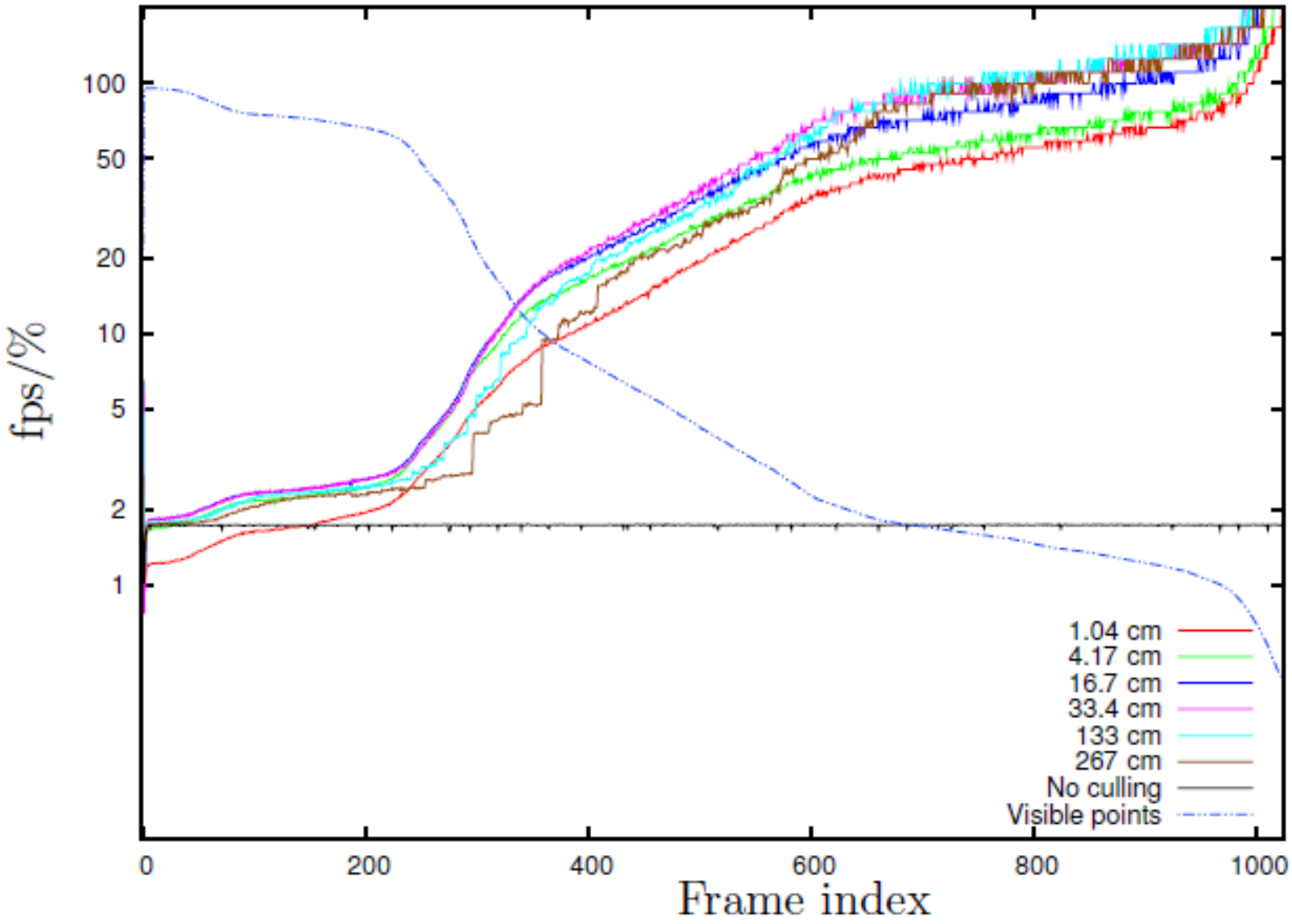


Anwendung: Frustum Culling auf Octrees



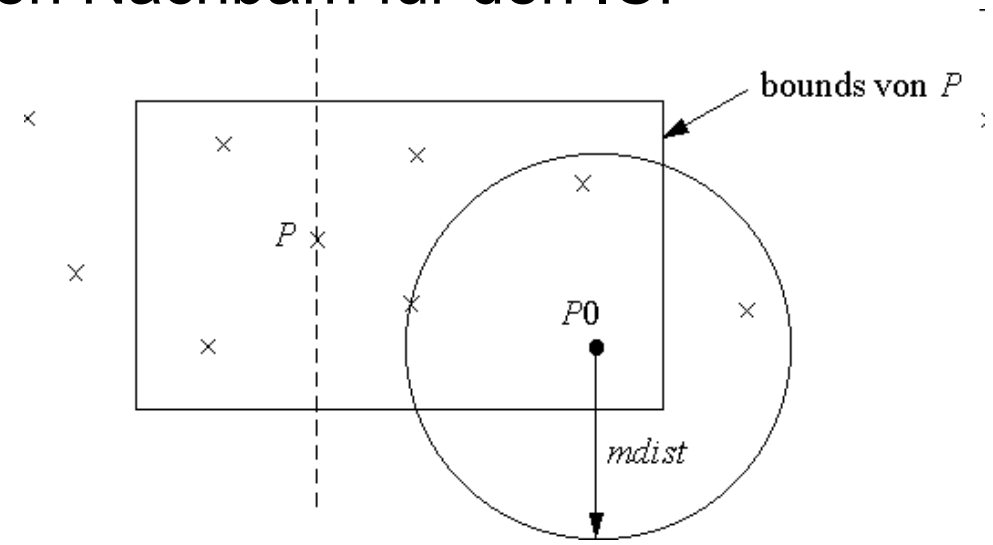
- Grün: Zellen im Frustum, Blau: Zellen Außerhalb
  Rot: Zellen müssen weiter untersucht werden (Programm)

Ergebnisse Frustum Culling mit Octrees



Weiter Anwendungen / Ausblick

- Ray-Casting
  - Vereinfachtes Ray-Tracing
- RANSAC Algorithmus
  - Zur Extraktion von 3D-Ebenen / Flächen aus Punktwolken
- Scanmatching
  - Berechnung von nächsten Nachbarn für den ICP-Algorithmus
    - Aktuell: 20% langsamer als k-d Baum
    - Aber: platzsparender



Zusammenfassung

- Effiziente Octree-Datenstruktur wurde vorgestellt
- Komprimierte 3D-Punktwolken ohne Genauigkeits-verlust
- Schnelles Verarbeiten von bis zu

1 Milliarde Punkte in 8GB Hauptspeicher

- Wir suchen nach einem Partner, um die entwickelte Software zu kommerzialisieren. Bei Interesse bitte an Andreas Nüchter (andreas@nuechti,.de) wenden