Konservative Interpolation zwischen blockstrukturierten Hexaedergittern

Hans-Peter Kersken, SISTEC

SISTEC-Workshop

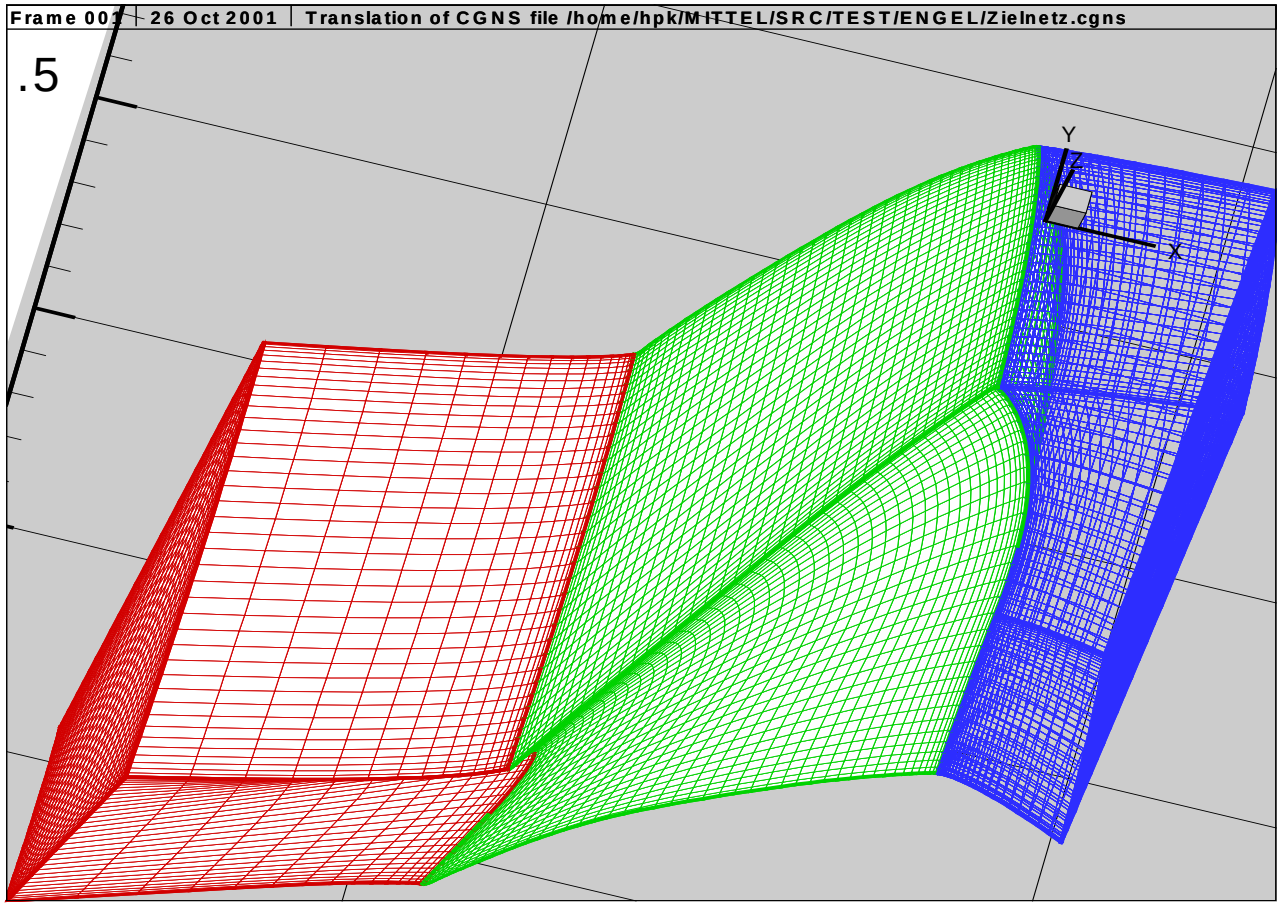
5. November 2001

Köln

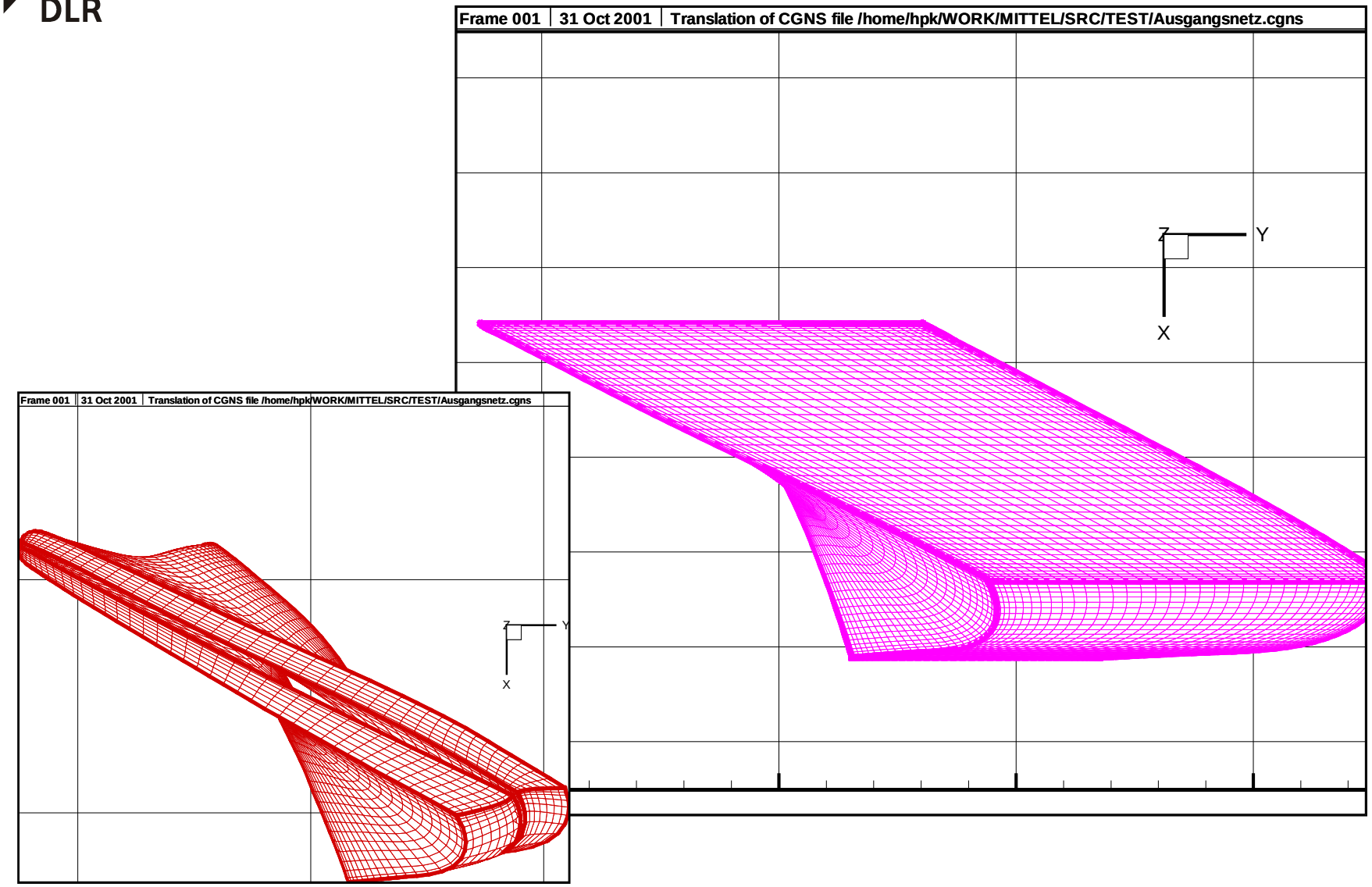
Übersicht

- Motivation
- Algorithmus
- Implementierung
- Ergebnisse
- Zusammenfassung

Blockstrukturierte Hexaedergitter



Motivation


Frame 001
31 Oct 2001
Translation of CGNS file /home/hpk/WORK/MITTEL/SRC/TEST/Ausgangsnetz.cgns
Z
Y
X
Frame 001
31 Oct 2001
Translation of CGNS file /home/hpk/WORK/MITTEL/SRC/TEST/Ausgangsnetz.cgns
Z
Y
X

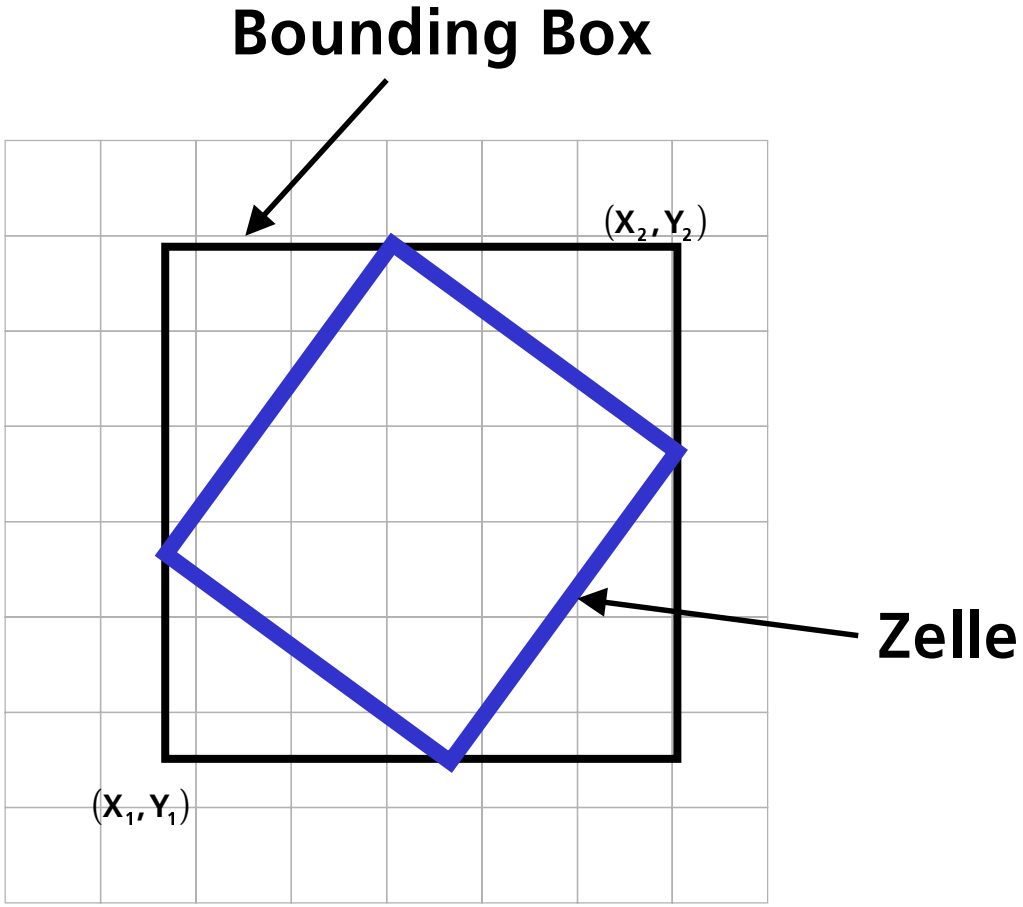
Interpolationsalgorithmus

- Finde Schnittvolumina zwischen Zellen des neuen und alten Gitters
  - Finde überlappende Zelle (Bounding box tree)
  - Berechne Schnittvolumina (Hexaederzerlegung)
- Berechne neue Werte auf dem Zielgitter

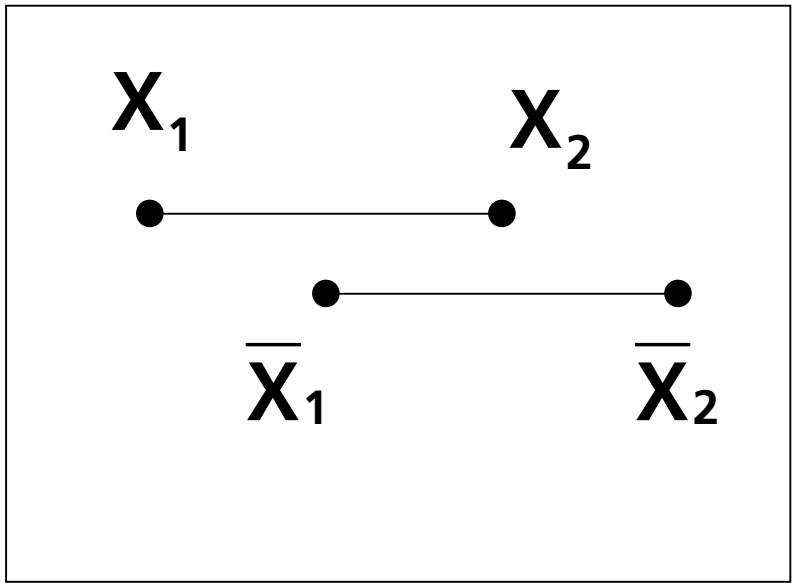
$$\rho_j^{neu} = \frac{1}{V_j} \sum_{v_{ij} \neq 0} \rho_i^{alt} v_{ij}$$

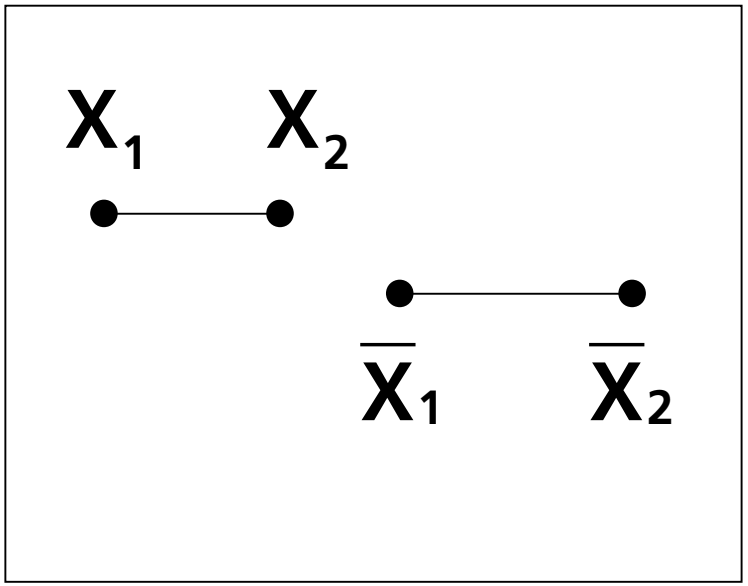
Conservative Remapping and Region Overlays by Intersecting Arbitrary Polyhedra, Jeffrey Grandy, JCP, 1999

Bounding Box



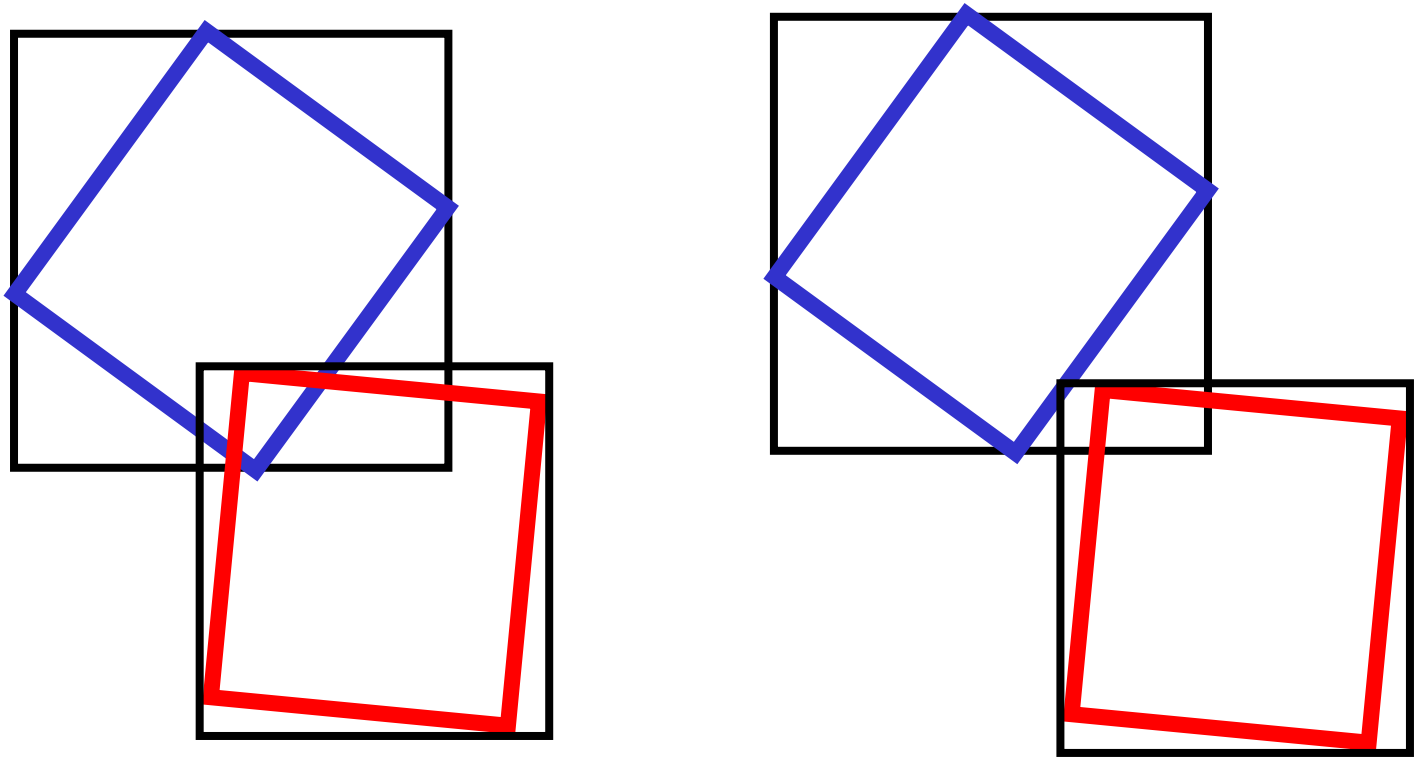
Test auf Überlappung 1D





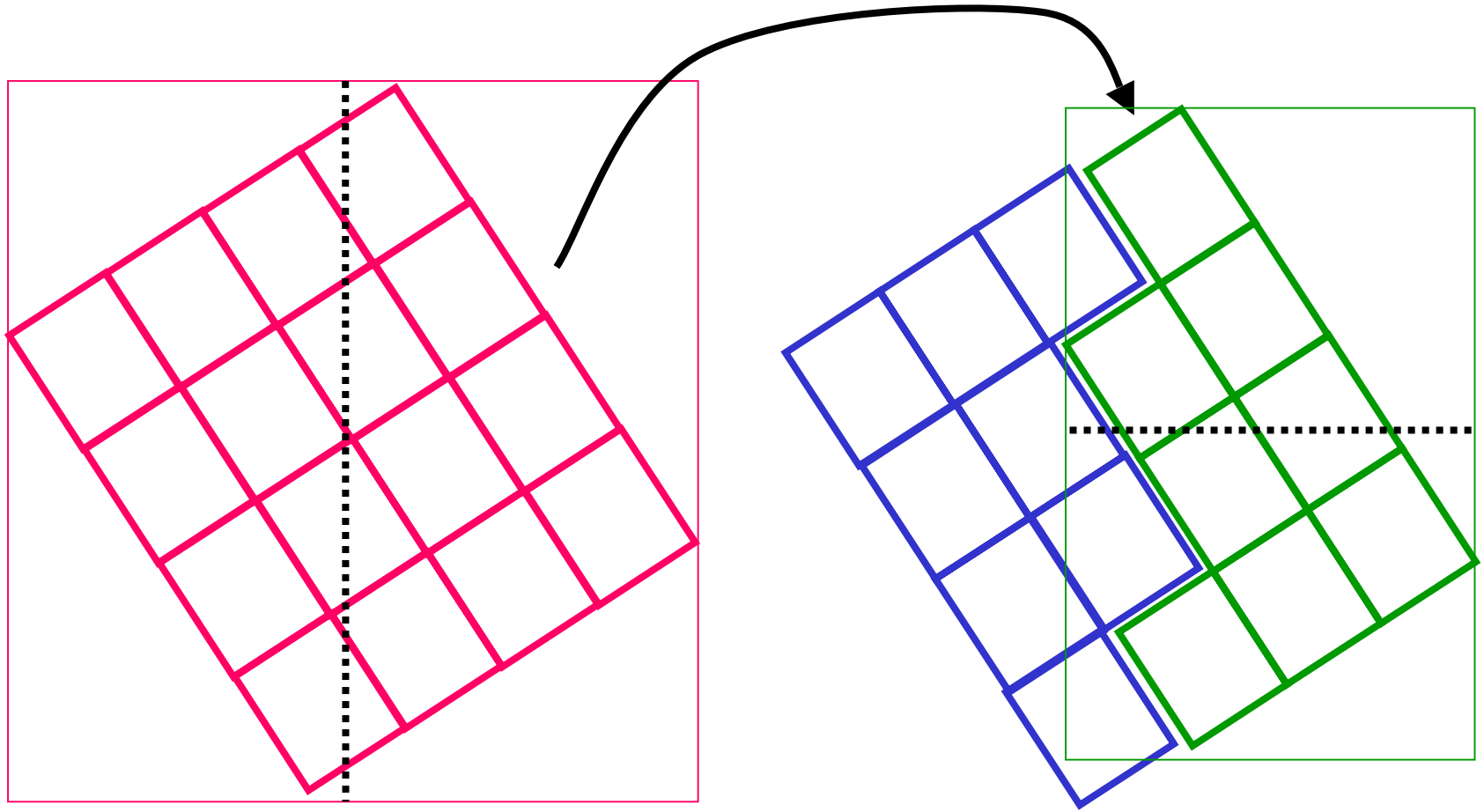
$$X_1 < \overline{X}_2 \quad \wedge \quad X_2 > \overline{X}_1$$

Test auf Überlappung 2D

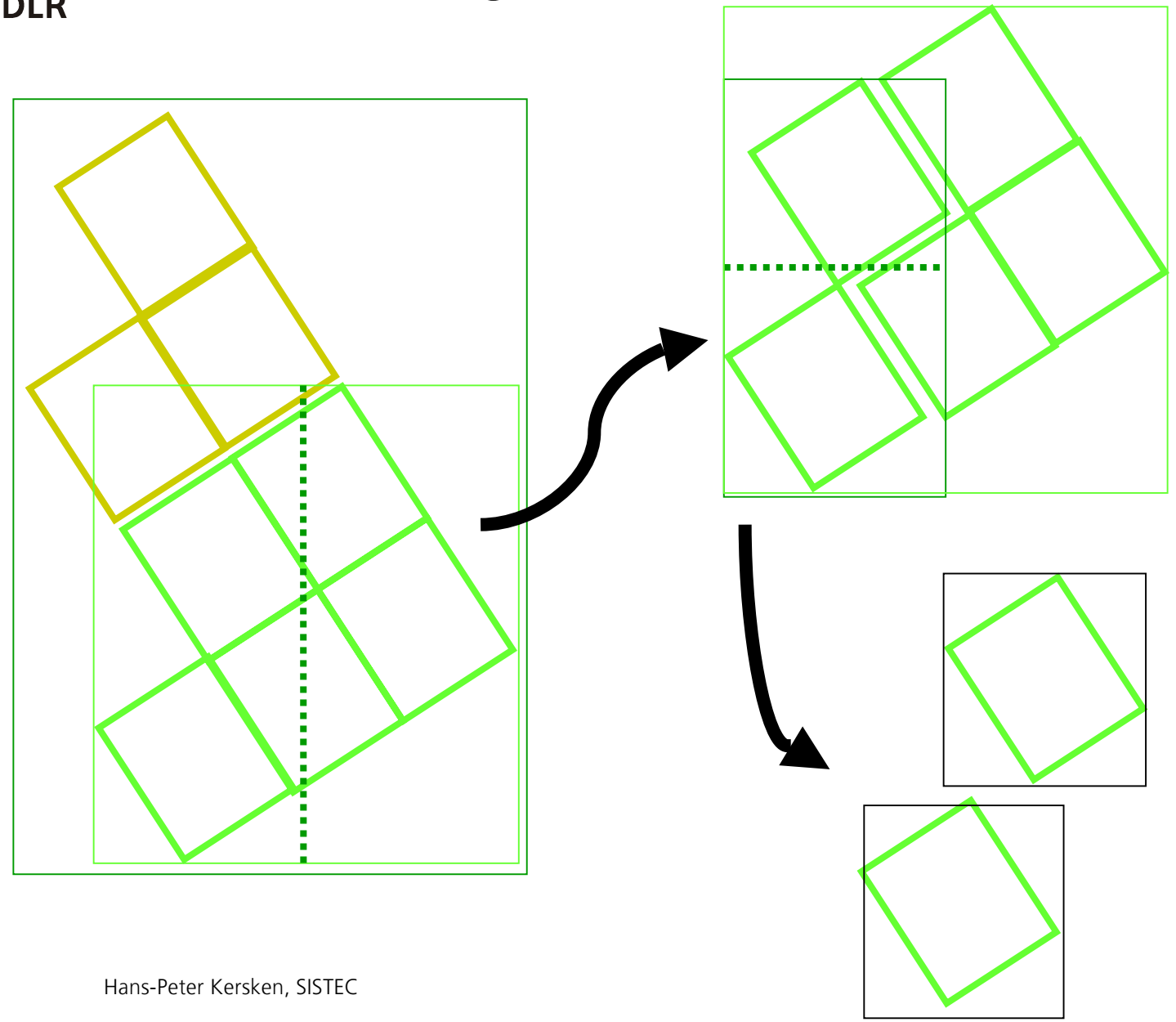


$$X_1 < \overline{X}_2 \quad \wedge \quad X_2 > \overline{X}_1$$
$$Y_1 < \overline{Y}_2 \quad \wedge \quad Y_2 > \overline{Y}_1$$

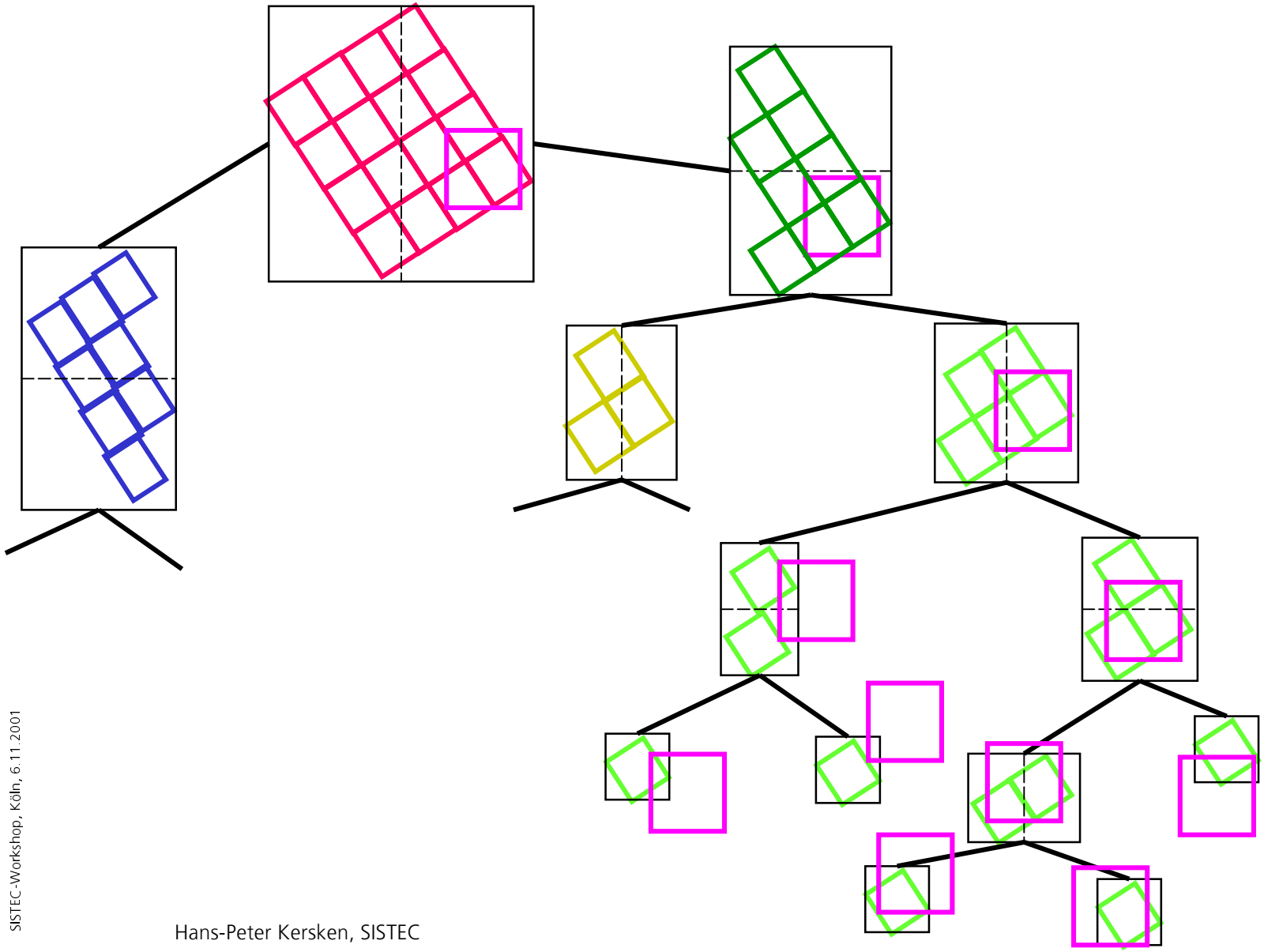
Bounding Box Tree (I)



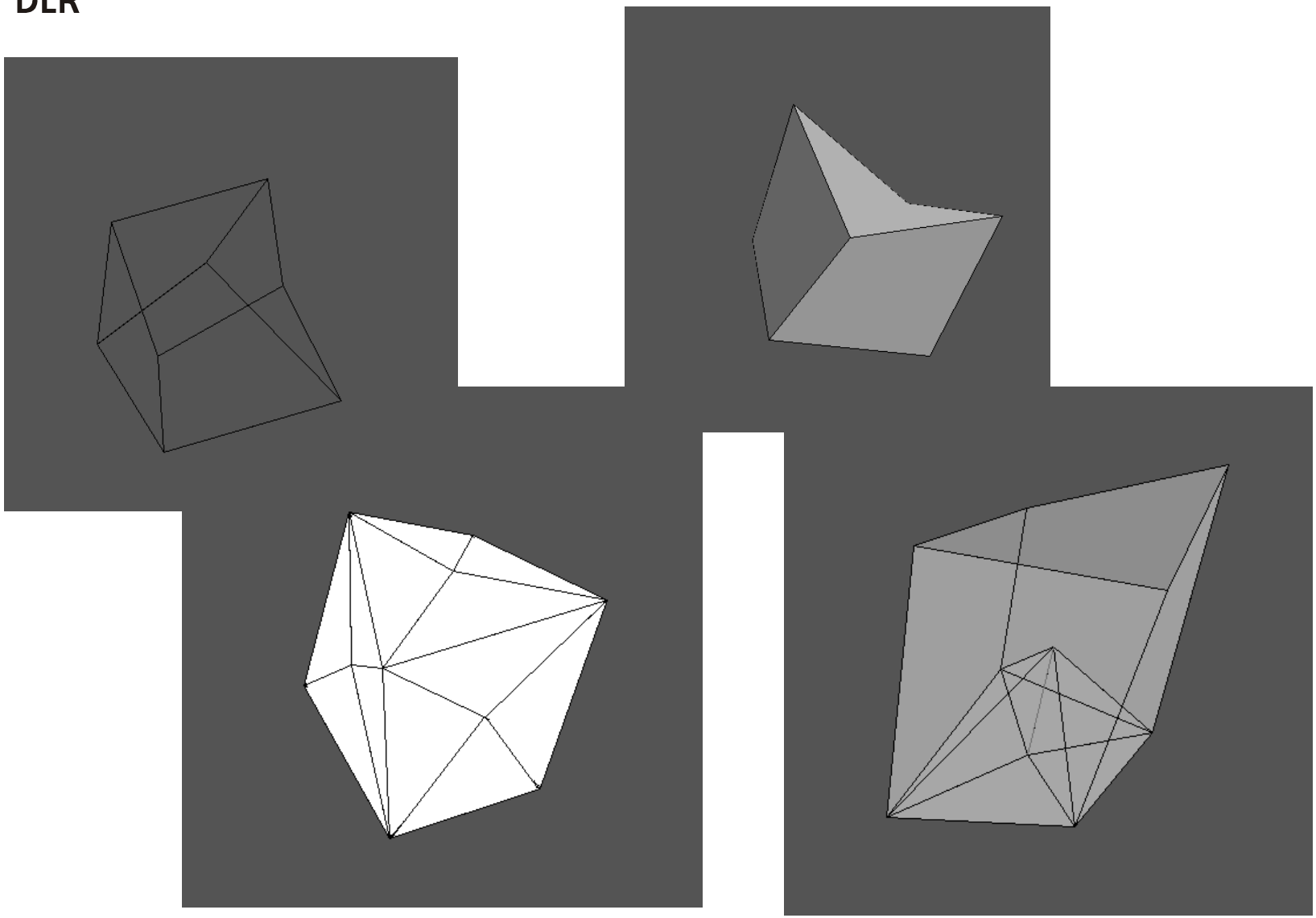
Bounding Box Tree(II)



Bounding Box Tree - Suche

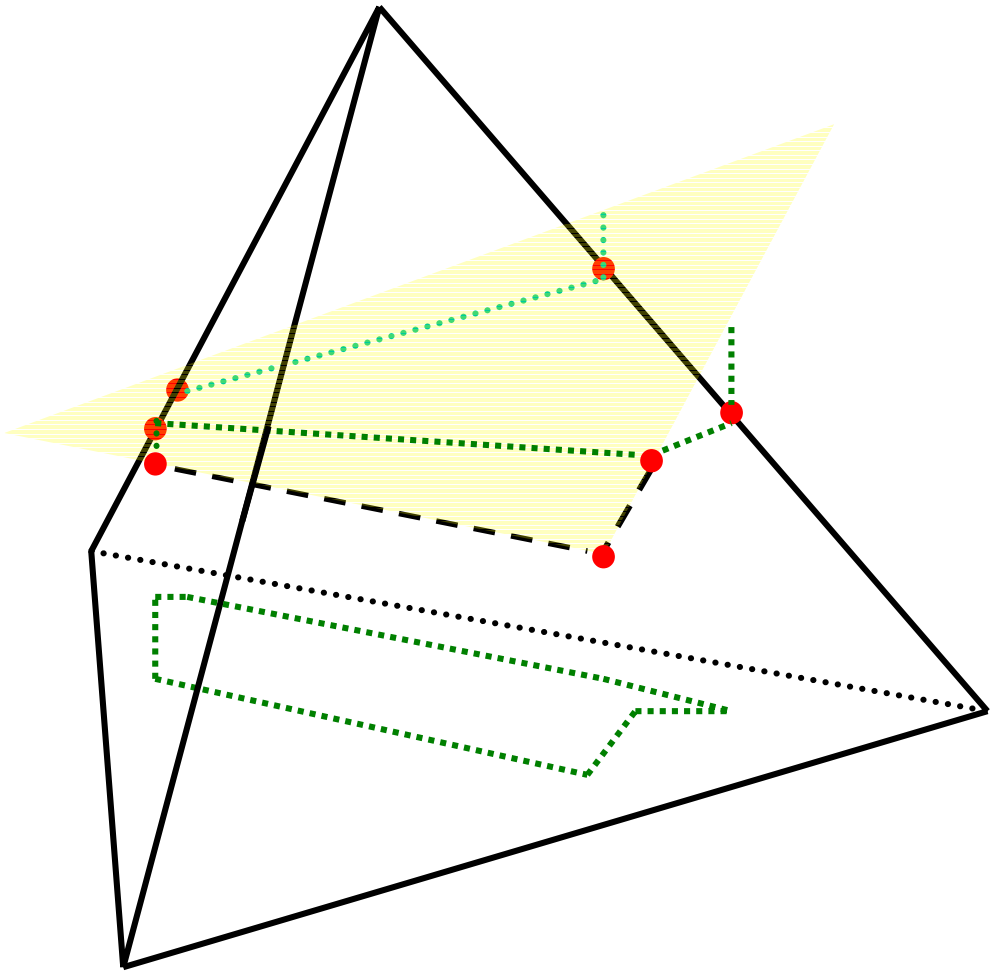


Hexaederzerlegung

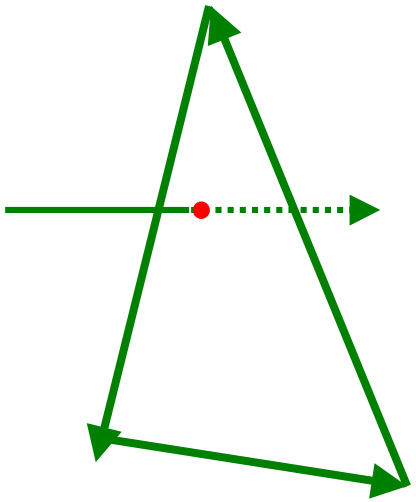


Berechnung der Schnittvolumina

Tetraeder-Dreieck Schnittpunkte



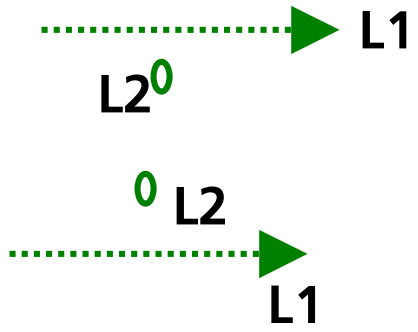
Segment-Dreieck-Schnittpunkte



Geradengleichung: $Q(t) = tU + P$

Plücker-Koordinaten: $L = \{U : UxP\} = \{U : V\}$

$$w = U_1 \quad V_1 + U_2 \quad V_2$$



w<0 L1 dreht im Uhrzeigersinn um L2

w>0 L1 dreht gegen Uhrzeigersinn um L2

Numerische Probleme

- **Endliche Genauigkeit erschwert:**
  - **Eindeutige Identifikation degenerierter Fälle**
    - **Segment-Segment-Schnitte**
    - **Segment-Knoten-Schnitte**
    - **Segment parallel zu Dreiecksfläche**
  - **Exakte Berechnung der Schnittpunkte**

Programmablauf (I)

intersect:

Für jeden zugeordneten Block des Ausgangsgitter:

Lese Blockgeometrie

Für jeden Block des Zielgitters:

Lese Blockgeometrie

Für jeden Drehwinkel:

Für jede Zelle des Zielblocks

Finde überlappende Zellen des Ausgangsblocks

Berechne und speichere Schnittvolumina

Schreibe Ergebnisdatei

Programmablauf (II)

average:

Für jeden Block des Ausgangsgitter:

- **Lese Blockgeometrie**
- **Lese Strömungsgrössen**
- **Für jeden Block Zielgitter:**
  - **Lese Blockgeometrie**
  - **Lese Datei mit Schnittdaten für Ziel/Ausgangs-Block-Paar**
  - **Berechne Werte auf dem Zielgitter**
  - **Schreibe Werte in CGNS-Datei**

Implementierung

- C++
- C-Felder
- MPI-parallelisert auf Block-Ebene
- 7219 Programmzeilen
  - 2636 Kommentare
  - 784 CGNS-Interface
  - 3799 Kernalgorithmus

Beispiel

Zielgitter	**3 Blöcke, 112640 Zellen**
Ausgangsgitter	**10 Blöcke, 70016 Zellen**
Pitchwinkel	**25,71°**
Zell-Zell-Schnitte	**$12 \cdot 10^6$**
Zell-Tetraeder-Schnitte	**$60 \cdot 10^6$**
Tetraeder-Dreick-Schnitte	**$353 \cdot 10^6$**
Segment-Ebenen-Schnitte	**$1174 \cdot 10^6$**
Erhaltungsfehler	**0.012 %**

Rechenzeiten

- ***intersect***:

CPU-Zeit (UltraSparc10)	5516sec
Wallclock-Zeit (5Prozessoren)	1140sec
Speedup	4,8
Schnittzellen-Suche	12%
Schnittvolumen-Berechnung	88%

- ***average***:

CPU-Zeit	63sec

Zusammenfassung

- Konservative Interpolation 1. Ordnung
- blockstrukturierte Hexaedergitter
- CGNS I/O
- Parallelisiert