



Geometrische Datenstrukturen für die Computer-Graphik

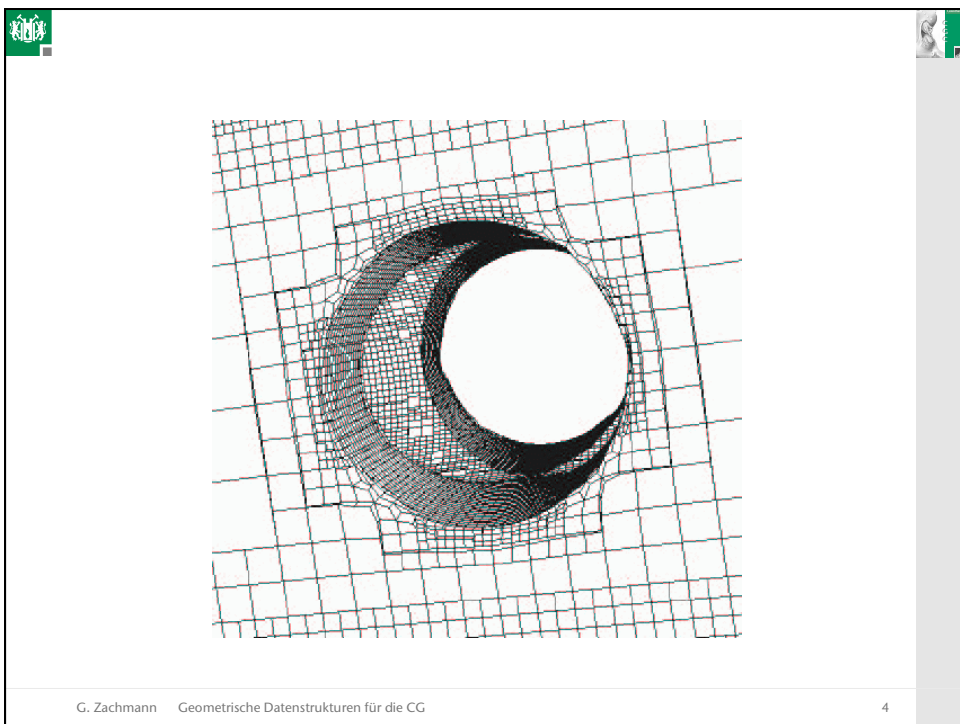
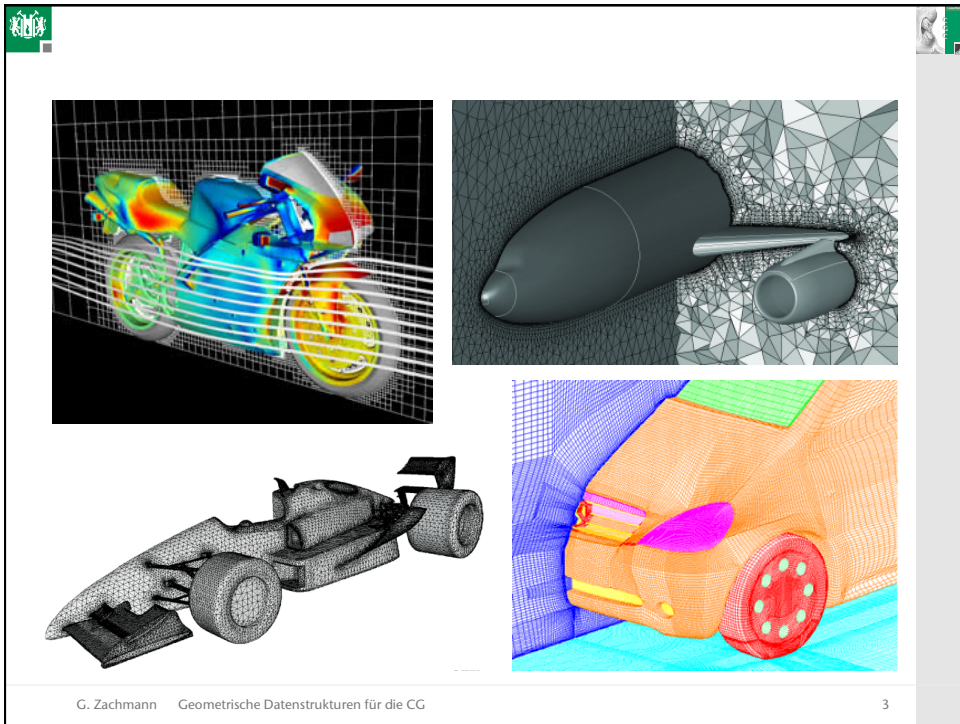
G. Zachmann
zach@tu-clausthal.de
Clausthal University, Germany



Meshing

- Wichtiger Preprocessing-Schritt in vielen Anwendungen
 - "Domain discretization" =
 - Komplexes Gebiet (2D oder 3D) wird in einfache Gebiete zerlegt (Dreiecke, Tetraeder)
- Anwendungen: FEM, CFD, VLSI = Simulation = Lösen von PDEs
 - PDEs lassen sich über regelmäßigem Gitter diskretisieren (über beliebige Gebiete nicht)

G. Zachmann Geometrische Datenstrukturen für die CG 2



Quadtree demo

<http://www.cs.utah.edu/~croberts/courses/cs7962/project/index.html>

G. Zachmann Geometrische Datenstrukturen für die CG
5

Ist IMHO buggy!
(unnötige Unterteilungen)

<http://donar.umiacs.umd.edu/quadtree/points/prquad.html>

G. Zachmann Geometrische Datenstrukturen für die CG
6

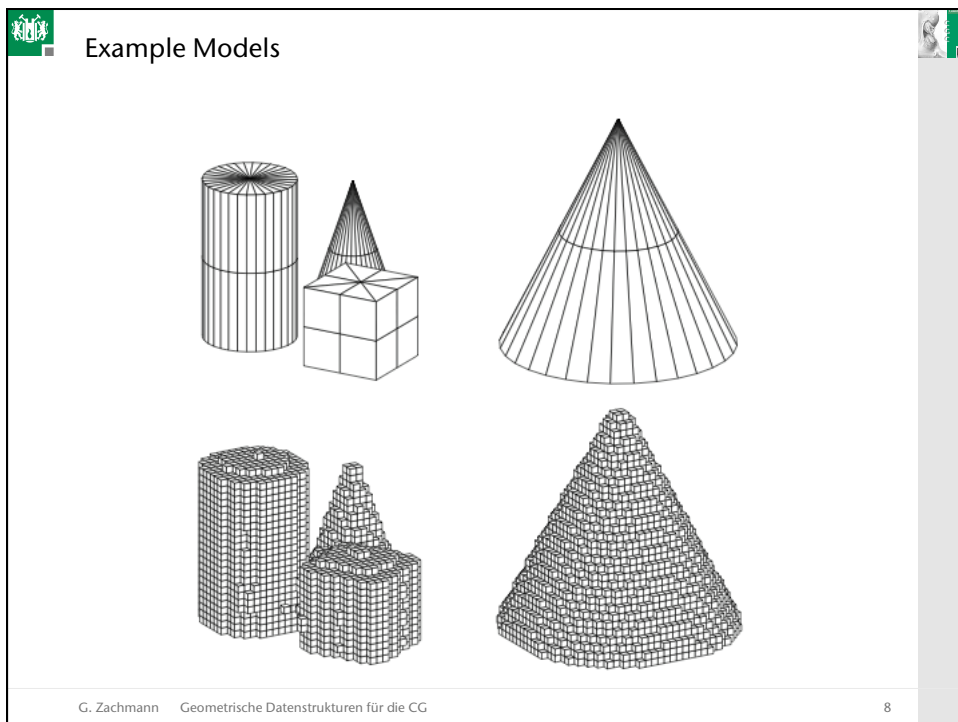
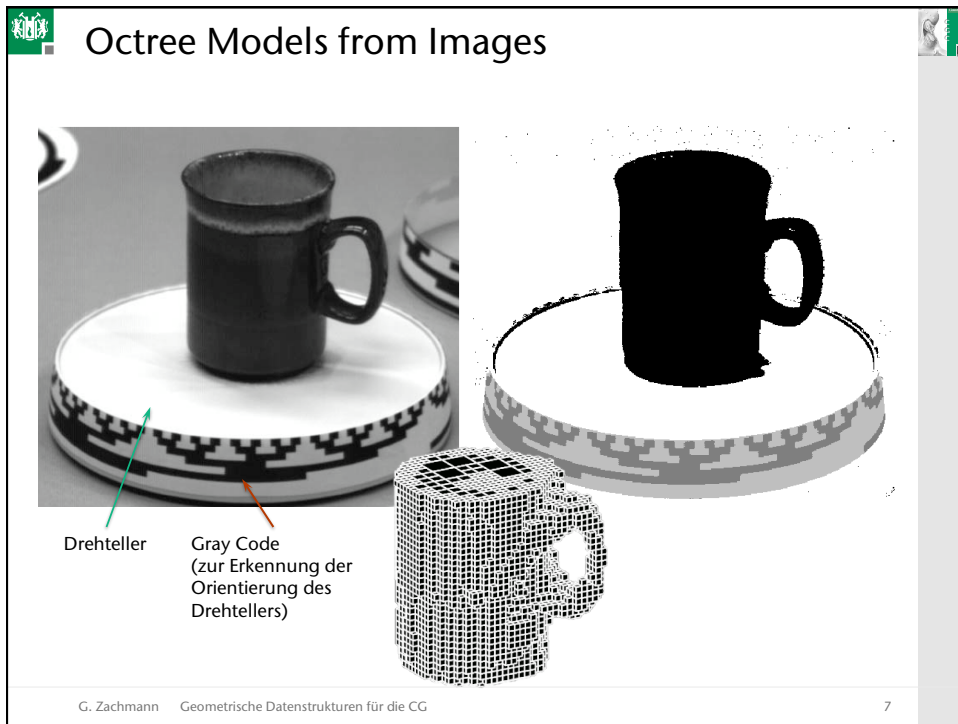
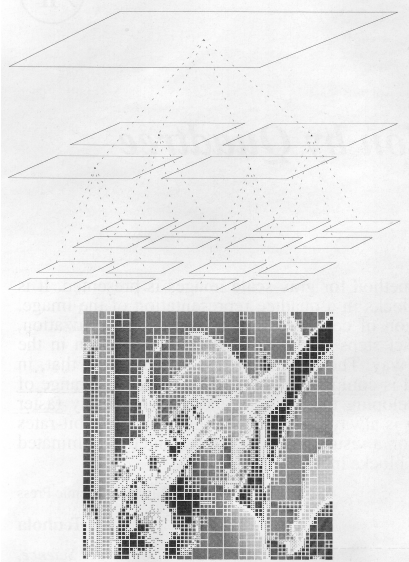



Image Compression using Quadtrees




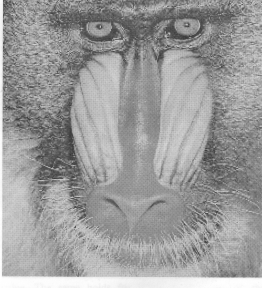

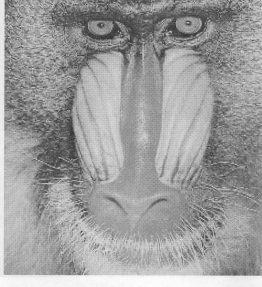
G. Zachmann Geometrische Datenstrukturen für die CG 9

Die beiden Test-Bilder schlechthin



G. Zachmann Geometrische Datenstrukturen für die CG 10




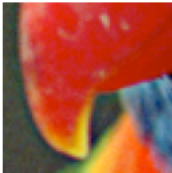
Resultate

QP: 1.03 bits per pixel 	QP: 1.95 bits per pixel 
JPEG: 1.00 bits per pixel 	JPEG: 1.99 bits per pixel 

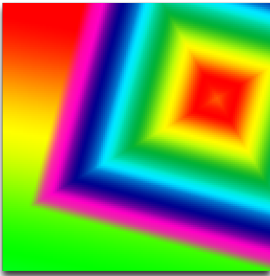
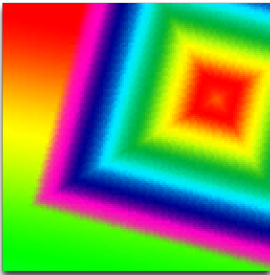
G. Zachmann Geometrische Datenstrukturen für die CG 11

S3TC texture compression

▪ Vergleich:


DXT1 	Uncompressed 
	

[Philipp Klaus Krause]


	Uncompressed [Simon Brown]
	DXT1

G. Zachmann Geometrische Datenstrukturen für die CG 12

- Vorteil: größere Texturen möglich → höhere Qualität
- Beispiel aus der Unreal Engine:



uncompressed



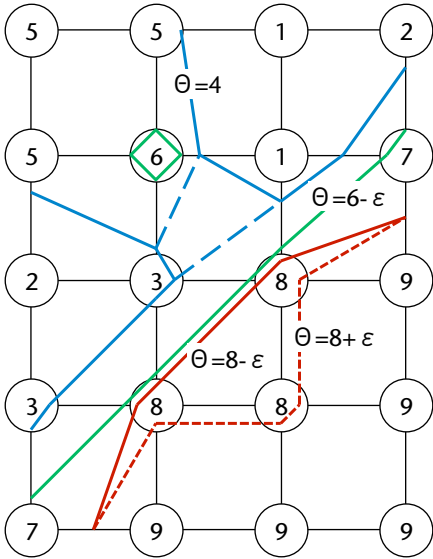
mit S3TC

Unreal Retexturing Project

G. Zachmann Geometrische Datenstrukturen für die CG 13

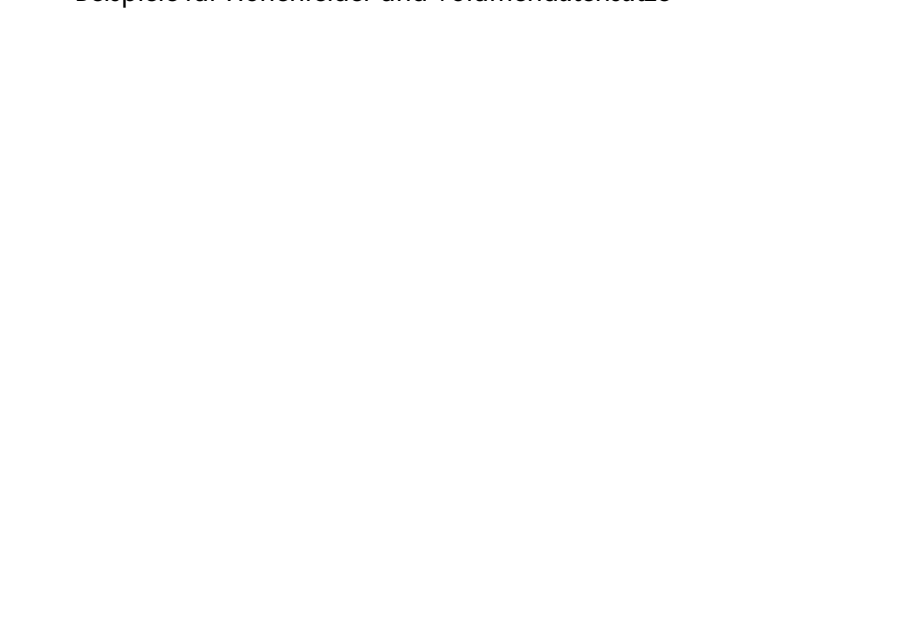
Isosurfaces

- Beispiel zur Motivation:
 - Gegeben ist ein 2D-Höhenfeld
 - Gesucht ist eine Visualisierung (in 2D!), so daß man die Form / den Verlauf des Höhenfeldes gut "erkennt"
- Eine Möglichkeit: Höhenlinien (= Konturen)
- Mögliche Probleme:
 - Plateaus
 - Singuläre "Isopunkte"
 - Uneindeutigkeiten



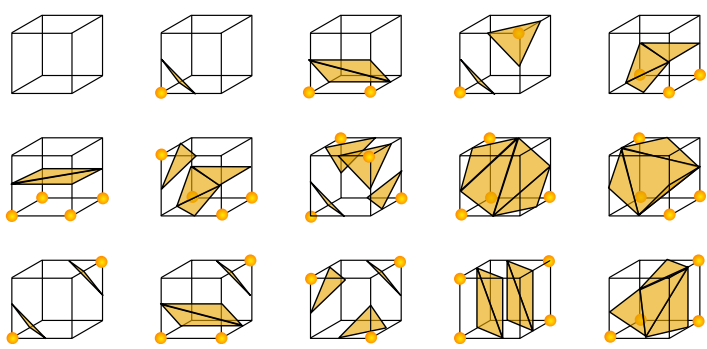
G. Zachmann Geometrische Datenstrukturen für die CG 19

Beispiele für Höhenfelder und Volumendatensätze



G. Zachmann Geometrische Datenstrukturen für die CG 20

Die 15 echt verschiedenen Fälle in 3D:



G. Zachmann Geometrische Datenstrukturen für die CG 21

Transformation
 Translate
 Rotate

Rendering
 Emerald
 Lambert
 Phong

Modeling
 With cube
 Without cube

Isosurface
 0.00

Case number
 37

Hidden

face 0.1.2.3
 face 4.5.6.7
 face 1.2.5.6
 face 0.1.4.7
 face 0.1.4.5
 face 2.3.6.7

Current cube
 Cube 0
 Vertex 0 (1)
 -100 -50 0 50 100 Min Max
 Vertex 1 (2)
 -100 -50 0 50 100 Min Max
 Vertex 2 (4)
 -100 -50 0 50 100 Min Max
 Vertex 3 (8)
 -100 -50 0 50 100 Min Max
 Vertex 4 (16)
 -100 -50 0 50 100 Min Max
 Vertex 5 (32)
 -100 -50 0 50 100 Min Max
 Vertex 6 (64)
 -100 -50 0 50 100 Min Max
 Vertex 7 (128)
 -100 -50 0 50 100 Min Max

Use ambiguous cases resolution

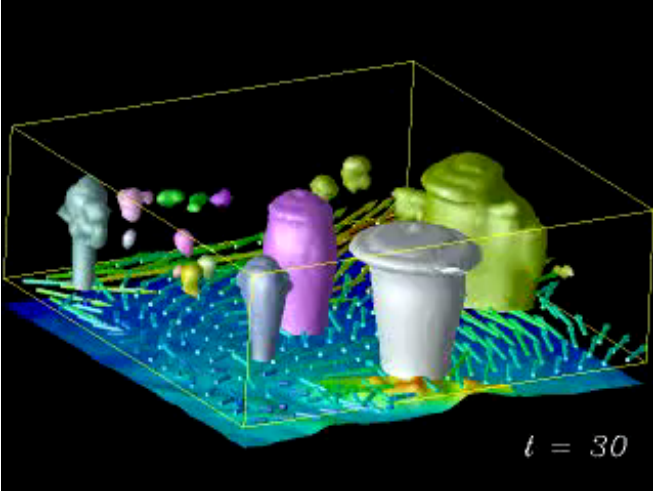
<http://users.polytech.unice.fr/~lingrand/MarchingCubes/accueil.html>

G. Zachmann Geometrische Datenstrukturen für die CG 22

▪ Output eines einfachen Marching-Cube-Algorithmus!:

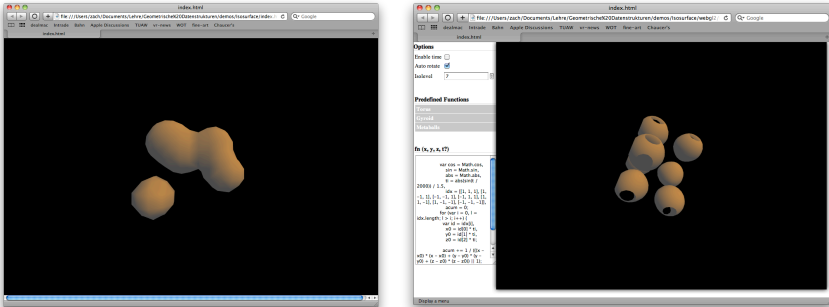
G. Zachmann Geometrische Datenstrukturen für die CG 23

Beispiel aus einer Wetter-Simulation



$t = 30$

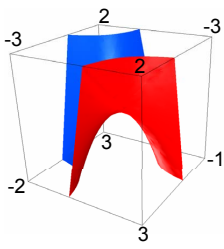
G. Zachmann Geometrische Datenstrukturen für die CG 24



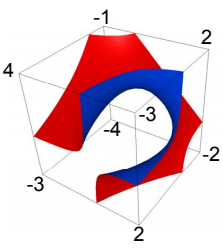
<http://blog.thejit.org/2010/12/10/animating-isosurfaces-with-webgl-and-workers/>

G. Zachmann Geometrische Datenstrukturen für die CG 25

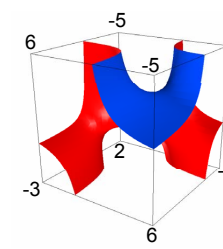
Knifflige Fälle für jeden Isosurface-Algorithmus



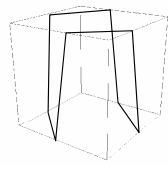
8-sided polygon



9-sided polygon



12-sided polygon



The 8-sided polygon has no valid triangulation!

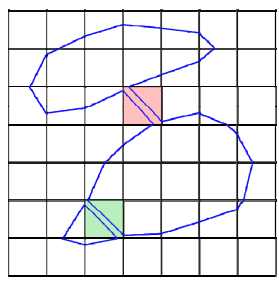
- either some triangles lie on faces of the cell
- or an extra vertex has to be used

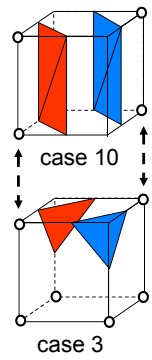
~/avs/networks/SciVis/AD*net

G. Zachmann Geometrische Datenstrukturen für die CG

26

- Manchmal passen die Dreiecke der benachbarten Zellen nicht zusammen:
- Uneindeutiger Fall im 2D:

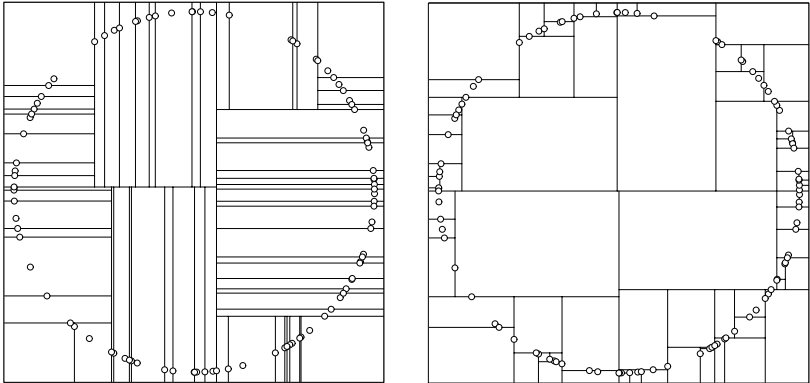




G. Zachmann Geometrische Datenstrukturen für die CG

27

Pivot-Strategien beim Aufbau von kd-Trees

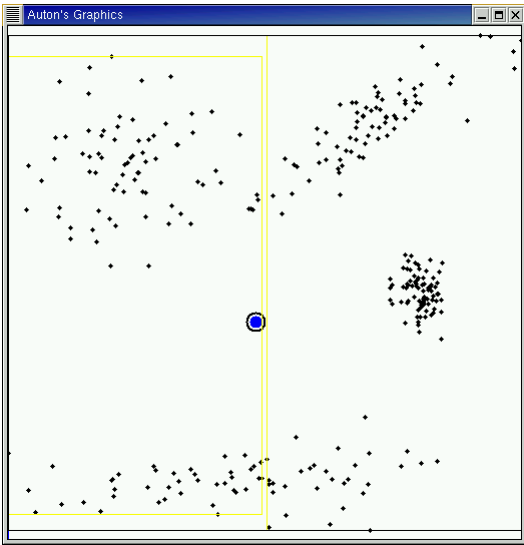


median along the dimension with the widest spread

the point closest to the center along the dimension with longest side of the region

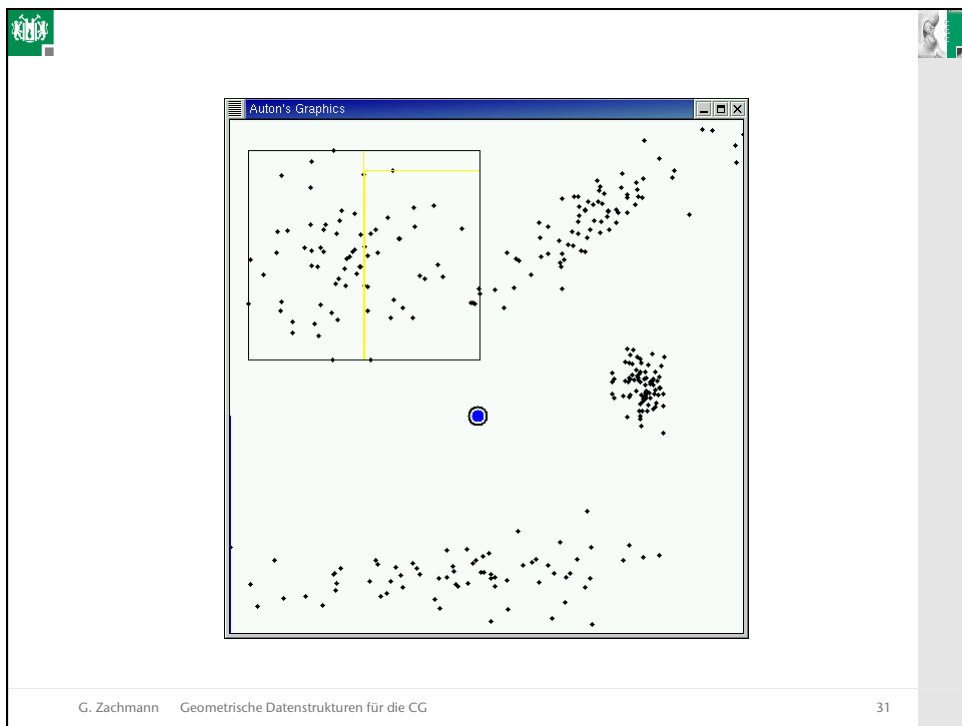
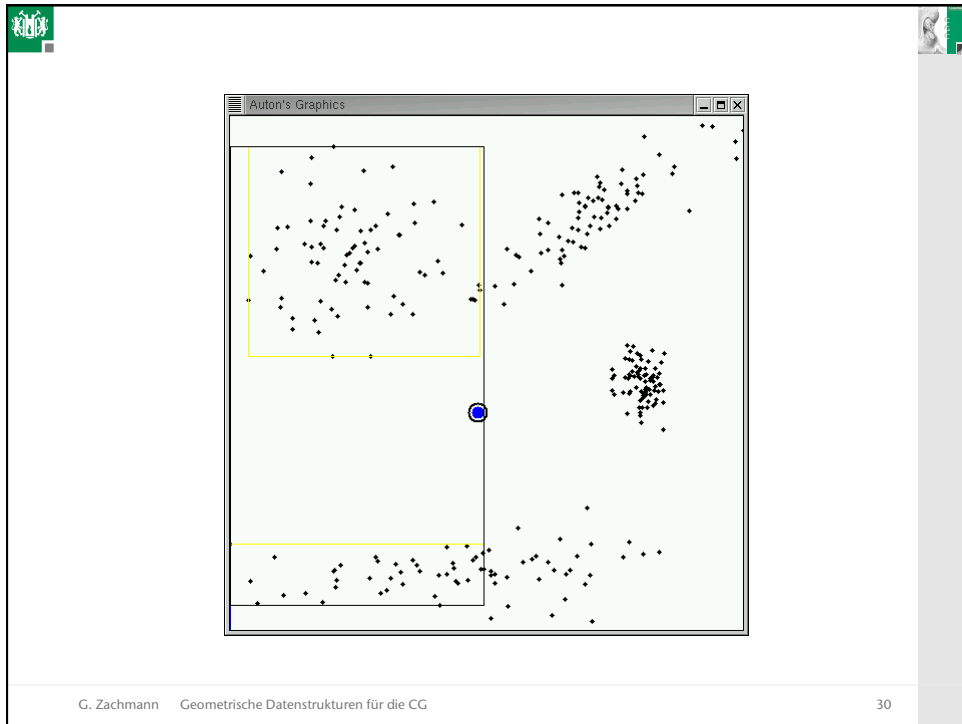
G. Zachmann Geometrische Datenstrukturen für die CG 28

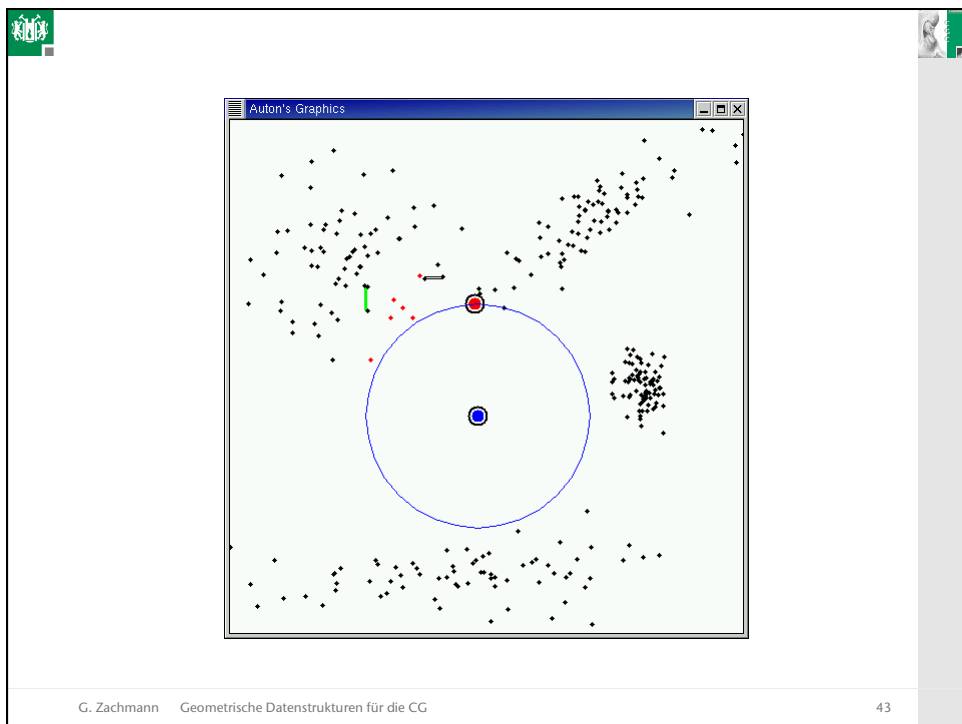
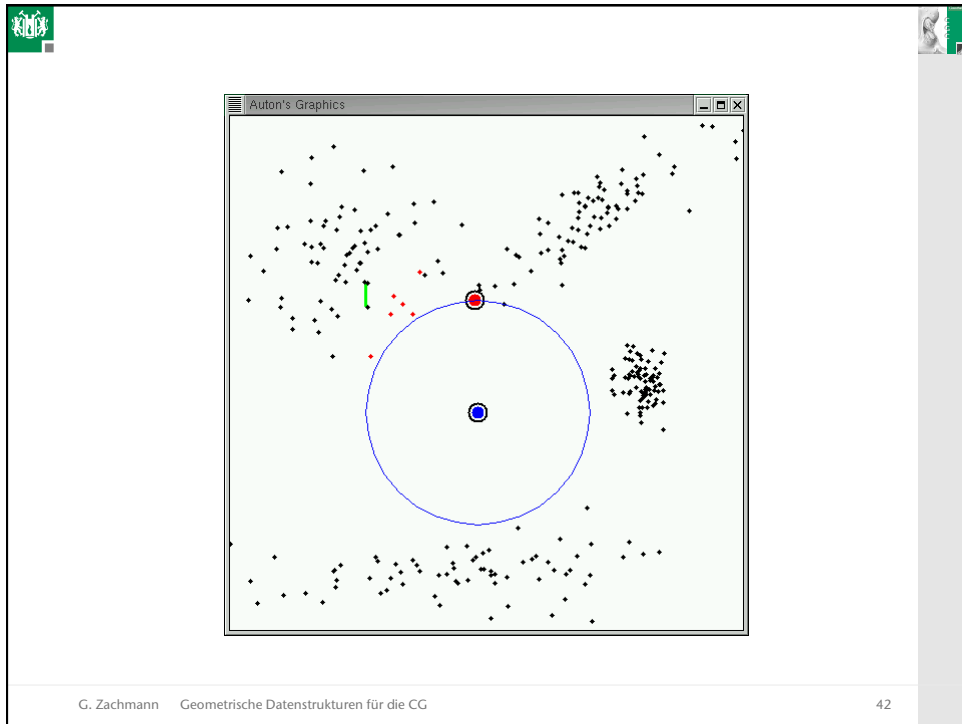
Animation of Nearest-Neighbor using kd-Trees

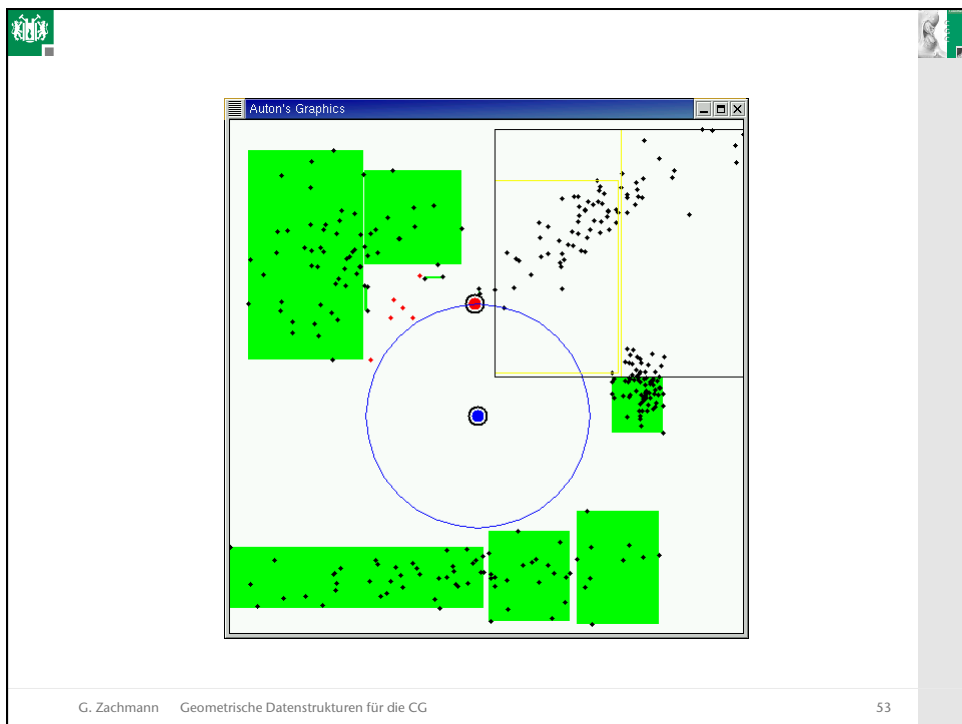
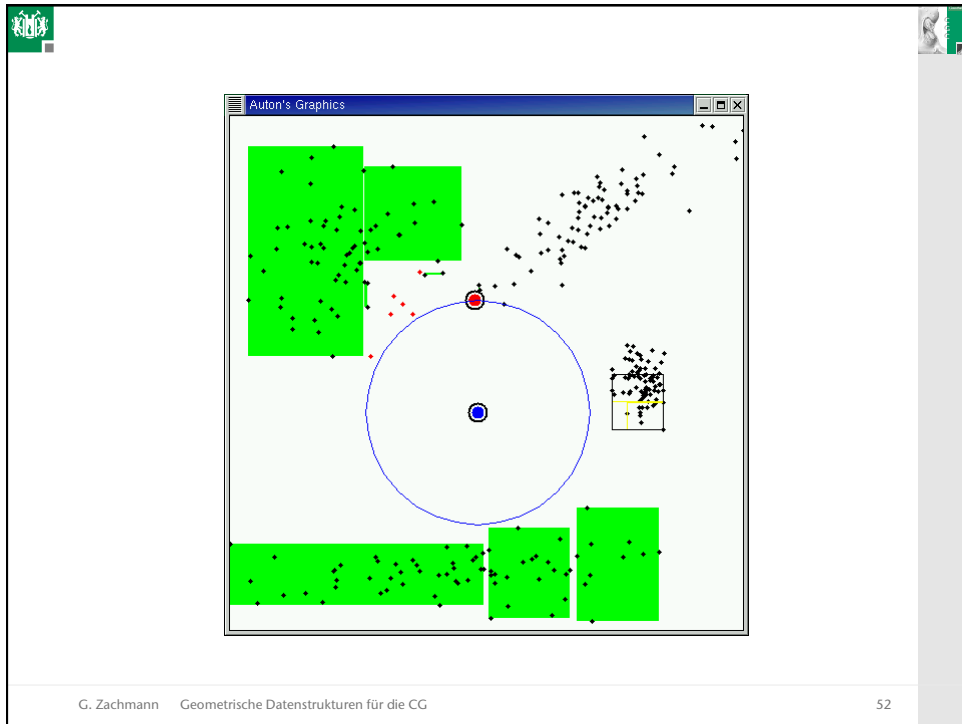


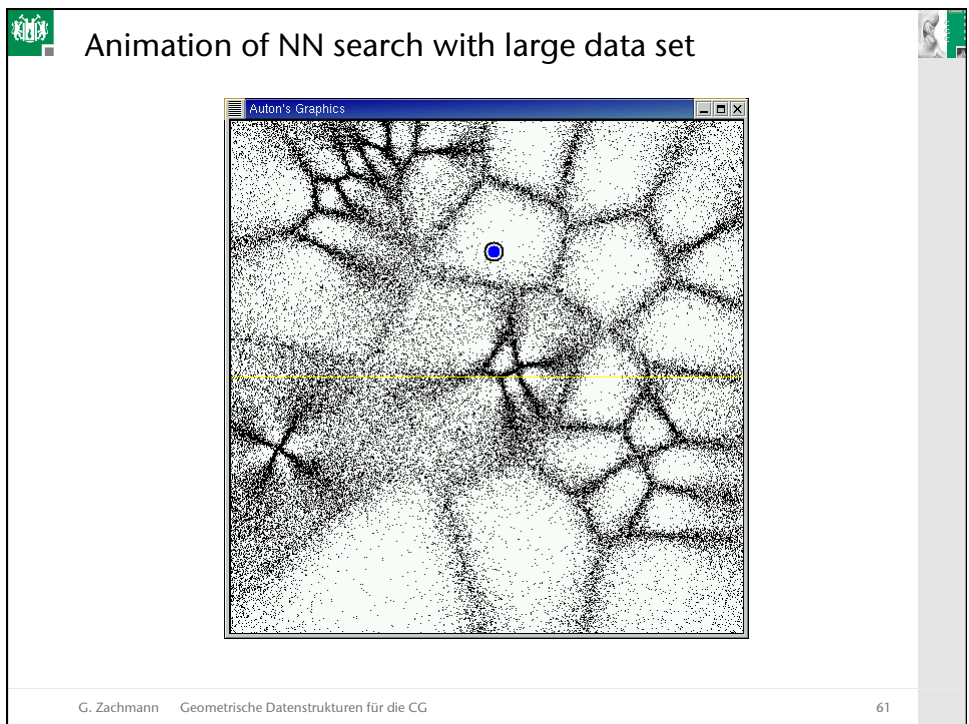
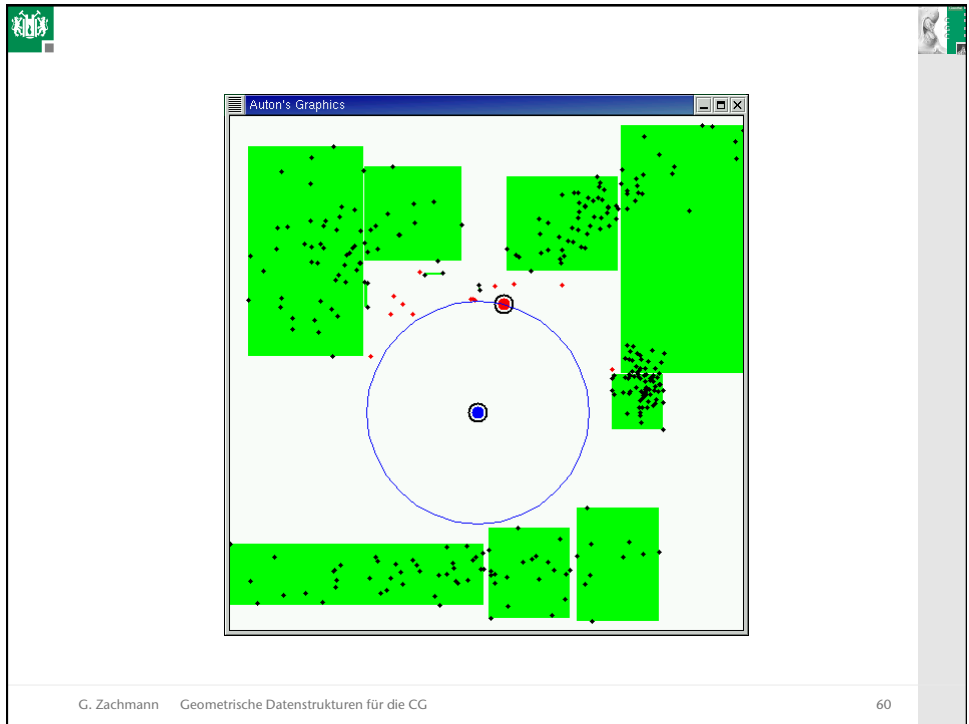
Andrew Moore, CMU

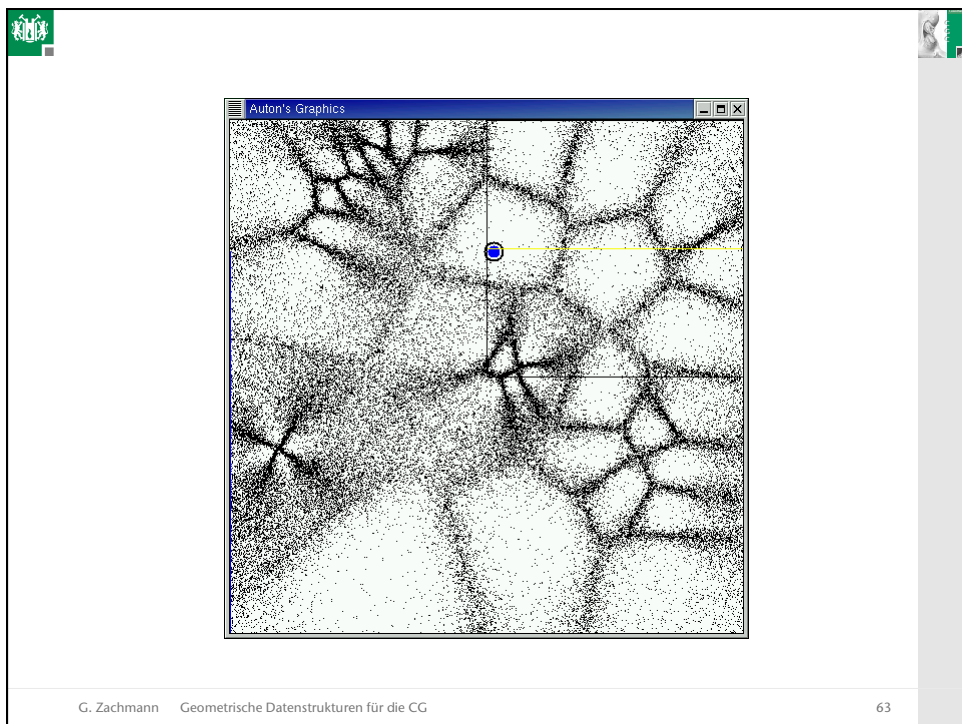
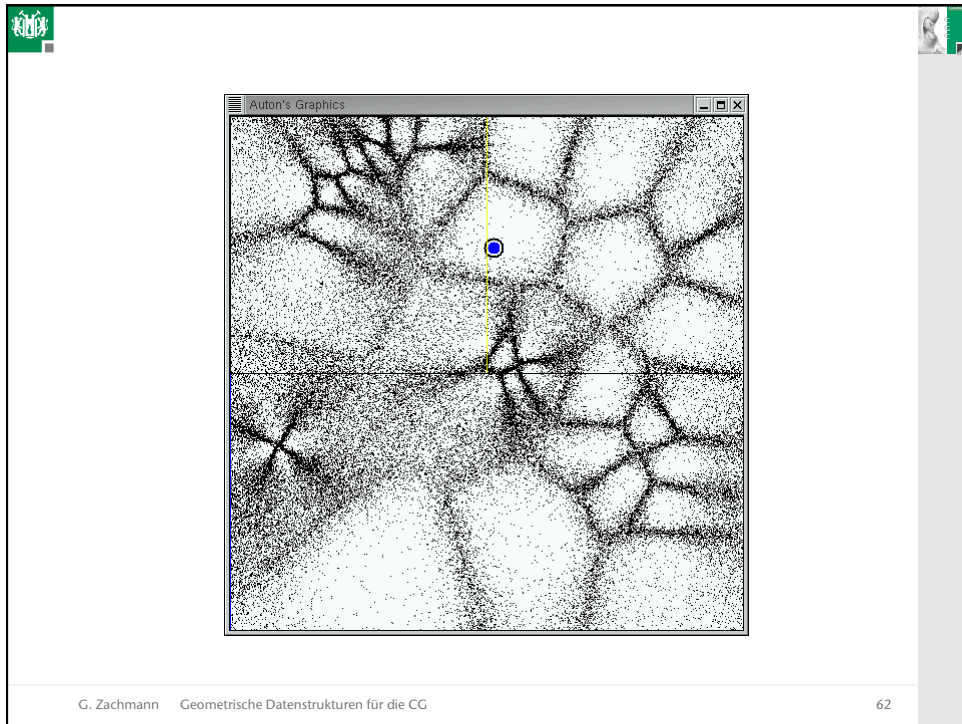
G. Zachmann Geometrische Datenstrukturen für die CG 29

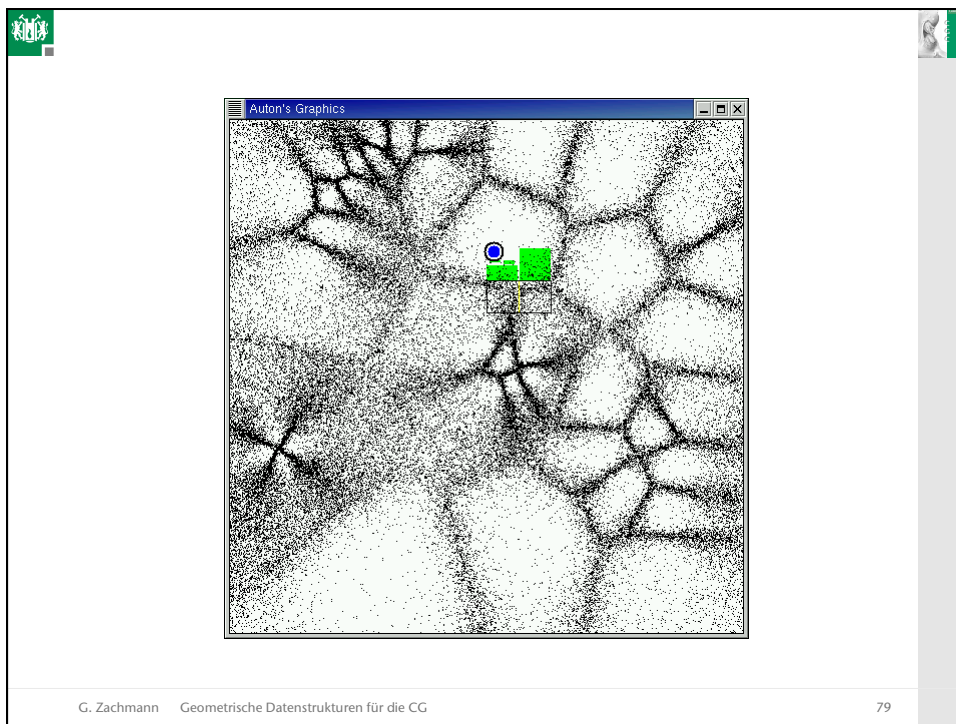
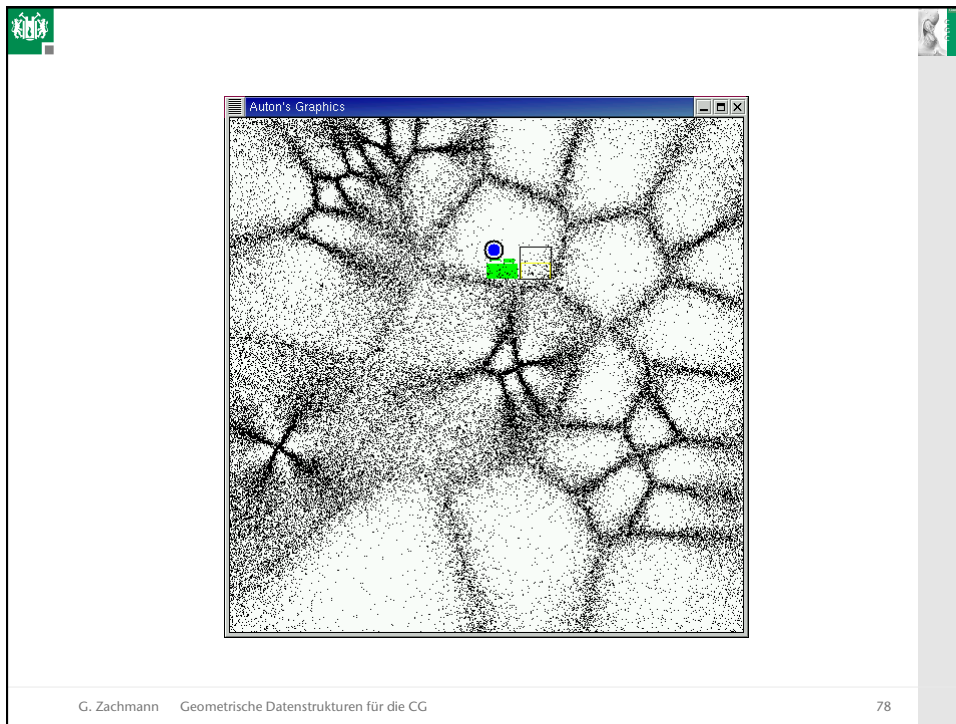


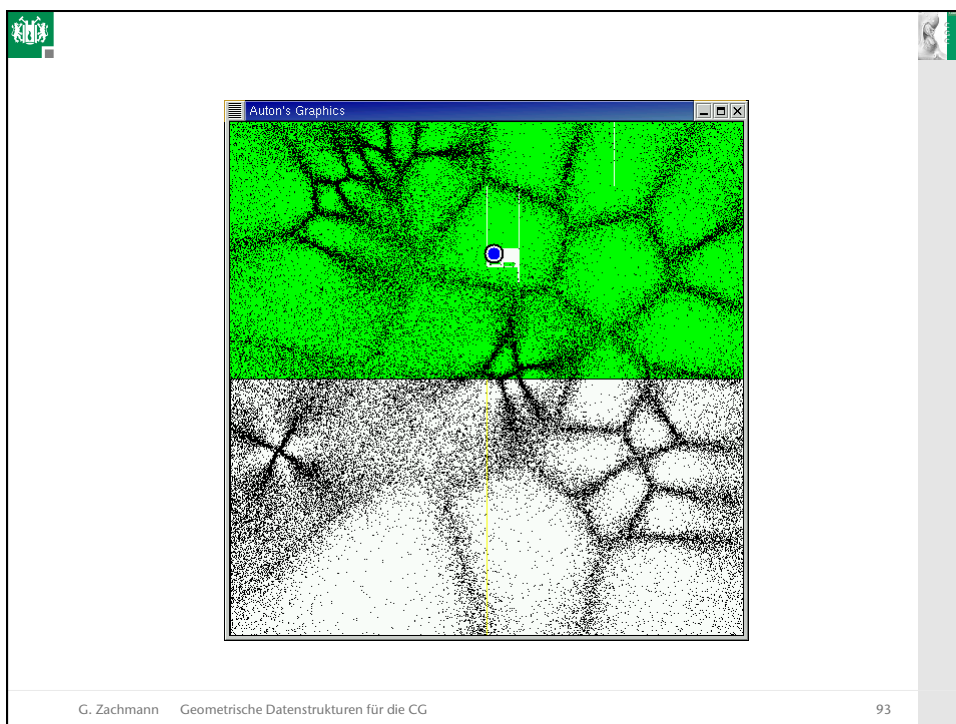
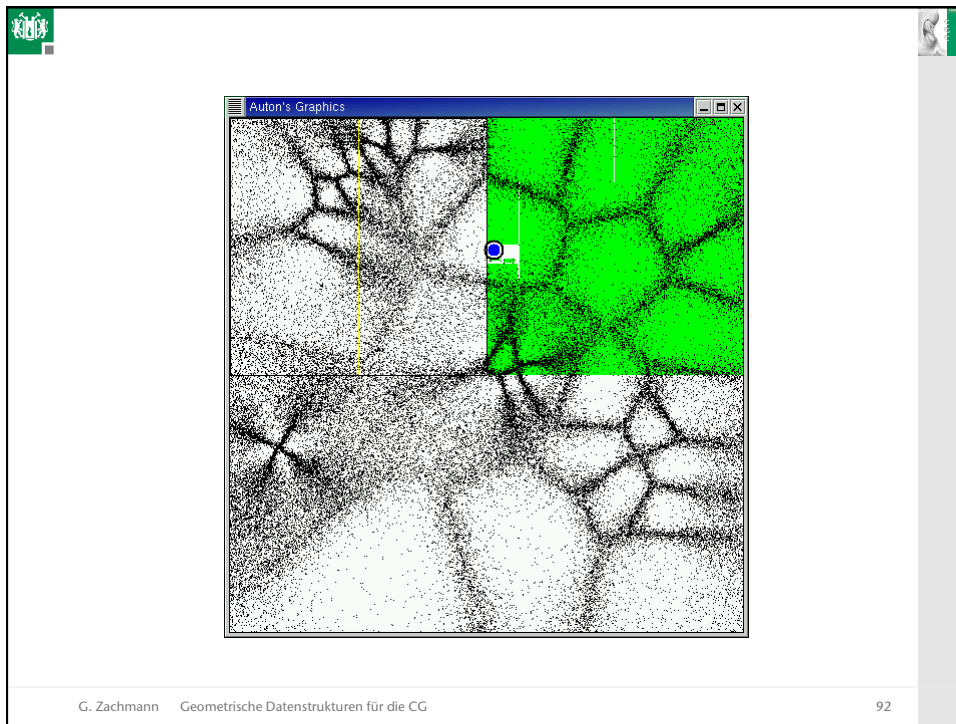


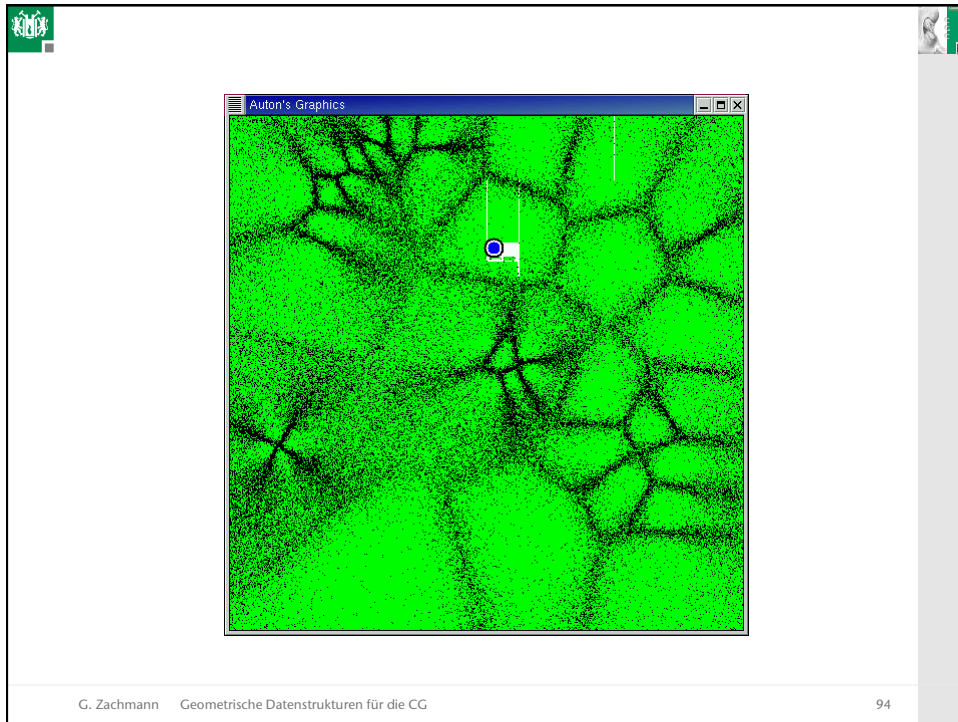












A worst-case for NN-Search using kd-Trees

Gutartiger Fall

Bösartiger Fall

Alle weißen Blätter muß der NN-Algorithmus besuchen!

Zum Verhalten von $\log^d(n)$

$\log^5(n)$

$\log^5(n)$

$\log^5(n)$

Der Algorithmus für die ANN-Suche ist also besser (asymptotisch) als brute-force-mäßig alle n Punkte zu besuchen und deren Abstand zum Query-Punkt q zu berechnen.

G. Zachmann Geometrische Datenstrukturen für die CG
96

Texture Synthesis

Wei & Levoy

G. Zachmann Geometrische Datenstrukturen für die CG
97

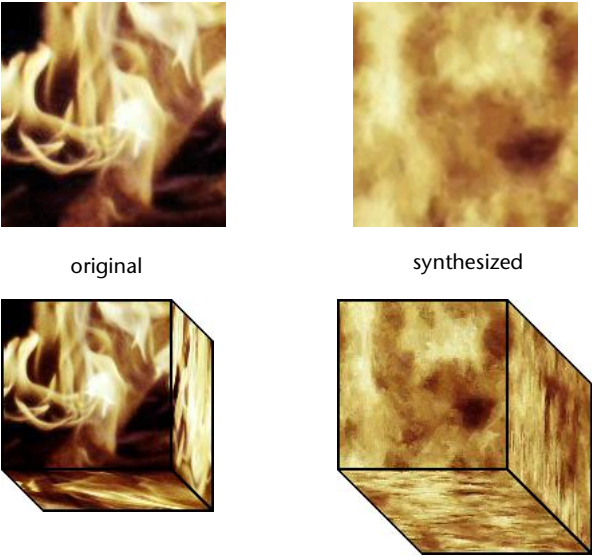
Wei & Levoy



Aspen Trees Harris and Love, Inc.

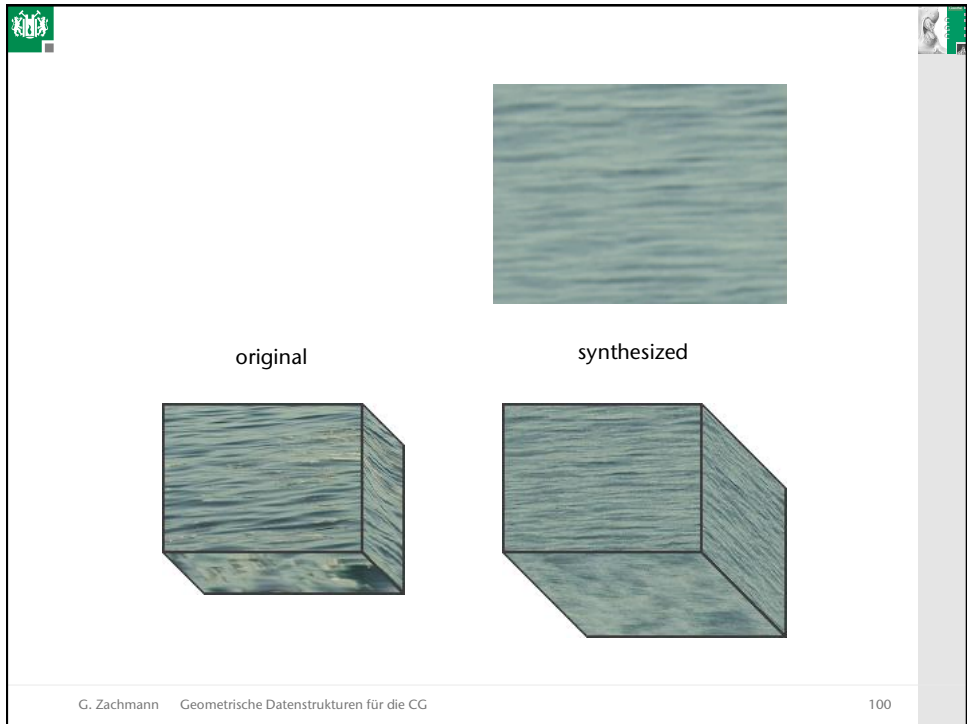
Aspen Trees Harris and Love, Inc.

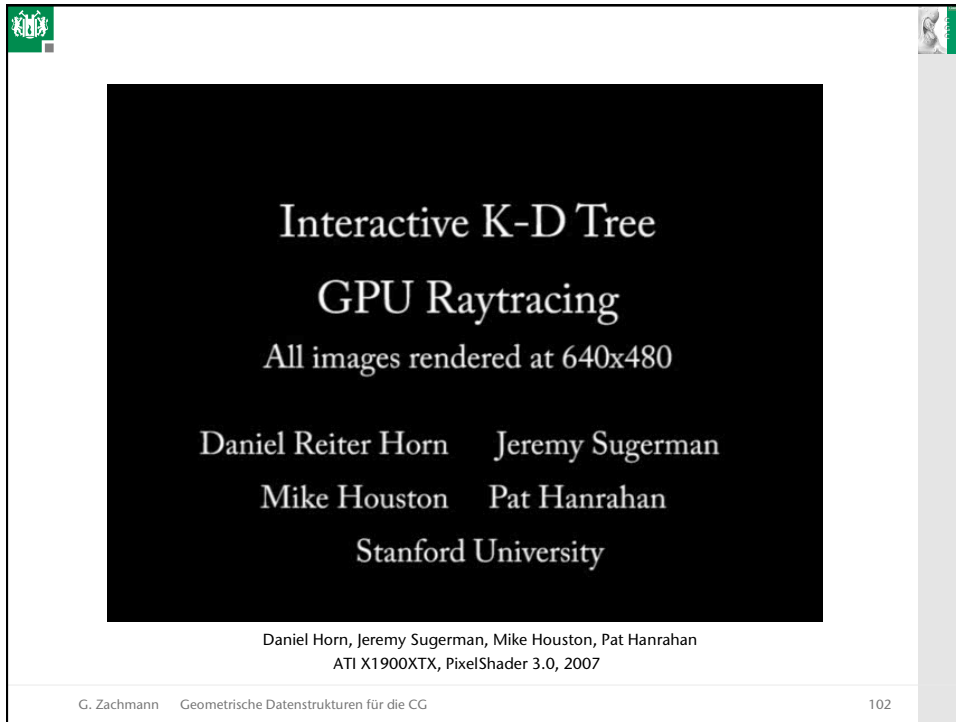
G. Zachmann Geometrische Datenstrukturen für die CG 98



original synthesized

G. Zachmann Geometrische Datenstrukturen für die CG 99



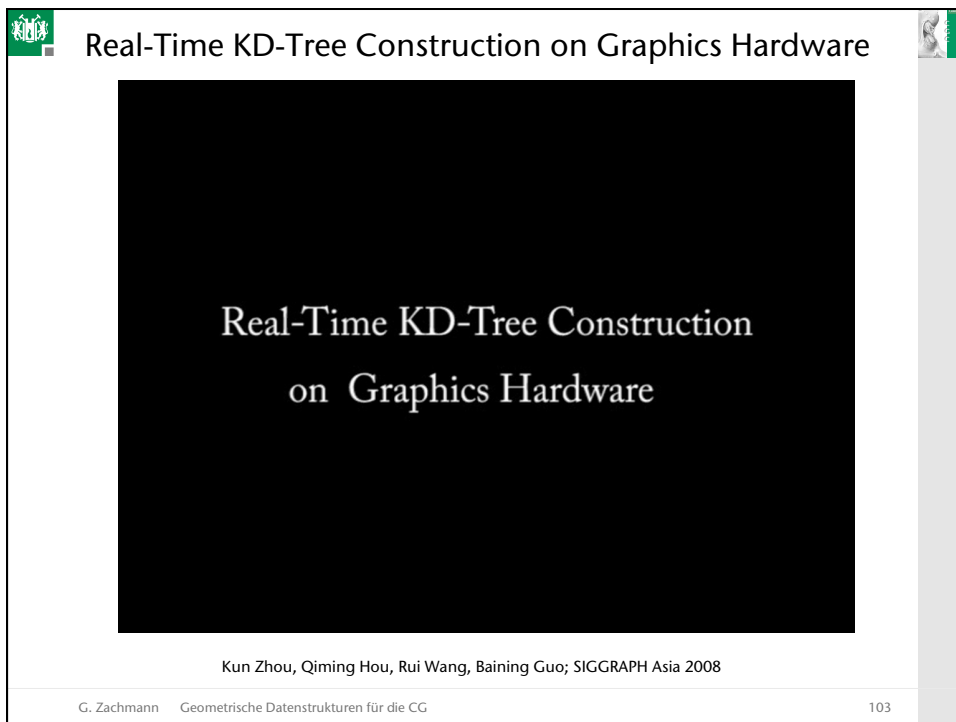


Interactive K-D Tree
GPU Raytracing
All images rendered at 640x480

Daniel Reiter Horn Jeremy Sugerman
Mike Houston Pat Hanrahan
Stanford University

Daniel Horn, Jeremy Sugerman, Mike Houston, Pat Hanrahan
ATI X1900XTX, PixelShader 3.0, 2007

G. Zachmann Geometrische Datenstrukturen für die CG 102



Real-Time KD-Tree Construction on Graphics Hardware

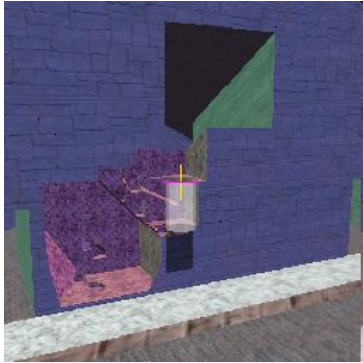
Real-Time KD-Tree Construction
on Graphics Hardware

Kun Zhou, Qiming Hou, Rui Wang, Baining Guo; SIGGRAPH Asia 2008

G. Zachmann Geometrische Datenstrukturen für die CG 103

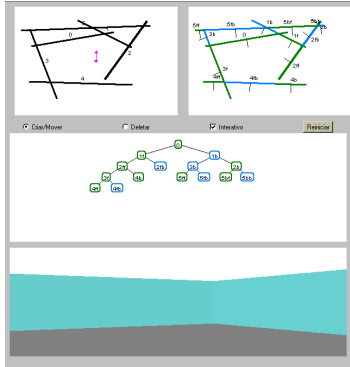
Applications of the BSP

Boolean Operations



Stan Melax

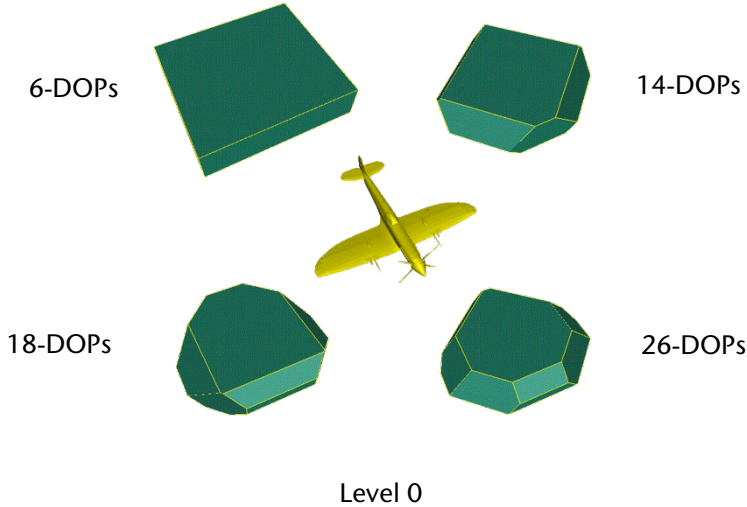
Painter's Algorithm



Paton J. Lewis

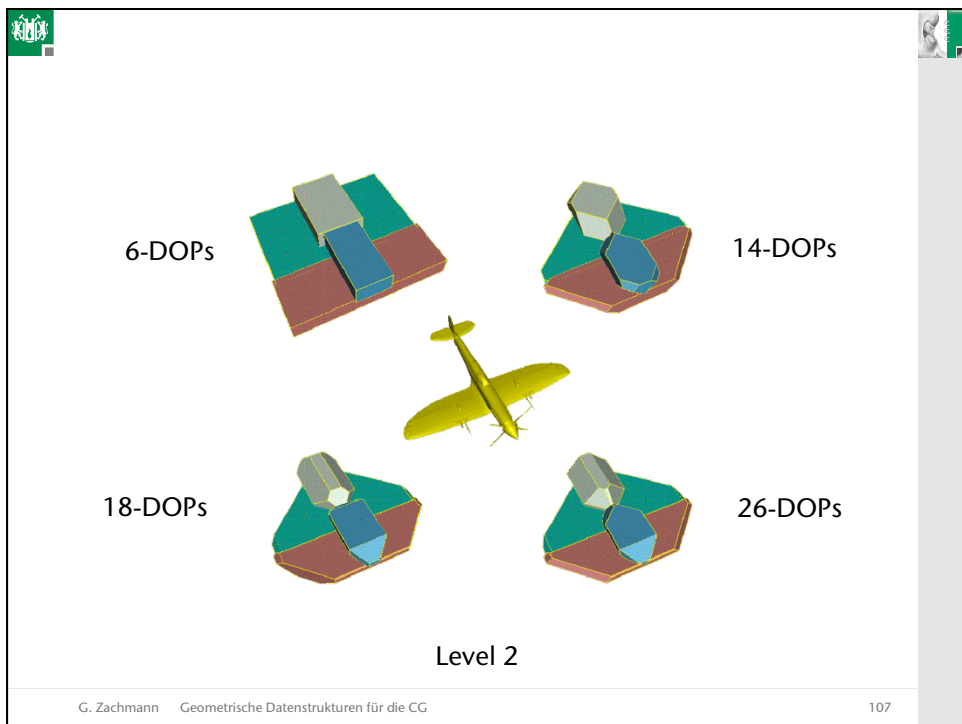
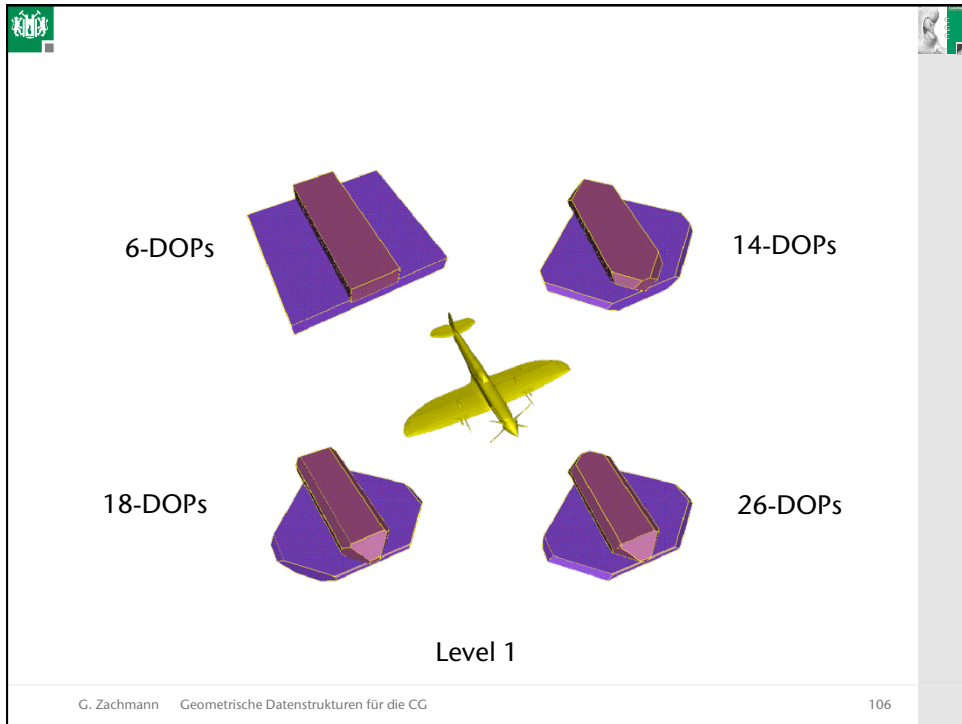
G. Zachmann Geometrische Datenstrukturen für die CG
104

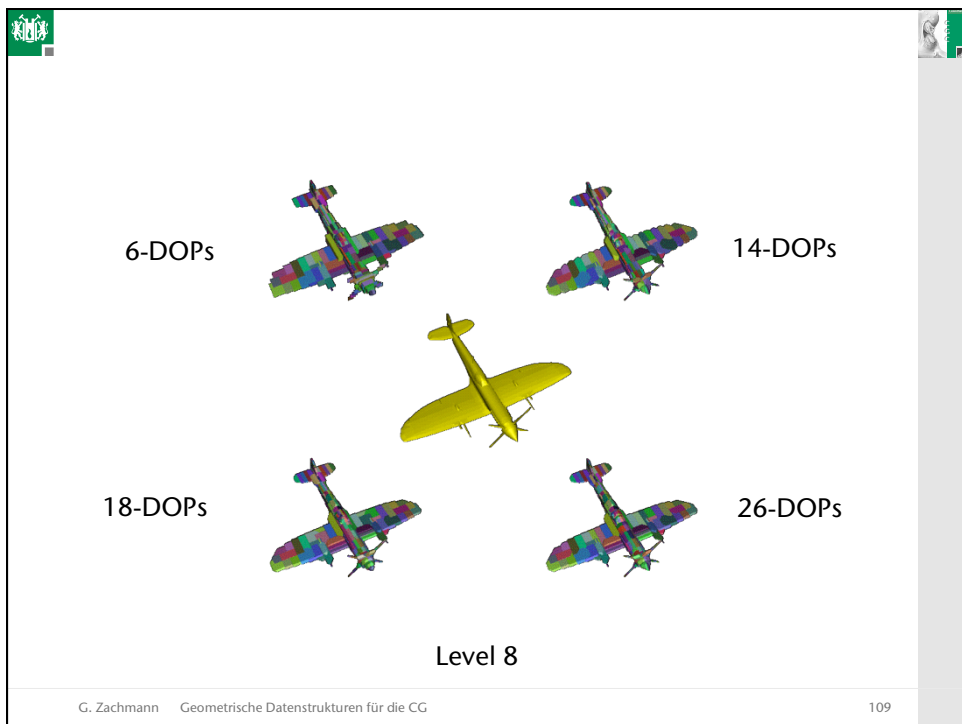
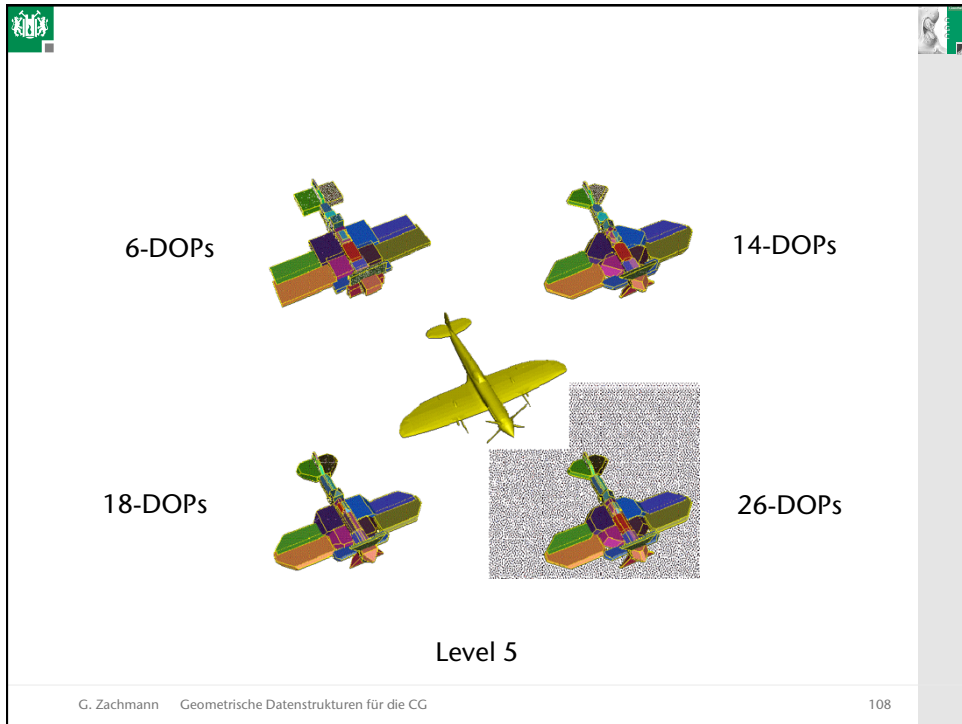
BVH mit k -DOPs

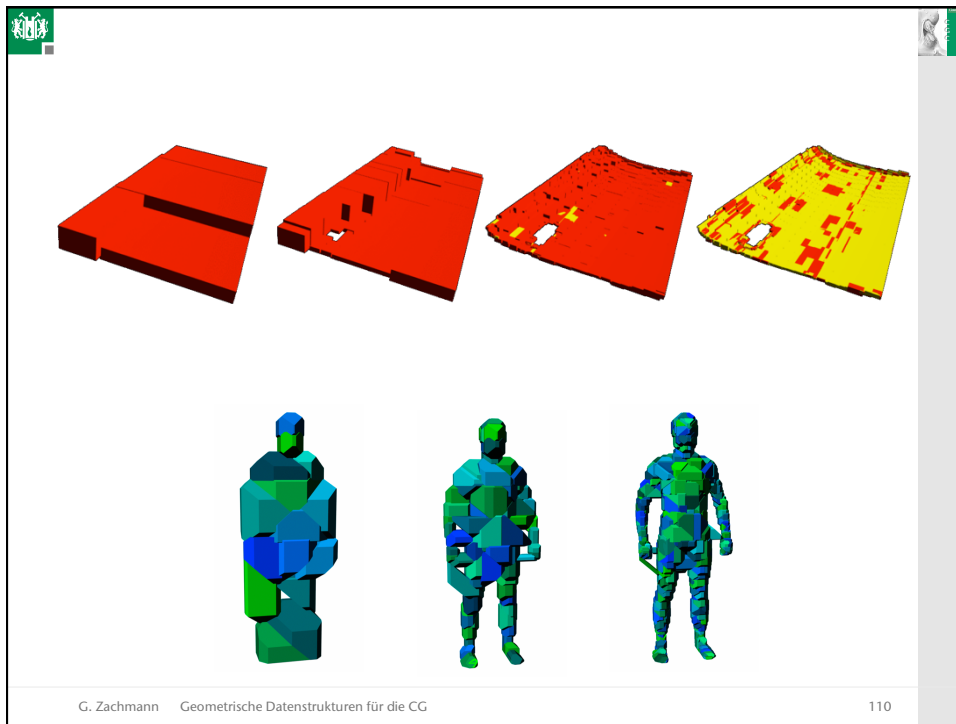


Level 0

G. Zachmann Geometrische Datenstrukturen für die CG
105








Hierarchische Kollisionserkennung mittels BVHs


```

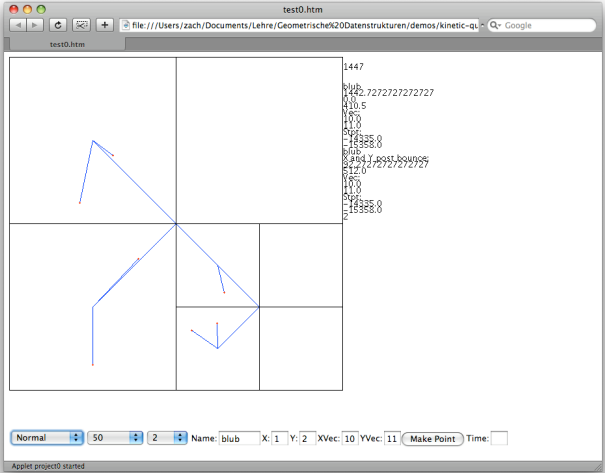
    traverse( X, Y )
    if X,Y do not overlap then
        return
    if X,Y are leaves then
        check polygons
    else
        for all children pairs do
            traverse( Xi, Yj )
    
```

The diagram illustrates BVH traversal. It shows two trees: a blue tree with root A and children B, C, D, E, F, G; and a red tree with root 1 and children 2, 3, 4, 5, 6, 7. To the right, a green shape is shown with bounding boxes B^P and B^Q , and sub-bounding boxes B^P_1 , B^P_2 , B^Q_1 , and B^Q_2 .

 Applications using Distance Fields

G. Zachmann Geometrische Datenstrukturen für die CG 112

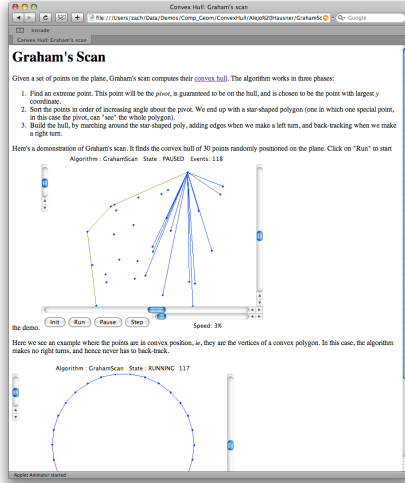
 Kinetic Quadtree Demo



Normal 50 2 2 Name: blub X: 1 Y: 2 XVec: 10 YVec: 11 Make Point Time: Applet project0 started

G. Zachmann Geometrische Datenstrukturen für die CG 113

Convex Hull Demos in 2D



Graham's Scan

Given a set of points on the plane, Graham's scan computes their [convex hull](#). The algorithm works in three phases:

1. Find an extreme point. This point will be the *pivot*, is guaranteed to be on the hull, and is chosen to be the point with largest *y* coordinate.
2. Sort the points in order of increasing angle about the pivot. We end up with a star-shaped polygon (one in which one special point, in this case the *pivot*, can "see" the whole polygon).
3. Build the hull, by traversing around the star-shaped poly, adding edges when we make a left turn, and back-tracking when we make a right turn.

Here's a demonstration of Graham's scan. It finds the convex hull of 30 points randomly positioned on the plane. Click on "Run" to start the demo.

Algorithm: GrahamScan State: PAUSED Iterations: 118

Speed: 3K

Buttons: Run, Pause, Step

Here we see an example where the points are in convex position, i.e., they are the vertices of a convex polygon. In this case, the algorithm makes no right turns, and hence never has to back-track.

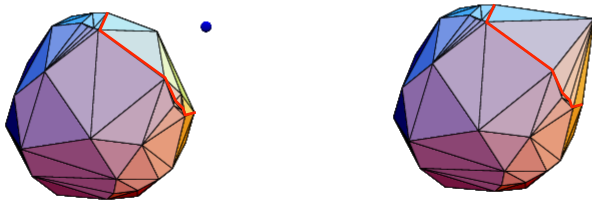
Algorithm: GrahamScan State: RUNNING Iterations: 117

Alejo Hausner - http://www.cs.princeton.edu/~ah/alg_anim/version1/GrahamScan.html

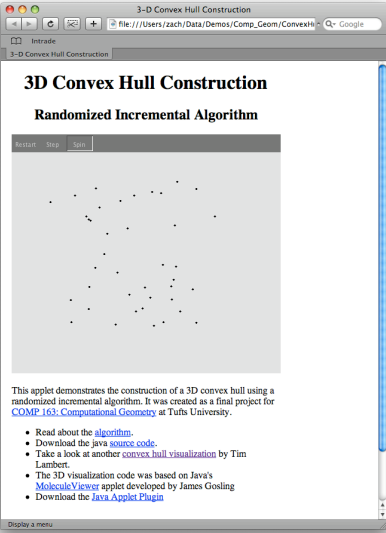
G. Zachmann Geometrische Datenstrukturen für die CG 114

Convex Hull in 3D

- Ein Schritt des inkrementellen Algorithmus':



G. Zachmann Geometrische Datenstrukturen für die CG 115



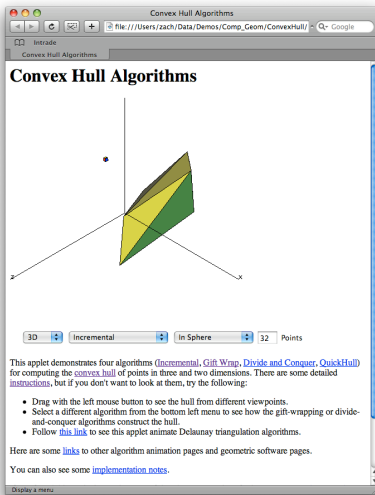
3-D Convex Hull Construction
Randomized Incremental Algorithm

This applet demonstrates the construction of a 3D convex hull using a randomized incremental algorithm. It was created as a final project for [COMP 163: Computational Geometry](#) at Tufts University.

- Read about the [algorithm](#).
- Download the [java source code](#).
- Take a look at another [convex hull visualization](#) by Tim Lambert.
- The 3D visualization code was based on Java's [Molensk's Java](#) applet developed by James Gosling
- Download the [Java Applet Plugin](#)

Michael Horn - <http://www.eecs.tufts.edu/~mhorn01/comp163/>

G. Zachmann Geometrische Datenstrukturen für die CG 116



Convex Hull Algorithms

3D Incremental In Sphere 32 Points

This applet demonstrates four algorithms ([Incremental](#), [Gift Wrap](#), [Divide and Conquer](#), [QuickHull](#)) for computing the [convex hull](#) of points in three and two dimensions. There are some detailed [instructions](#), but if you don't want to look at them, try the following:

- Drag with the left mouse button to see the hull from different viewpoints.
- Select a different algorithm from the bottom left menu to see how the gift-wrapping or divide-and-conquer algorithms construct the hull.
- Follow [this link](#) to see this applet animate Delaunay triangulation algorithms.

Here are some [links](#) to other algorithm animation pages and geometric software pages.

You can also see some [implementation notes](#).

Tim Lambert - <http://www.cse.unsw.edu.au/~lambert/java/3d/hull.html>

G. Zachmann Geometrische Datenstrukturen für die CG 117

Simplification of Urban Models

**Legible Simplification
of Large Textured
Urban Models**

Remco Chang,
Thomas Butkiewicz,
Caroline Ziemkiewicz,
Zachary Wartell, Nancy Pollard,
and William Ribarsky

Remco Chang, Thomas Butkiewicz, Caroline Ziemkiewicz, Zachary Wartell, Nancy Pollard, William Ribarsky

G. Zachmann Geometrische Datenstrukturen für die CG 118

Convex Collision Detection

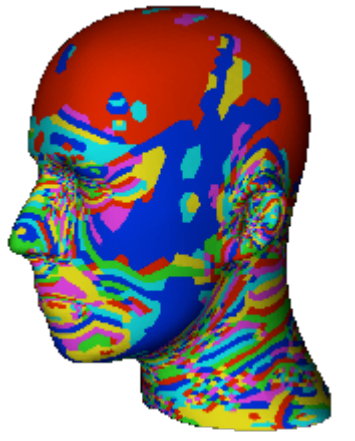
**Incremental
Collision Detection
for Polygonal Models**

Madhav K. Ponamgi
Jonathan D. Cohen
Ming C. Lin
Dinesh Manocha


Achtung: der hier demonstrierte Algo ist in Wahrheit
etwas komplexer als der in der Vorlesung dargestellte!
(aber möglicherweise nicht schneller ...)

G. Zachmann Geometrische Datenstrukturen für die CG 119

Convex Surface Decomposition



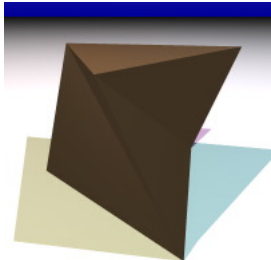
Zerlegung in
konvexe Surface-Patches



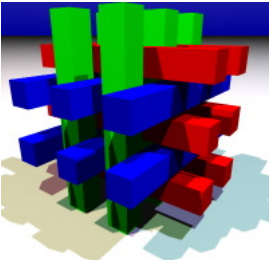
Konvexe Stücke auf einem
mittleren Level der Hierarchie
(grün = orig. Fläche, rot = freie Fläche,
gelb = "contained")

G. Zachmann Geometrische Datenstrukturen für die CG
120

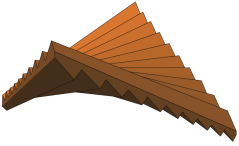
Untetrahedralizable Objects



Schönhardt's
Polyeder
(1928)

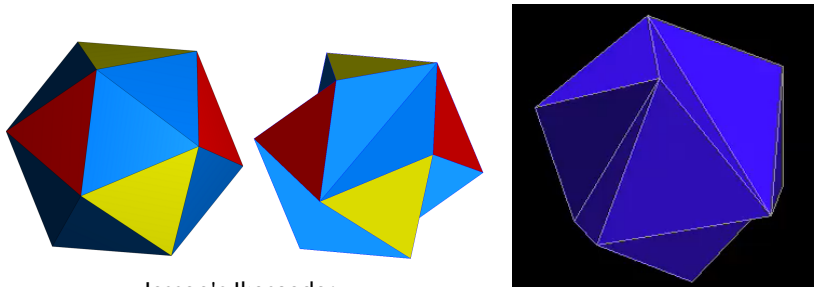


Thurston-
Polyeder
(1971)



Chazelle's
Polyeder
(1984)

G. Zachmann Geometrische Datenstrukturen für die CG
121



Jessen's Icosaeder

G. Zachmann Geometrische Datenstrukturen für die CG 122

The image displays three different renderings of Jessen's Icosahedron. The first two are multi-colored, showing the 12 triangular faces in shades of red, blue, yellow, and green. The third is a monochromatic blue version. The text 'Jessen's Icosaeder' is centered below the first two. The footer contains the author's name 'G. Zachmann', the course title 'Geometrische Datenstrukturen für die CG', and the page number '122'.