Sichtbarkeitsberechnung

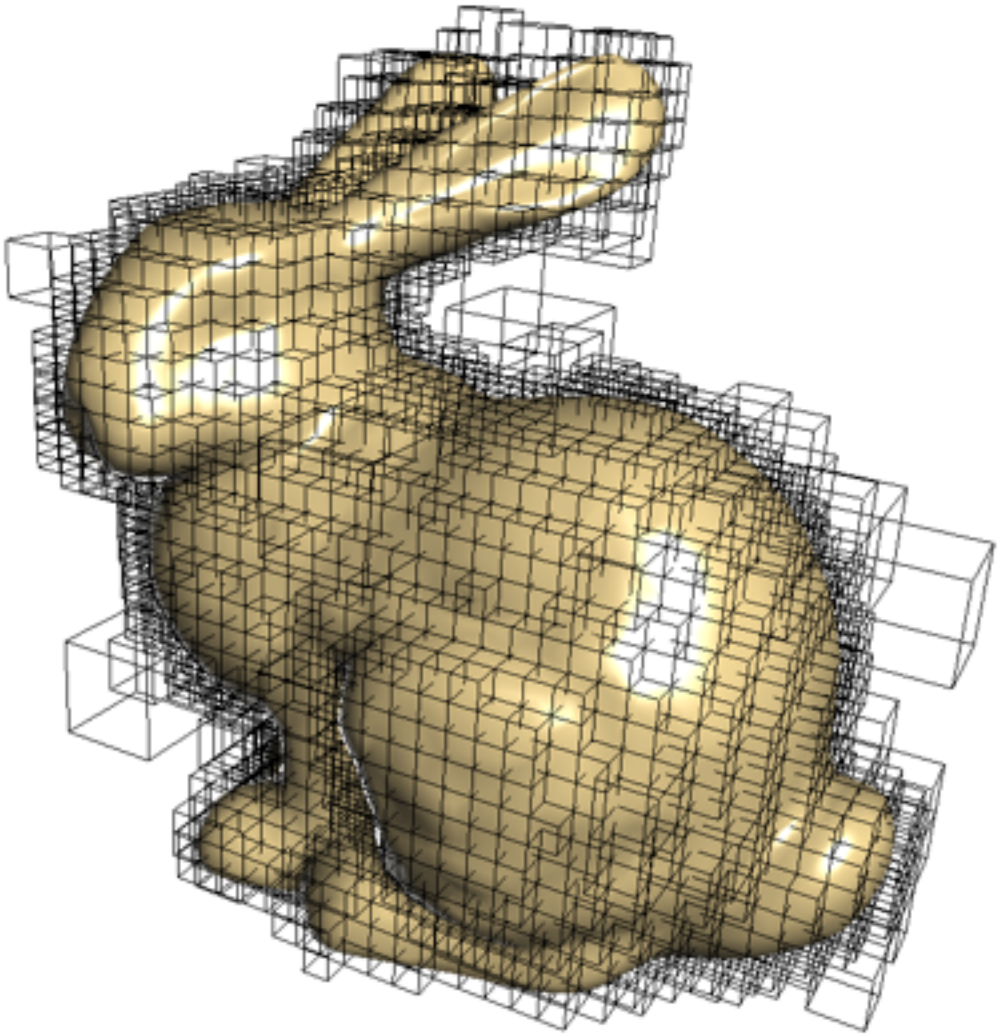
Einführung in die Computergrafik

Wiederholung

- Schatten
  - Schattenvolumen
- Globale Beleuchtungsrechnung
  - Raytracing

Ausblick

Worum gehts?



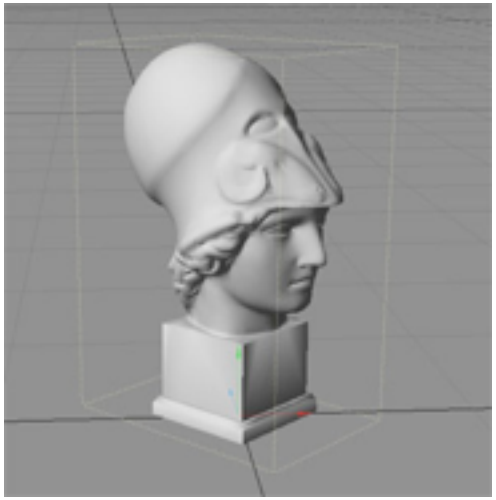
Agenda

- Hüllkörper
- Hüllkörper-Hierarchien
    - Octrees (Quadtrees)
    - BSP-Bäume
- Sichtbarkeitsberechnung

Hüllkörper

Bounding Volume Hierarchien

- Hierarchien von Hüllkörpern
- Hüllkörper (engl. bounding volume)
    - einfacher geometrischer Körper
    - der ein komplexes dreidimensionales Objekt oder einen komplexen Körper umschließt.
- Beispiele für Hüllkörper
    - Kugeln
    - Quader (Würfel)
    - Hyperebenen
    - Polyeder



Hüllkörper um 3D-Modell
(Quelle: [1])

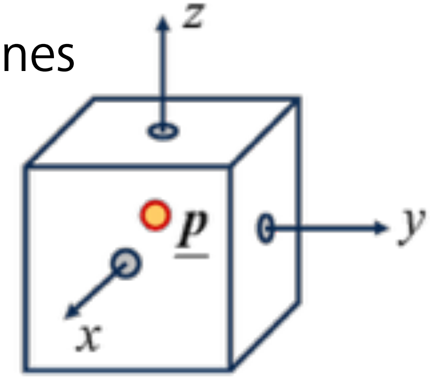
Würfel als Hüllkörper

- Primitiv als Menge
  - Spezifikation von Primitiven als Teilmenge von R³

$$W = \{p = \begin{pmatrix} p_x \\ p_y \\ p_z \end{pmatrix} \in \mathbb{R}^3 | \forall p_i \in [-1, 1]\}$$

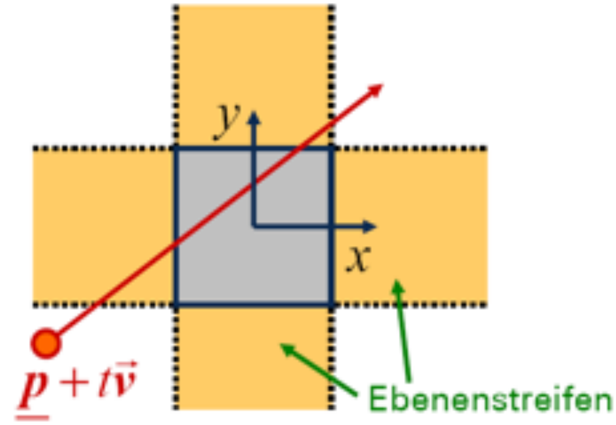
- Punktinklusionstest
  - Test, ob Punkt im Innern des Primitives liegt bzw. dazu gehört
  - aus Mengendefinition ablesbar

- Anwendungen
  - Kollisionstest
  - Markieren aller vom Primitiv überdeckten Zellen eines Volumengitters



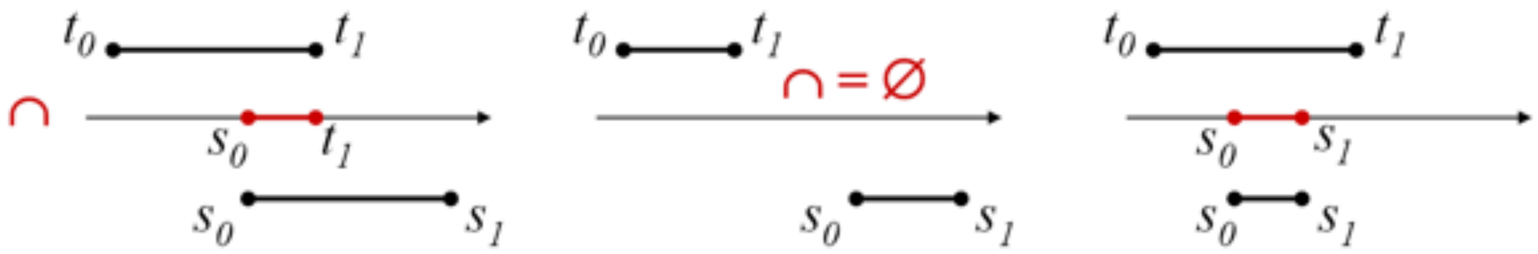
Schnitt Strahl-Würfel

- Würfel ist konvex
    - daher maximal ein Eintritts- und ein Austrittspunkt
- die Ebenen durch 2 gegenüberliegende Würfelflächen spannen 3 Ebenenstreifen auf
    - Schnittmenge = Würfel
- Idee:
    - Schnitt mit Ebenenstreifen
    - dann Schnitt mit achsenparallelen Ebenen
    - Mengendurchschnitt mit Intervallarithmetik auf Intervallen des Strahlparameters t



Schnitt Strahl-Würfel

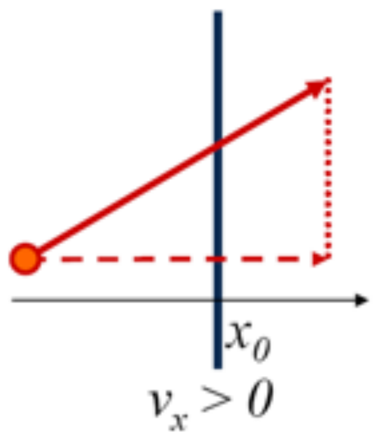
- Intervallarithmetik
- Reduktion auf 1D-Fall reduzieren
- damit: Reduktion der Objekte auf Intervalle
    - z.B: Strahlparameter
- Zeitintervall
    - $T = [t_0, t_1]$ mit $t_0 < t_1$ oder $T = \{\}$ (leeres Intervall)
- wichtig ist die Schnittoperation auf Intervallen:

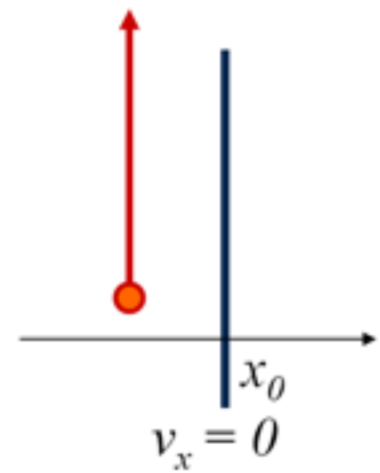


Schnitt Strahl-Würfel

- Schnitt mit achsenparalleler Ebene $x = x_0$
  - das Schnittproblem lässt sich auf eine Dimension reduzieren
  - Strahl beginnt bei $x(t = 0) = p_x$
  - x-Komponente wächst mit der Geschwindigkeit v_x
  - aus $x_0 = p_x + t_0 v_x$ ergibt sich der Schnittparameter t0 zu:

$$t_0 = \begin{cases} \dfrac{x_0 - p_x}{v_x} : v_x \neq 0 \\ undef : v_x = 0 \end{cases}$$





Schnitt Strahl-Würfel

- Beispiel
- Strahl

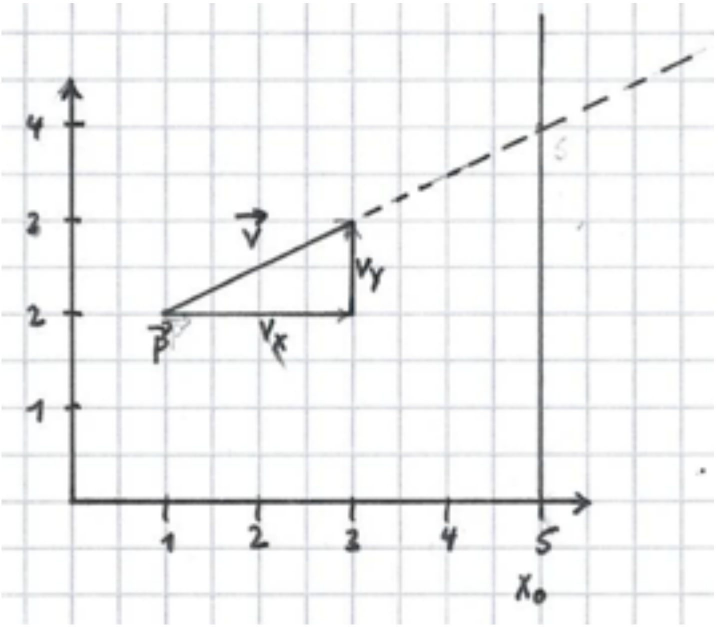
$$s(t) = \vec{p} + t\vec{v} = \begin{pmatrix} 1 \\ 2 \end{pmatrix} + t\begin{pmatrix} 2 \\ 1 \end{pmatrix}$$

- Schnittebene: $x_0 = 5$

$$t_0 = \begin{cases} \dfrac{x_0 - p_x}{v_x} : v_x \neq 0 \\ undef : v_x = 0 \end{cases}$$

$$t_0 = \frac{5-1}{2} = 2$$

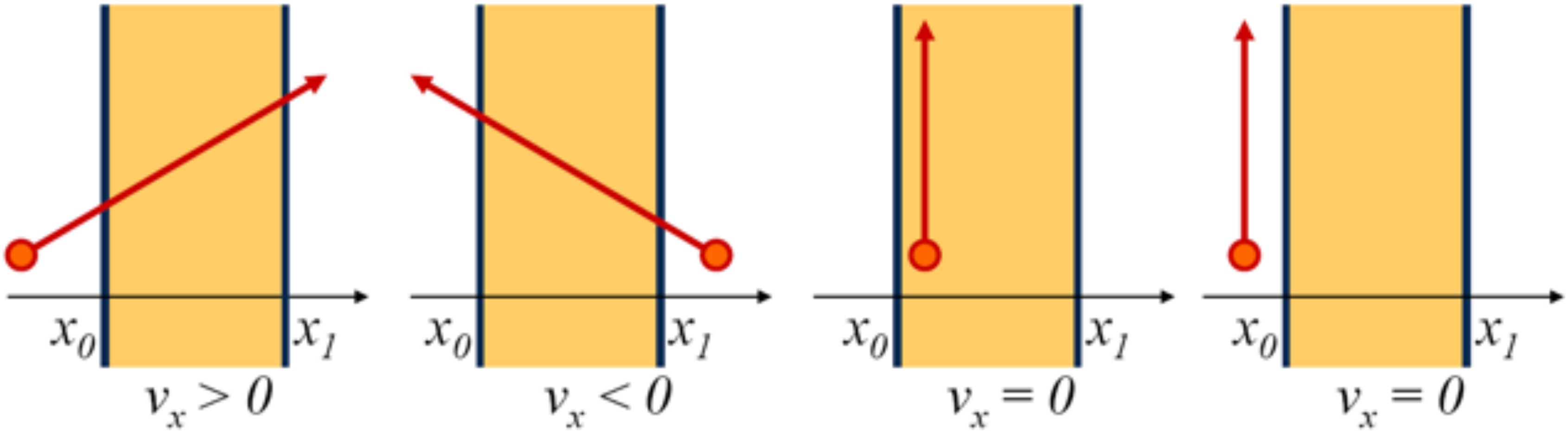
$$s(2) = \begin{pmatrix} 1 \\ 2 \end{pmatrix} + 2\begin{pmatrix} 2 \\ 1 \end{pmatrix} = \begin{pmatrix} 5 \\ 4 \end{pmatrix}$$



Schnitt Strahl-Würfel

- Schnitt mit achsenparallelem Ebenenstreifen $x \in [x_0, x_1]$
  - Intervall wird abhängig vom Vorzeichen von vx berechnet
  - weitere Fallunterscheidung bei Parallelität

$$[t_0, t_1] = \begin{cases} \left[\frac{x_0 - p_x}{v_x}, \frac{x_1 - p_x}{v_x}\right] : v_x > 0 \\ \left[\frac{x_1 - p_x}{v_x}, \frac{x_0 - p_x}{v_x}\right] : v_x < 0 \\ [-\infty, \infty] : v_x = 0 \wedge p_x \in [x_0, x_1] \\ \{\} : sonst \end{cases}$$

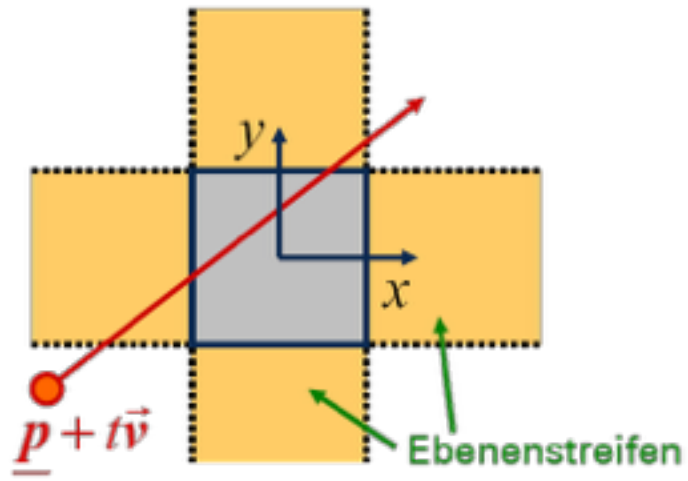


Schnitt Strahl-Würfel

- Schnitt Strahl-Würfel
  - man die Schnittintervalle $T_{x/y/z}$ mit den drei Ebenenstreifen (gemäß der vorigen Folie)
  - Schnittintervall T_W mit Würfel ergibt sich aus dem Schnitt über alle drei Intervalle:

$$T_W = T_X \cap T_Y \cap T_Z$$

  - abschließend muss geprüft werden, ob es ein
    - $t \in T_W$ gibt
    - mit $t > 0$



Übung: Schnitt Strahl-Box

- Berechnen Sie die Schnittpunkte zwischen dem Strahl

$$s = \vec{p} + t\vec{v} = \begin{pmatrix} 1 \\ 2 \end{pmatrix} + t \begin{pmatrix} 2 \\ 1 \end{pmatrix}$$

- und der Box mit den Eckpunkten (2,2) und (3,4)

- Erinnerung:

$$[t_0, t_1] = \begin{cases} \left[\frac{x_0 - p_x}{v_x}, \frac{x_1 - p_x}{v_x} \right] : v_x > 0 \\ \left[\frac{x_1 - p_x}{v_x}, \frac{x_0 - p_x}{v_x} \right] : v_x < 0 \\ [-\infty, \infty] : v_x = 0 \wedge p_x \in [x_0, x_1] \\ \{\} : sonst \end{cases}$$

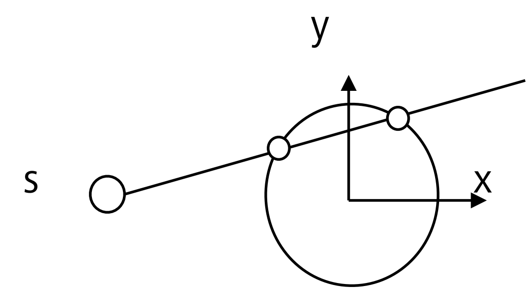
Kugel als Hüllkörper

- Strahlschnitt führt auf quadratische Gleichung mit 0, 1 oder 2 Lösungen
  - Kugel K

$$K = \{x \in R^3 \mid \| \overrightarrow{x} - \overrightarrow{m} \|^2 - r^2 = 0\}$$

  - Strahl s

$$s : \overrightarrow{p} + \lambda \overrightarrow{v}$$



  - Strahl in Kugelgleichung:

$$\| \overrightarrow{p} + \lambda \overrightarrow{v} - \overrightarrow{m} \|^2 - r^2 = 0$$

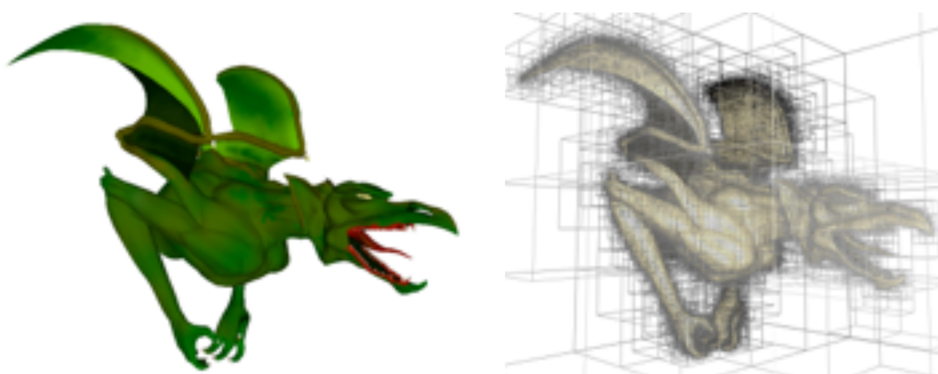
Übung: Schnitt Strahl-Kugel

- siehe Globale Beleuchtungsrechnung, Raytracing

Octree

Octree

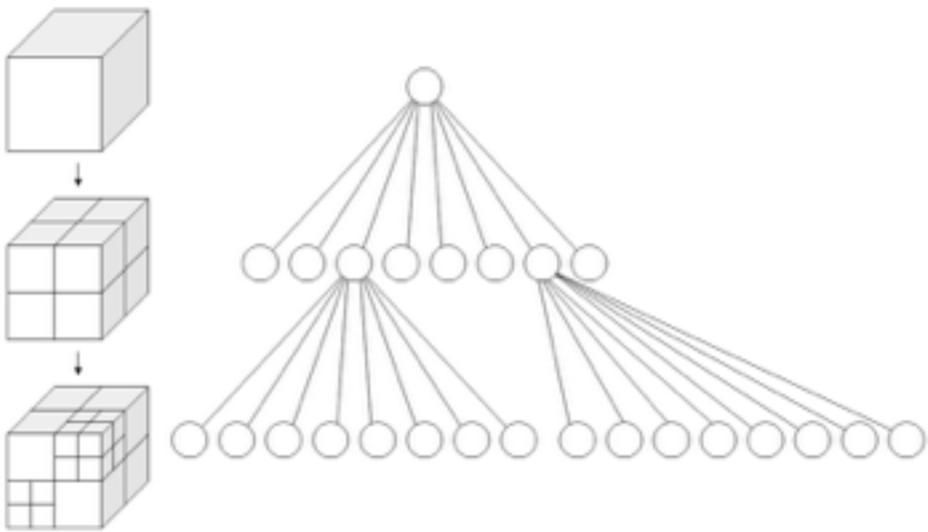
- Datenstruktur in der Computergrafik
- von lat. octo „acht“, und engl. tree „Baum“
- jeder Knoten (Würfel) hat ...
  - keinen
  - oder acht Kinder
- Wurzelknoten beinhaltet alle Daten



3D-Modell und zugehöriger Octree (Quelle: [2])

Octree

- Konstruktion
- erstelle umschließenden Würfel
- solange (Würfel zu grob)
    - unterteile Knoten in acht Kindwürfel



Aufbau eines Octrees (Quelle: [3])

Octree

- Unterteilungskriterien
  - maximale Tiefe
  - maximale Anzahl von Elementen in Blattknoten

- Achtung: ein Objekt kann auch in mehreren Knoten sein

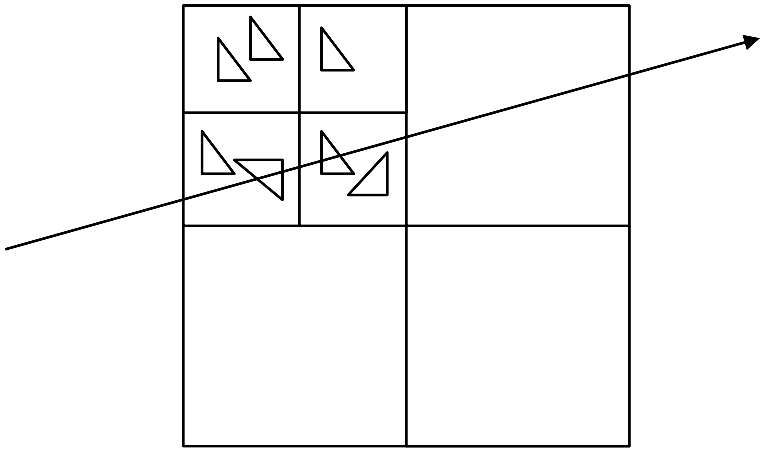
Octree

- Verwendung
- Beispiel
    - Octree aus Dreiecken
    - Aufgabe: Finde alle Dreiecke, die vom Strahl geschnitten werden
- Naive Lösung: Schnitt aller Dreiecke mit Strahl
    - Laufzeit $O(n)$, n = Anzahl der Dreiecke

Octree-Traversierung

- **Pseudocode:**

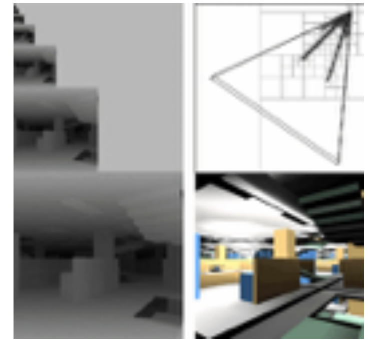
```
function traverse(ray, box)
        if (!intersect(ray, box))
                return
        for each triangle in box
                intersect(ray, triangle)
        for each childbox
                traverse(ray, childbox)
```



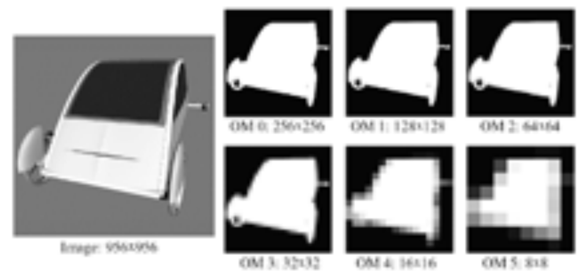
- Hinweis: ein Octree im 2D ist ein Quadtree
  - 4 Kindquadrate statt 8 Kinnwürfel

Occlusion Culling: Ansätze

- Hierarchischer Tiefenpuffer
  - *Greene, Ned, Michael Kass, and Gavin Miller, "Hierarchical Z-Buffer Visibility", Computer Graphics (SIGGRAPH 93 Proceedings), pp. 231-238, August 1993.*



- Hierarchische Occlusion-Map
  - *Zhang, H., D. Manocha, T. Hudson, and K.E. Hoff III, "Visibility Culling using Hierarchical Occlusion Maps", Computer Graphics (SIGGRAPH 97 Proceedings), pp. 77-88, August 1997. http://www.cs.unc.edu/~zhangh/hom.html*

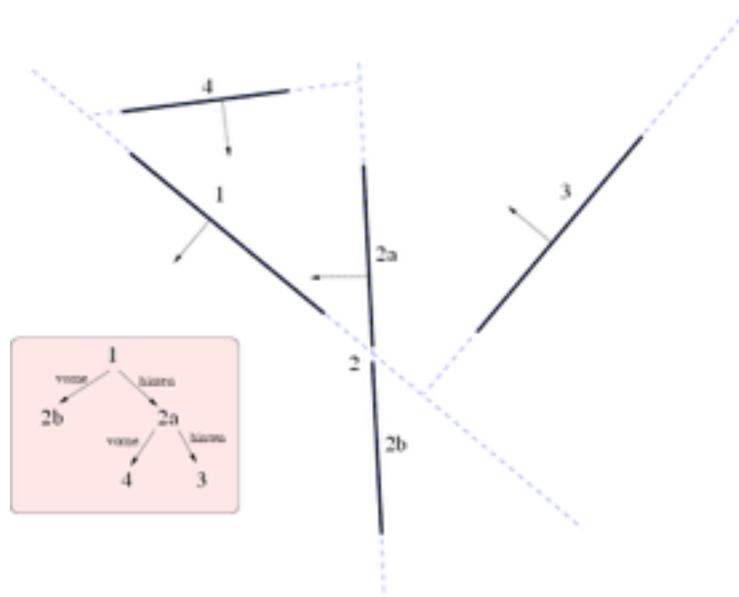


- *und weitere*

Binary Space Partition

Binary Space Partion (BSP)

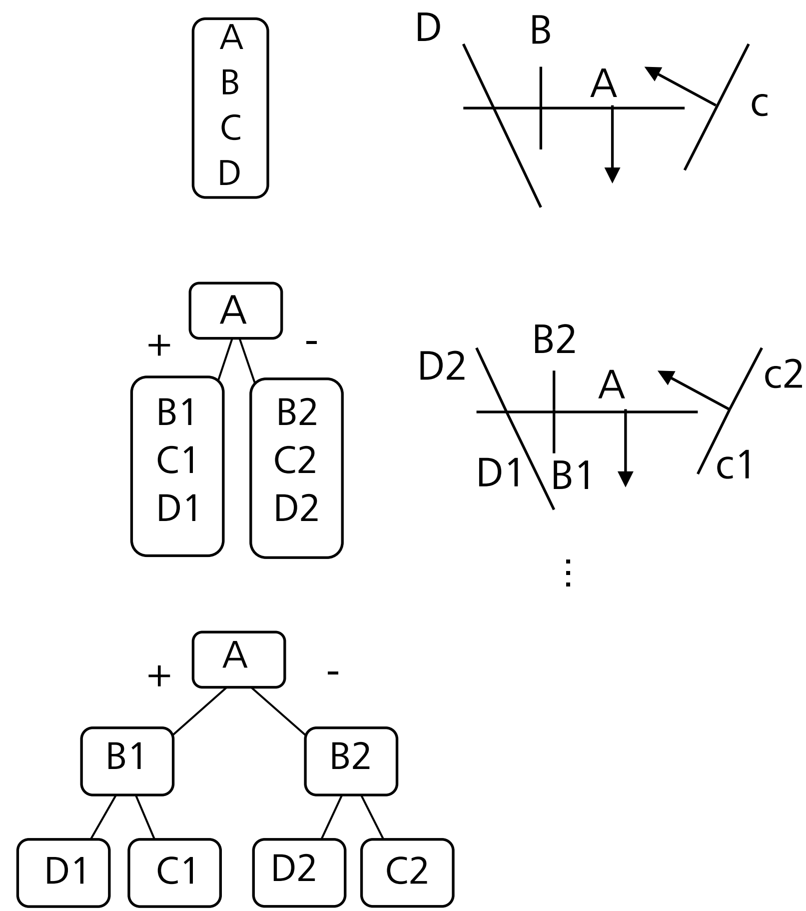
- Unterteilung durch Menge von Hyperebenen
  - im 2D: Gerade mit orthogonalem Vektor
  - im 3D: Ebene mit Normale
  - vorne vs. hinten
- Idee:
  - rekursive Unterteilung des Raumes
  - solange, bis in jeder Region nur noch ein Teilobjekt enthalten ist
  - alternativ: ausreichend wenige Objekte



BSP-Baum (Quelle: [5])

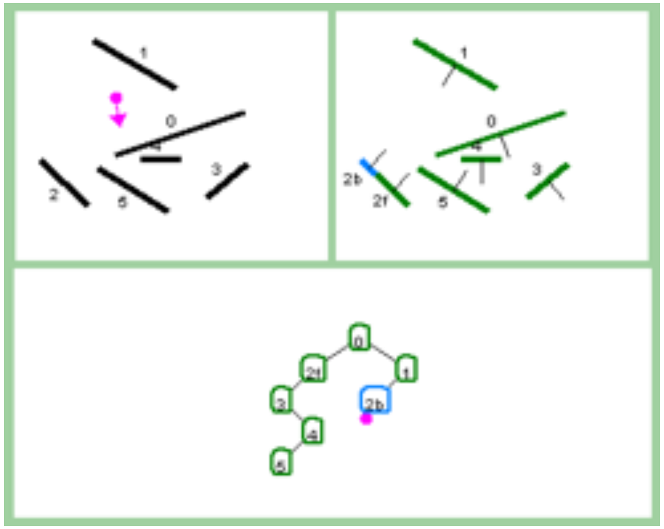
Binary Space Partion (BSP)

- Konstruktion
- beginne mit beliebiger Hyperebene h
  - +: sortiere alle Hyperebenen "vor" h in linken Teilbaum
  - -: sortiere alle Hyperebenen "hinter" h in rechten Teilbaum
  - unterteile Hyperebenen, falls notwenig
- sortiere linken und rechten Teilbaum



Binary Space Partition (BSP)

- Demo: *http://www.symbolcraft.com/products/bsptrees/*



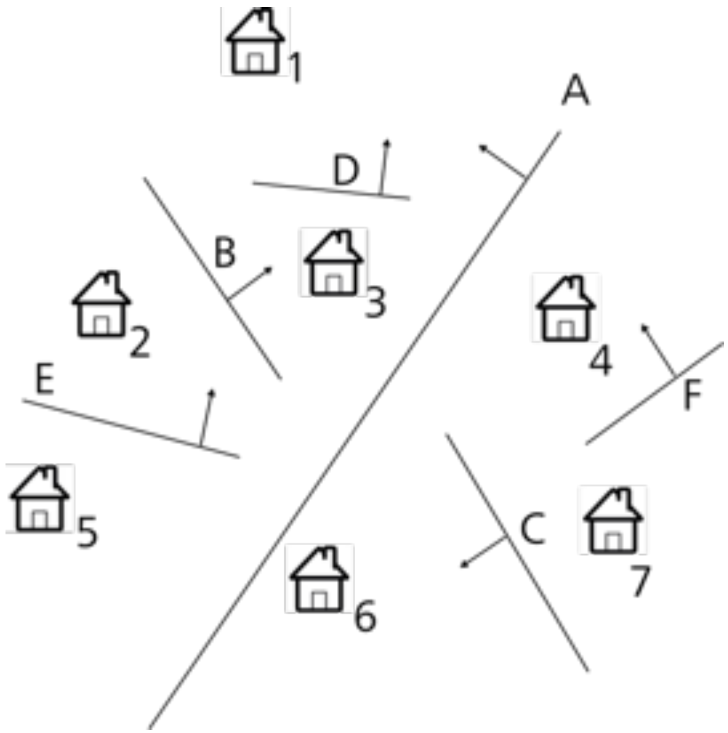
Interaktive Demo für BSP-Bäume (Quelle:[4])

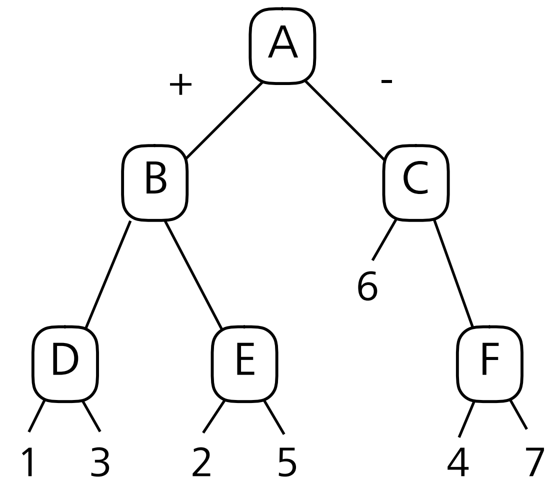
BSP-Baum erstellen

- Wie könnte ein BSP-Baum für die Szene bestehend aus den Objekten 1…7 aussehen?

BSP-Baum

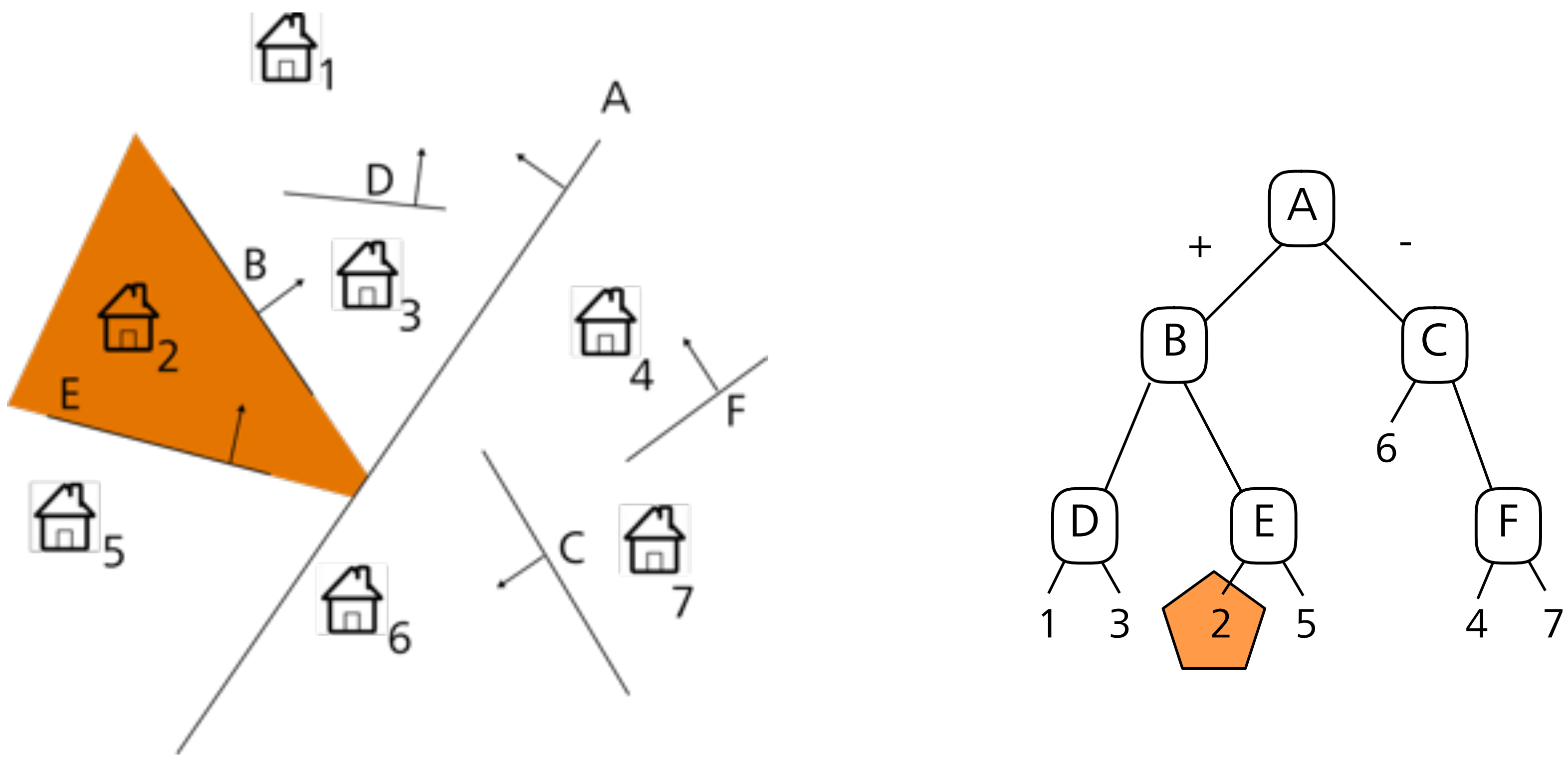
- eine mögliche Unterteilung der Szene





Zellen im BSP-Baum

- Unterteilung der Szene durch BSP-Ebene ergibt Zellen

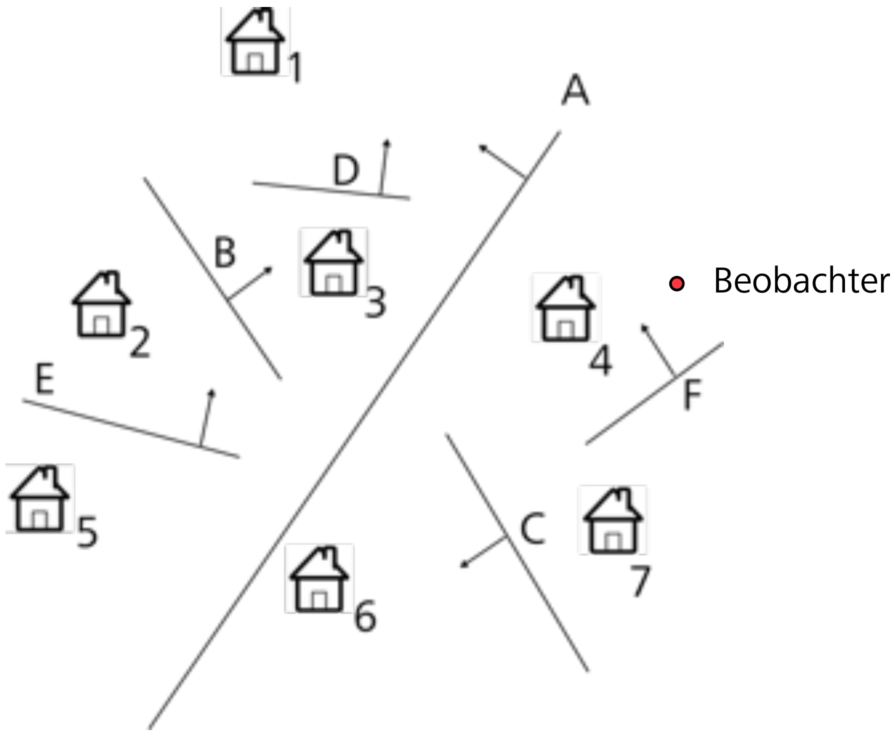


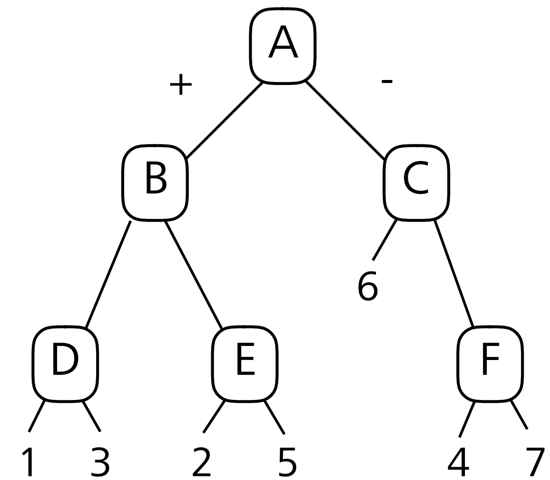
Binary Space Partion (BSP)

- Back-To-Front Sortierung
- kann vom BSP-Baum abgelesen werden
- Algorithmus
  - starte am Wurzelknoten
  - zeichne
    - falls Position "hinter" der Hyperebene: alle Knoten des Teilbaums "vor" der Hyperebene (rekursiver Durchlauf)
    - ansonsten: alle Knoten des Teilbaums "hinter" der Hyperebene (rekursiver Durchlauf)
  - zeichne Knoten selber
  - zeichne übrigen Teilbaum

Übung: Back-To-Front Sortierung

- Leiten Sie aus dem BSP-Baum eine Back-To-Front-Traversierung für die Beobachterposition her.

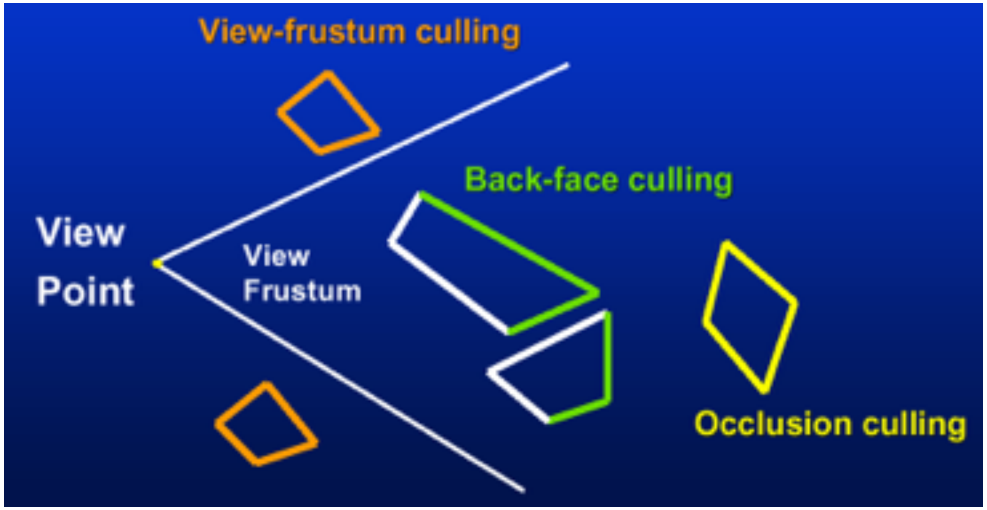




Sichtbarkeitsberechnung

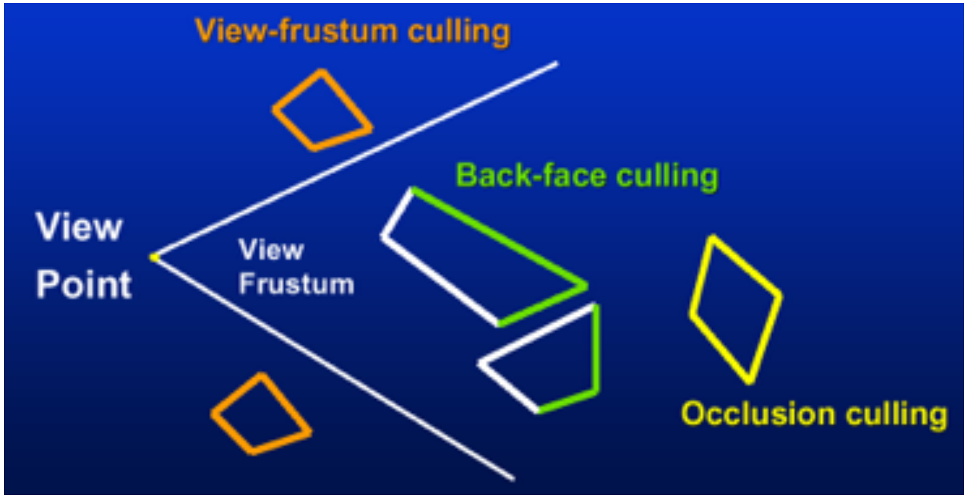
Culling

- Objekte, die vom Beobachter nicht gesehen werden, müssen nicht dargestellt werden
  - View-Frustum Culling
  - Back-Face Culling
  - Occlusion Culling



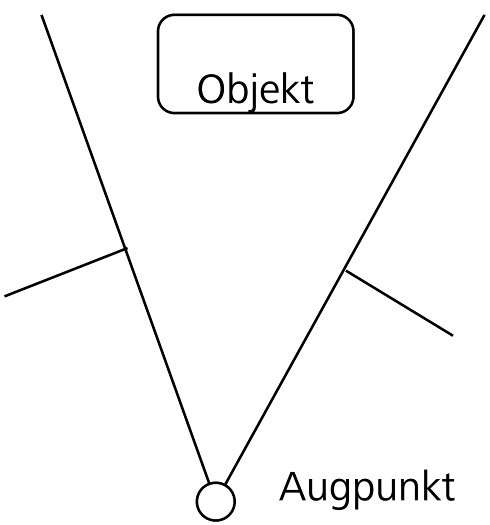
View Frustum Culling

- Ignorieren aller Objekte, die nicht im Sichtfeld liefen
- Schnitt mit Sichtbarkeitsvolumen
    - Sichtbarkeitspyramide

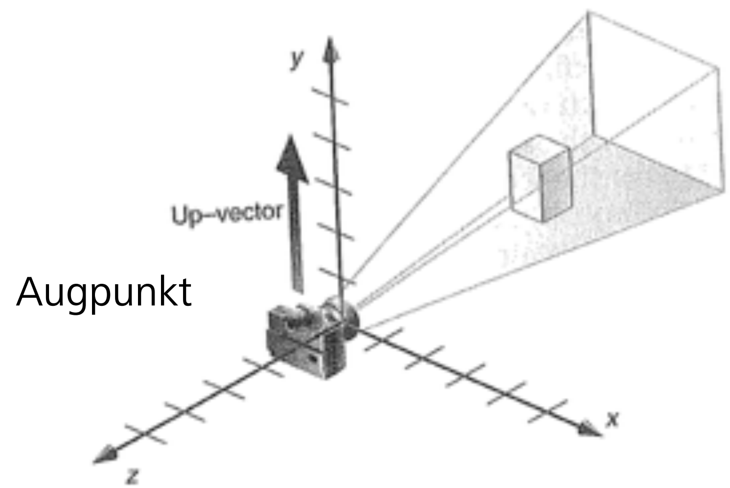


View Frustum Culling

- Berechnung
- Sichbarkeitskegel durch Ebenen einschränken
    - 2D: 2 Geraden
    - 3D: 4 Ebenen



Sichtbarkeitsvolumen im 2D:
2 Begrenzungsgeraden



Sichtbarkeitsvolumen im 3D:
4 Begrenzungsebenen

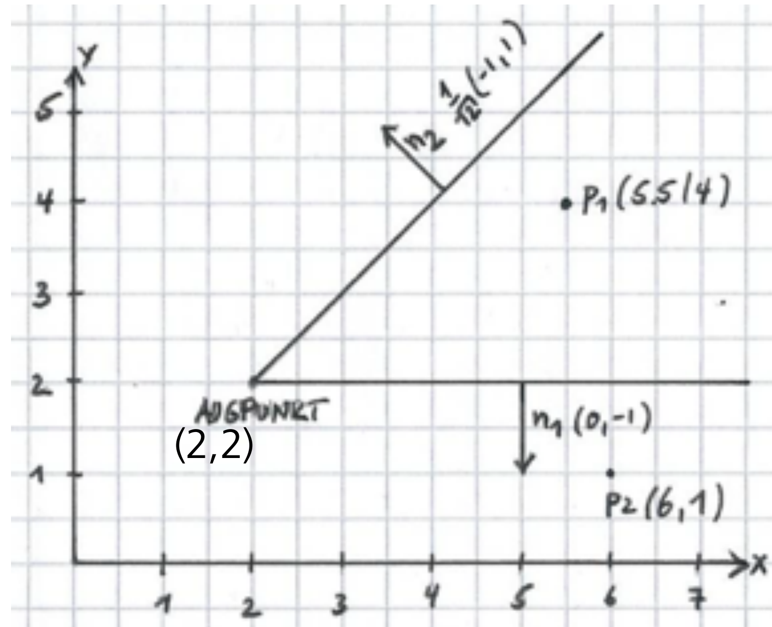
View Frustum Culling

- Beispiel: Berechnung im 2D
- Darstellung der der Begrenzungs- Geraden in Hesse-Normal-Form (HNF)

$$E_1 : (\vec{x} \cdot \begin{pmatrix} 0 \\ -1 \end{pmatrix} - \begin{pmatrix} 2 \\ 2 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ -1 \end{pmatrix}) = 0$$

$$E_1 : \vec{x} \cdot \begin{pmatrix} 0 \\ -1 \end{pmatrix} + 2 = 0$$

$$E_2 : \frac{1}{\sqrt{2}} \vec{x} \cdot \begin{pmatrix} -1 \\ 1 \end{pmatrix} = 0$$



beide Abstände negativ: innen

nicht beide Abstände negativ: außen

$$E_1(p_1) : -4 + 2 = \boxed{-2}$$

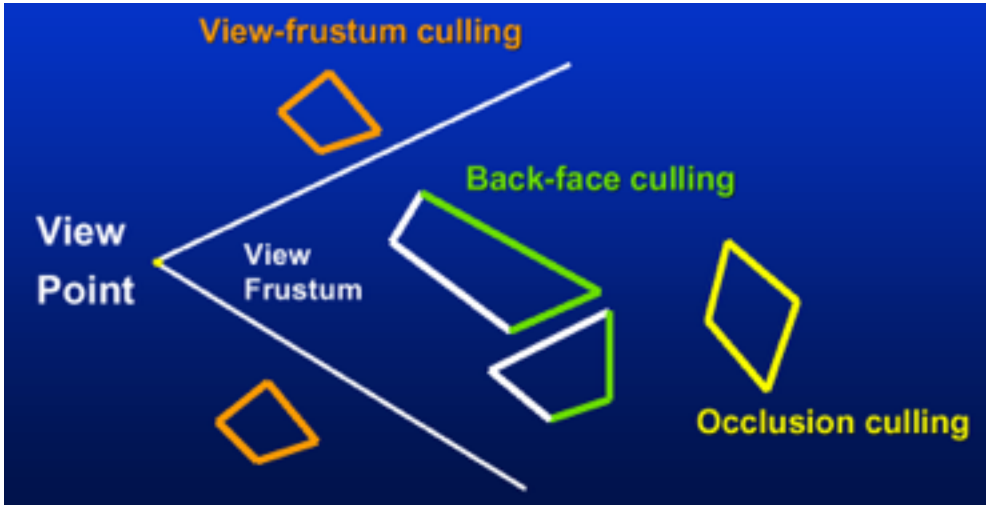
$$E_2(p_1) : \frac{-5.5 + 4}{\sqrt{(2)}} = \frac{-1.5}{\sqrt{(2)}} \approx \boxed{-1.06}$$

$$E_1(p_2) : -1 + 2 = \boxed{1}$$

$$E_2(p_2) : \frac{-6 + 1}{\sqrt{(2)}} = \frac{-5}{\sqrt{(2)}} \approx \boxed{-3.54}$$

Backface Culling

- Ignorieren aller Flächen, die vom Beobachter weg zeigen



Backface Culling

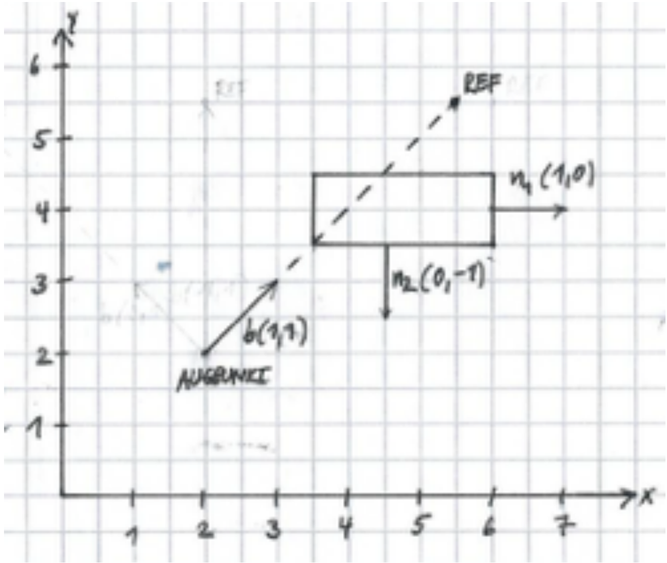
- Berechnung
- Entscheidung anhand der Oberflächennormalen
    - Skalarprodukt zwischen Beobachterrichtung und Flächennormale

nicht sichtbar!

$$b \cdot n_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} = 1 > 0$$

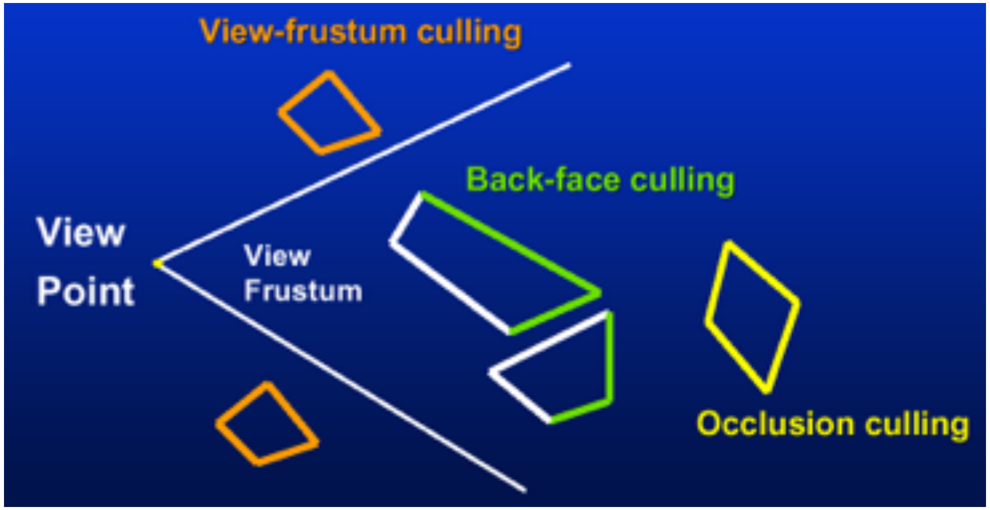
sichtbar!

$$b \cdot n_2 = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ -1 \end{pmatrix} = -1 < 0$$



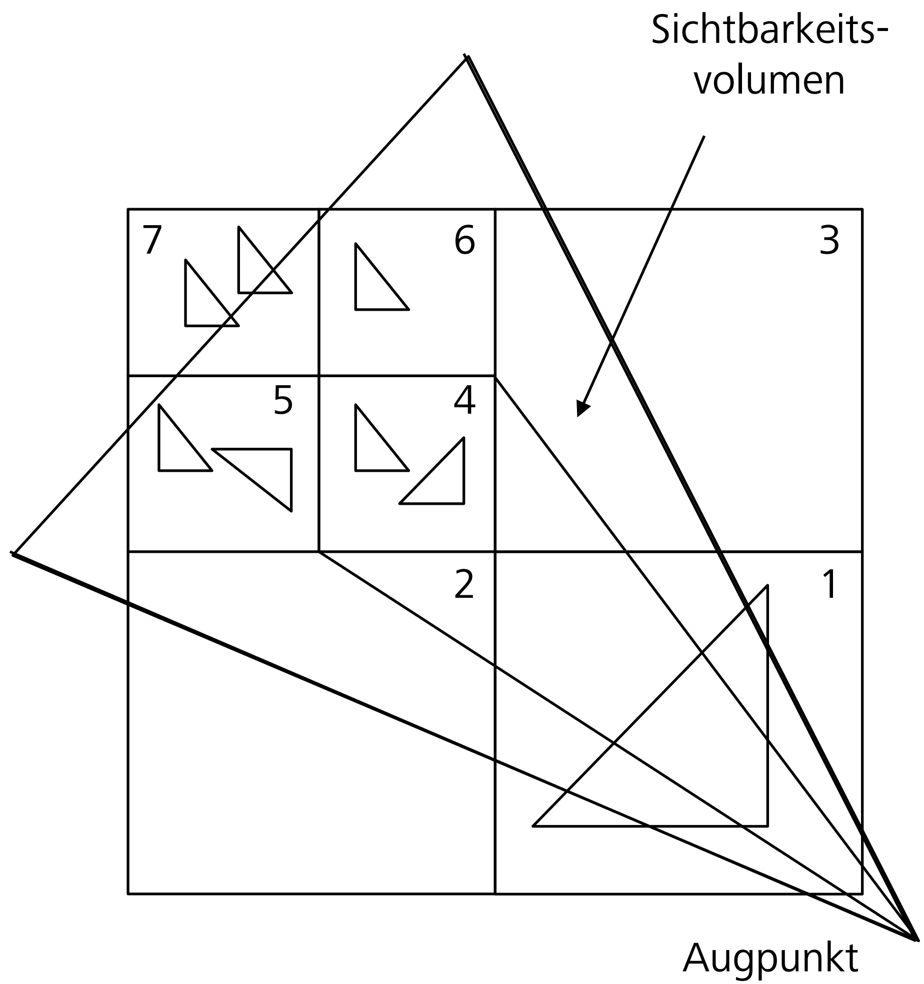
Occlusion Culling

- Ignorieren aller Objekte, die von näher liegenden Objekte verdeckt werden



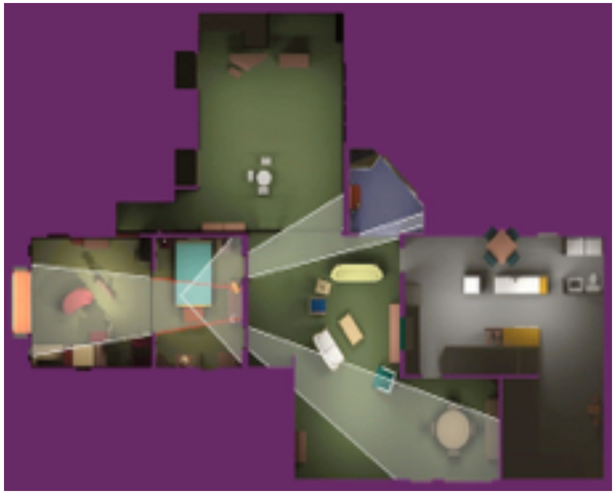
Occlusion Culling

- Performante Auswertung über hierarchische Datenstrukturen
- Berechnungsidee:
  - Zelle 4 ist durch das Objekt in Zelle 1 vollkommen verdeckt: graue, gestrichelte Linien
  - Objekte in Zelle 4 müssen nicht gezeichnet werden

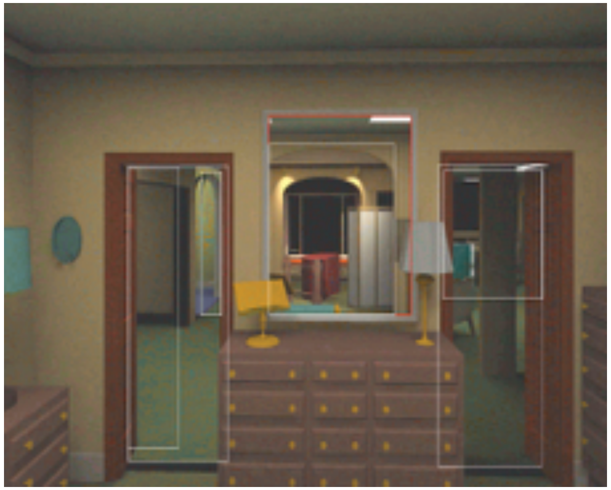


Portal Culling

- Motivation: Szene meist unterteilt in geschlossene Räume
  - Wenige Übergange zwischen Räumen (z.B. Türen, Fenster) = Portale
  - Idee: Portale schränken Sichtfeld ein



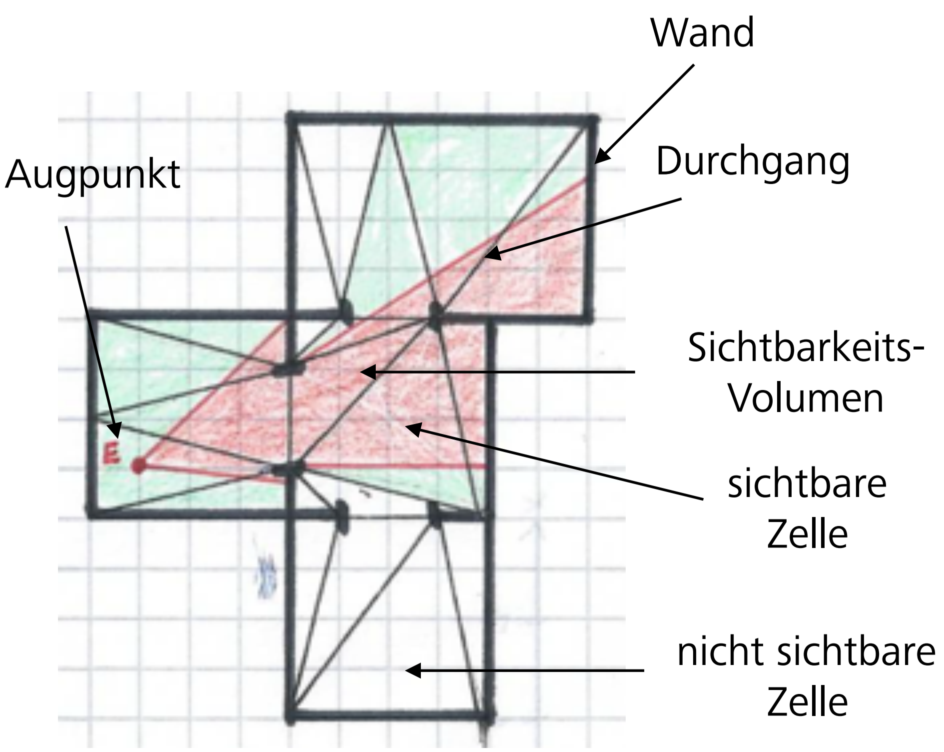
Grundriss einer Szene mit Portalen wie Türen oder Spiegel (Quelle: [2]).



Gerendertes Bild (Quelle: [2]).

Übung: Portal Culling

- Gegeben ist eine 2D-Szene, die aus Zellen besteht.
- Jede Parzelle ist ein Dreieck.
- Jede Kante eines Dreiecks kann entweder eine Wand oder ein Durchgang sein.
- Erstellen Sie einen Algorithmus (Pseudocode) zum Finden aller sichtbaren Zellen (genannt Potentially Visible Set oder PVS)



Zusammenfassung

- Hüllkörper
- Hüllkörper-Hierarchien
  - Octrees (Quadtrees)
  - BSP-Bäume
- Sichtbarkeitsberechnung

Quellen

- Die Folien basieren Teilweise auf Vorlesungsfolien von Prof. Dr. Stefan Gumhold (Technische Universität Dresden) und Prof. Dr. Wolfgang Straßer (Universität Tübingen, emeritiert)
- Als Basis dient außerdem folgendes Buch: Thomas Akenine-Möller, Eric Haines, Naty Hoffman: Real-Time Rendering, CRC Press, 2008

- [1] Wikipedia Hüllkörper: http://de.wikipedia.org/wiki/H%C3%BCllk%C3%B6rper, abgerufen am 7.12.2013
- [2] Octree (hier im Kontext GPU-Programmierung): http://www.aracknea-core.com/sylefeb/octreetex/, abgerufen am 7.12.2013
- [3] Wikipedia Octree: http://de.wikipedia.org/wiki/Octree, abgerufen am 7.12.2013
- [4] Interaktive Demo zu BSP-Bäumen: http://www.symbolcraft.com/products/bsptrees/, abgerufen am 7.12.2013
- [5] Wikipedia Binary Space Partitioning, http://de.wikipedia.org/wiki/Binary_Space_Partitioning, abgerufen am 7.12.2013
- [6] Wikipedia: Z-Buffering, http://en.wikipedia.org/wiki/Z-buffering, abgerufen am 1.1.2014
- [7] http://users.csc.calpoly.edu/~zwood/teaching/csc572/final12/iseletsk/, abgerufen am 1.1.2014