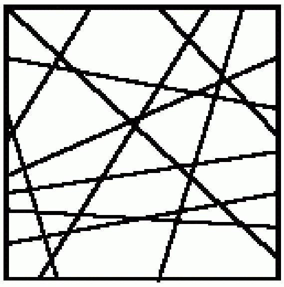
9. Arrangements von Geraden

(Kollektionen von Geraden, bzw. in höheren Dimensionen:
von Ebenen, von Hyperebenen)

Arrangements von Geraden im 2-d (bzw. Ebenen im 3-d) bilden nach konvexer Hülle und dem Voronoi-Diagramm / Delaunay-Triangulierung die dritte wichtige Struktur in der algorithmischen Geometrie:



(Hinrichs 2001)

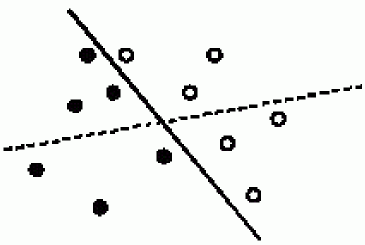
Beachte: Die Abb. zeigt das Arrangement nur unvollständig!

Anwendungen:

- Sichtbarkeitsgraphen in der Ebene (2 Punkte *a* und *b* verbunden, wenn *a* von *b* sichtbar)
- Bestimmung des größten leeren konvexen Polygons mit Ecken aus gegebener Menge *S*

    ungelöstes Problem: Enthält jede genügend große Menge von Punkten in der Ebene ein leeres Sechseck? (Erdös; Horton 1983)

- *Hidden Surface Removal*:
  Gegeben sei eine endliche Menge von Polyedern im $\mathbb{R}^3$.
  Gesucht sind die von einem Punkt im $\mathbb{R}^3$ aus sichtbaren (Teile von) Seitenflächen der Polyeder.
- *Ham Sandwich Cut*:
  Gegeben seinen zwei Punktmengen A und B im $\mathbb{R}^2$, die durch eine Gerade voneinander trennbar sind.
  Gesucht ist eine Gerade, die $A \cup B$ so in zwei Punktmengen zerlegt, daß A und B jeweils gleichmäßig aufgeteilt werden.



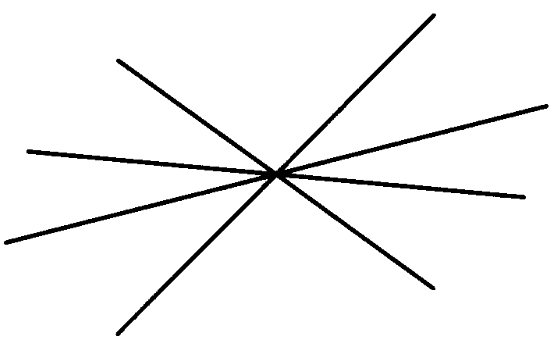
Herkunft der Bezeichnung "Ham Sandwich Cut":
3D-Analogon; jedes Schinken- und Käse-Sandwich kann so durch eine Ebene in 2 Teile zerlegt werden, dass beide Teile genau dieselbe Menge an Brot, Schinken und Käse abbekommen.

Definition:

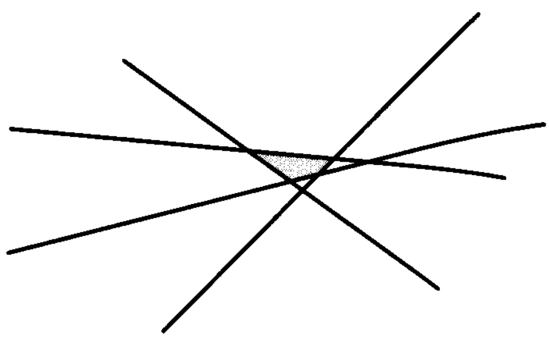
Sei $\mathcal{G}$ eine (endliche) Menge von Geraden in der Ebene. Dann induziert $\mathcal{G}$ eine Partition der Ebene in offene konvexe Regionen (Zellen), Schnittpunkte von Geraden und offene Geradensegmente, die auf den Geraden zwischen Schnittpunkten liegen. Diese Partition wird als das durch $\mathcal{G}$ induzierte *Arrangement* $\mathcal{A}_\mathcal{G}$ bezeichnet.
Ein Arrangement $\mathcal{A}_\mathcal{G}$ heißt *einfach*, falls jeder Schnittpunkt durch genau zwei Geraden bestimmt wird und keine zwei es induzierenden Geraden parallel sind.

- *Kombinatorische Komplexität* eines Arrangements $\mathcal{A}_\mathcal{G}$
  = Anzahl Knoten + Anzahl Kanten und + Anzahl Flächen
- Durch Beschränkung auf *einfache Arrangements*
  wird kombinatorische Komplexität nicht verändert!

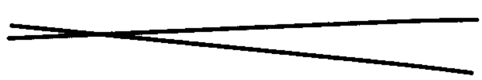
"Einfach-machen" eines nicht-einfachen Arrangements durch kleine Perturbationen: (a) mehr als 3 Geraden durch einen Punkt; (b) parallele Geraden



(a)



(b)



Satz:

Sei $\mathcal{G}$ eine Menge von n Geraden in der Ebene. $\mathcal{A}_{\mathcal{G}}$ bezeichne das durch $\mathcal{G}$ induzierte Arrangement. Dann gilt für die Anzahl V der Knoten, die Anzahl E der Kanten und die Anzahl F der Flächen:

$$V \le \binom{n}{2} \qquad E \le n^2 \qquad F \le \binom{n}{2} + n + 1$$

Ist das Arrangement einfach, so gilt:

$$V = \binom{n}{2} \qquad E = n^2 \qquad F = \binom{n}{2} + n + 1$$

Beweis:

Die Knoten sind genau die Schnittpunkte der Geraden aus $\mathcal{G}$ $\Rightarrow$
$V \le \binom{n}{2}$ für ein einfaches Arrangement.
Diese Anzahl wird nur erreicht, wenn jedes Paar von Geraden einen eindeutigen Schnittpunkt, d.h. Knoten, generiert.

Anzahl Kanten auf einer Geraden
= 1 + Anzahl Schnittpunkte auf der Geraden
$\le 1 + n - 1 = n$
$\Rightarrow$ Gesamtanzahl Kanten $\le n^2$
Diese Anzahl wird nur bei einem einfachen Arrangement erreicht.

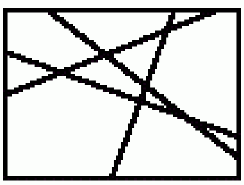
Schranke für die Anzahl F der Flächen:
Füge die Geraden nacheinander in das Arrangement ein und beschränke die Anzahl der in jedem Schritt neu entstehenden Flächen.
Sei $\mathcal{G} = \{g_1, g_2, \ldots, g_n\}$ und $\mathcal{G}_i = \{g_1, g_2, \ldots, g_i\}$ und $\mathcal{A}_i$ das durch $\mathcal{G}_i$ induzierte Arrangement. Wie wächst die Anzahl F der Flächen beim Übergang von $\mathcal{G}_{i-1}$ nach $\mathcal{G}_i$?
Jede Kante von g_i teilt eine Fläche von $\mathcal{G}_{i-1}$ in zwei Flächen auf $\Rightarrow$ F wächst um die Anzahl Kanten von $\mathcal{A}_i$ auf g_i, die beschränkt ist durch i $\Rightarrow$ Gesamtanzahl F der Flächen ist beschränkt durch

$$F \leq 1 + \sum_{i=1}^{n} i = \binom{n}{2} + n + 1$$

Diese Anzahl wird nur bei einem einfachen Arrangement erreicht. ■

Aus dem Satz folgt:
Ein von *n* Geraden induziertes Arrangement hat quadratische Komplexität (*V*, *E* und *F* sind alle $O(n^2)$).

- Speicherung eines Arrangements $\mathcal{A}_{\mathcal{G}}$ durch eine DCEL:
  DCEL kann nur beschränkte Kanten (d.h. keine Halbgeraden) speichern → Umgebe das Arrangement mit einem achsenparallelen Container $C_{\mathcal{G}}$, der alle Schnittpunkte von Geraden aus $\mathcal{G}$ enthält:



$\Rightarrow$

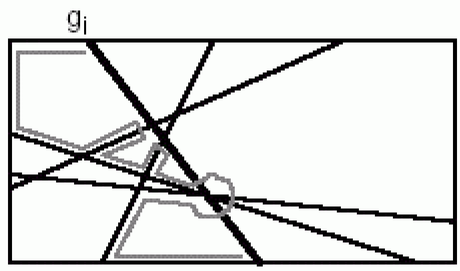
Planare Unterteilung, die durch den Container und den im Container liegenden Teil des Arrangements definiert wird, hat nur beschränkte Kanten und kann daher in einer DCEL gespeichert werden.

Bestimmung des Containers in $\Theta(n^2)$ Zeit:

Berechne alle Schnittpunkte von Geraden und bestimme die Schnittpunkte mit kleinster x- bzw. y-Koordinate und die mit größter x- bzw. y-Koordinate.

- Konstruktion der DCEL für eine vorgegebene Menge von Geraden $\mathcal{G}$?
- 1. Idee:
  Anpassung des Plane-Sweep für Liniensegmentschnitt liefert einen Algorithmus mit $\Theta(n^2 \cdot \log n)$ Zeitaufwand, da $k \in \Theta(n^2)$.

- Besser: Inkrementelle Konstruktion
  Füge die Geraden $g_1, g_2, \ldots, g_n$ von $\mathcal{G}$ nacheinander ein und aktualisiere jeweils die DCEL nach jeder Einfügeoperation.
  $\mathcal{A}_i$ bezeichne das durch $\mathcal{G}_i = \{g_1, g_2, \ldots, g_i\}$ induzierte Arrangement innerhalb des Containers $C_{\mathcal{G}}$.
  Um die Gerade g_i hinzuzufügen, teile die Flächen in $\mathcal{A}_{i-1}$ auf, die durch g_i geschnitten werden:



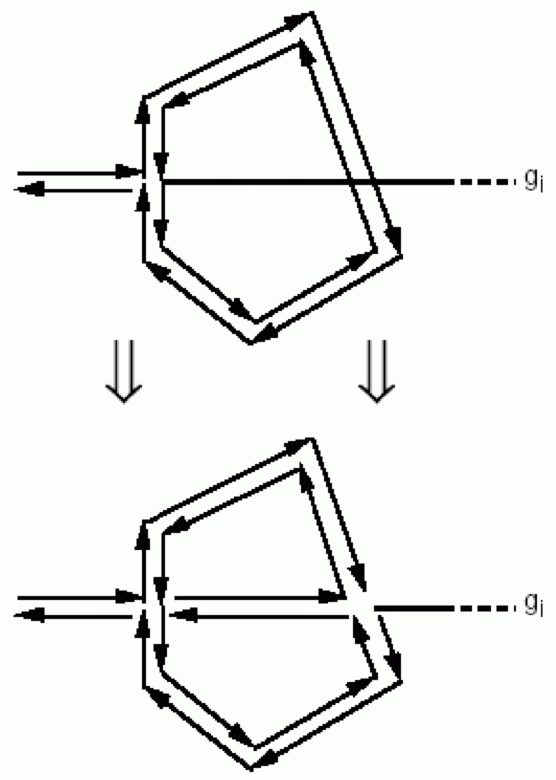
Auffinden der von g_i geschnittenen Flächen durch links-nach-rechts Wanderung entlang g_i:

Betreten einer Fläche f durch eine Kante e: Wandere entlang dem Rand von f (z.B. entgegen dem Uhrzeigersinn), bis die nächste Kante e' oder ein Eckpunkt v gefunden wird, wo g_i die Fläche f verläßt.

- Kante e': Trete über den Twin()-Zeiger in die Nachbarfläche ein und verfahre ebenso.
- Eckpunkt v: Wandere entgegen dem Uhrzeigersinn um v herum, bis die nächste von g_i geschnittene Fläche angetroffen wird.

Zeitaufwand ist proportional zur Komplexität von f bzw. zum Grad von v.

- Der am weitesten links liegende Schnittpunkt von g_i und $\mathcal{A}_{i-1}$ wird gefunden durch Testen von g_i gegen alle Kanten von $C_{\mathcal{G}}$.
- Die zu diesem Schnittpunkt benachbarte Fläche ist die erste, die durch g_i aufgeteilt wird. (Sonderfälle: g_i geht durch Eckpunkt von $C_{\mathcal{G}}$ oder g_i vertikal.)
- Aufteilen einer Fläche f:



ConstructArrangement($\mathcal{G}$)

Eingabe: Eine Menge $\mathcal{G} = \{g_1, g_2, \ldots, g_n\}$ von n Geraden im $\mathbb{R}^2$.

Ausgabe: Der durch eine DCEL repräsentierte, innerhalb eines Containers $C_\mathcal{G}$ liegende Teil des durch $\mathcal{G}$ induzierten Arrangements $\mathcal{A}_\mathcal{G}$.

1. Bestimme einen Container $C_\mathcal{G}$, der alle Schnittpunkte von Geraden aus $\mathcal{G}$ enthält.
2. Konstruiere eine DCEL für die durch $C_\mathcal{G}$ induzierte planare Partition.
3. for i = 1 to n do
4.   Finde die Kante e auf dem Rand von $C_\mathcal{G}$, die den am weitesten links liegenden Schnittpunkt von g_i und $\mathcal{A}_{i-1}$ enthält.
5.   f = zu e inzidente Fläche, die innerhalb von $C_\mathcal{G}$ liegt.
6.   while f ist nicht unbeschränkte Fläche außerhalb $C_\mathcal{G}$ do
7.   Teile f auf.
8.   f = nächste von g_i geschnittene Fläche.

Zeitaufwand für

- Schritt 1: $\Theta(n^2)$
- Schritt 2: $\Theta(1)$
- Schritt 4: $\Theta(n)$
- Aufteilen der von g_i geschnittenen Flächen:

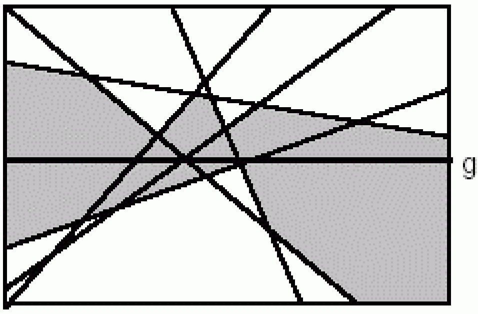
  $\mathcal{A}_\mathcal{G}$ einfach $\Rightarrow$ Zeitaufwand zum Aufteilen einer Fläche f ist linear in der Komplexität von f $\Rightarrow$ Gesamtaufwand ist linear in der Summe der Komplexitäten der von g_i geschnittenen Flächen.

$\mathcal{A}_\mathcal{G}$ beliebig $\Rightarrow$ Verlassen einer Fläche durch einen Eckpunkt v: Die beim Umlauf um v angetroffenen Kanten gehören zum Rand von Flächen, deren Abschluß von g_i geschnitten wird.

Definition:

Die *Zone* einer Geraden g in einem Arrangement $\mathcal{A}_{\mathcal{G}}$, das durch eine Menge $\mathcal{G}$ von Geraden induziert wird, ist die Menge der Flächen von $\mathcal{A}_{\mathcal{G}}$, deren Abschluß durch g geschnitten wird. Die *Komplexität einer Zone* ist die Summe der Komplexitäten ihrer Flächen, d.h. die Summe der Anzahl Kanten und der Anzahl Eckpunkte dieser Flächen.

- Beispiel:



Zone von g hat Komplexität
$2 \cdot 5 + 2 \cdot 4 + 2 \cdot 5 + 2 \cdot 3 + 2 \cdot 4 + 2 \cdot 4 + 2 \cdot 6 = 62$

- Zeitaufwand zum Aufteilen der von g_i geschnittenen Flächen in $\mathcal{A}_{i-1}$ ist linear in der Komplexität der Zone von g_i in $\mathcal{A}_{i-1}$.

Es gilt der *Zonensatz* (Chazelle, Guibas & Lee 1985):

Die Komplexität der Zone einer Geraden in einem Arrangement von *m* Geraden ist *O*(*m*).

Beweis:

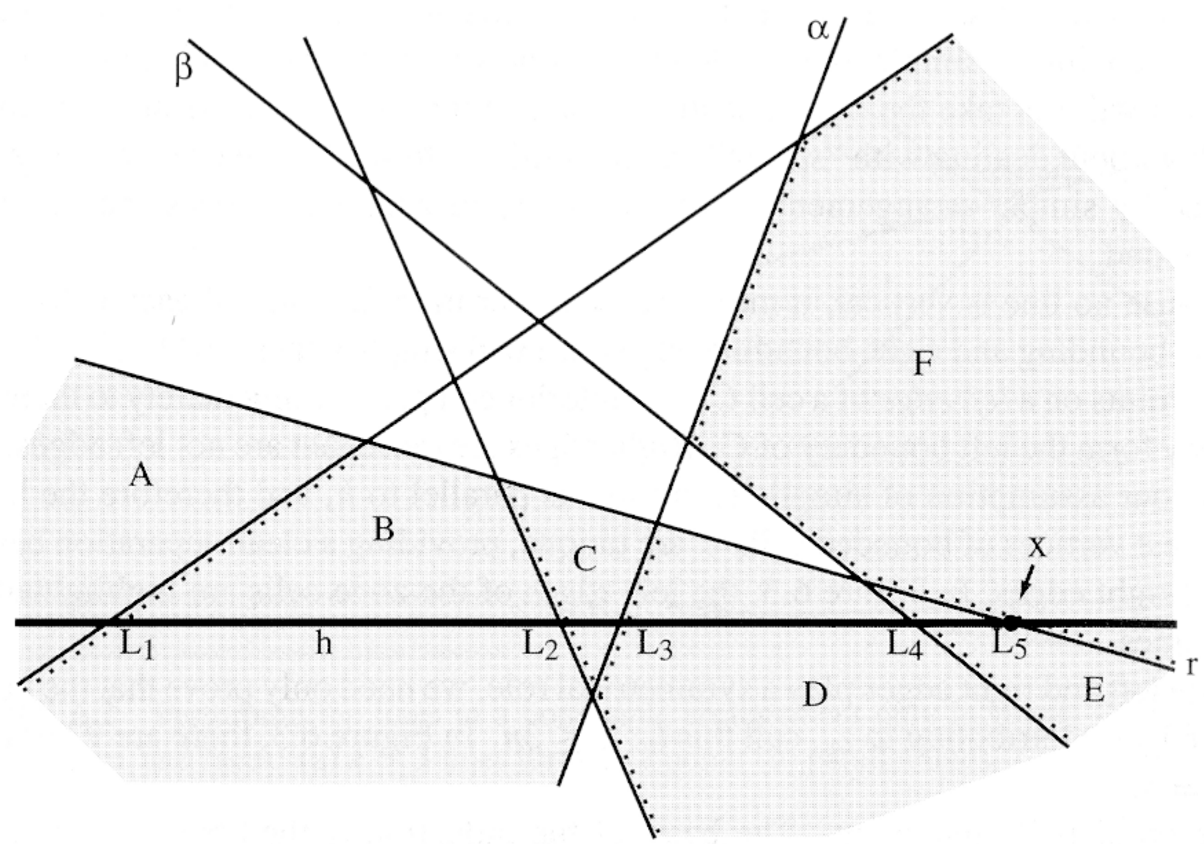
Sei $\mathcal{G} = \{g_1, g_2, \ldots, g_m\}$ Menge von m Geraden und g eine weitere Gerade.

oBdA: g fällt mit der x-Achse zusammen.

Kein g_i sei parallel zu g.

Jede Kante in $\mathcal{A}_{\mathcal{G}}$ begrenzt zwei Flächen. Eine Kante heißt *linke Kante* für die zu ihrer Rechten liegende Fläche und *rechte Kante* für die zu ihrer Linken liegende Fläche.

in der folgenden Abb. sind die linken Kanten mit Punkten markiert:



Sei L die Anzahl der linken Kanten, die zur Zone von g gehören.

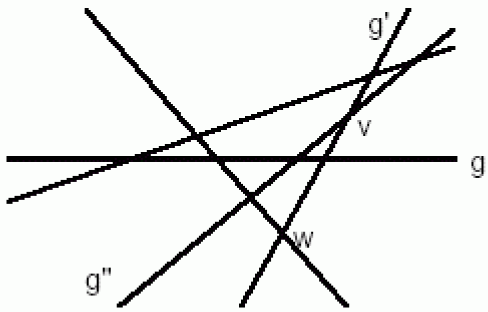
zu zeigen: $L \leq 5 \cdot m$

Beweis durch vollständige Induktion:

m = 1: trivial

$m - 1 \rightarrow m$:

m > 1:
Sei g' die Gerade aus $\mathcal{G}$, die g am weitesten rechts schneidet.



1. Fall:
g' eindeutig definiert.

Induktionsannahme $\Rightarrow$
Die Zone von g in $\mathcal{A}_{\mathcal{G} \setminus \{g'\}}$ hat höchstens 5·(m–1) linke Kanten.

Hinzufügen von g' $\Rightarrow$ Erhöhung von L durch neue linke Kanten auf g' und Aufteilen alter linker Kanten durch g':
Sei v der erste Schnittpunkt von g' mit einer anderen Geraden oberhalb von g und w der erste Schnittpunkt unterhalb von g.
$\overline{vw}$ ist eine neue linke Kante von g', weiterhin teilt g' vorhandene linke Kanten in v und w $\Rightarrow$ L erhöht sich um 3.
Existieren v oder w nicht $\Rightarrow$ geringere Erhöhung von L.

Die Teile von g', die oberhalb von v bzw. unterhalb von w liegen, können keine zur Zone von g gehörigen linken Kanten enthalten.

$\Rightarrow (5\cdot(m-1) + 3 \leq 5\cdot m$

2. Fall: Es gibt mehrere Geraden aus $\mathcal{G}$, die *g* im weitesten rechts liegenden Schnittpunkt schneiden. Dieser Fall ist komplizierter.
g' sei eine dieser Geraden. Durch Argumentation analog zum 1. Fall kann man zeigen (Hinrichs 2001): Anzahl der linken Kanten erhöht sich höchstens um 5.
Somit: $L \leq 5(m-1) + 5 = 5m$.

hier noch nicht behandelt: Sonderfälle – Geraden parallel zu *g*.
Siehe de Berg & van Kreveld (2000).

Es folgt:
Die DCEL eines Arrangements von *n* Geraden kann in der Zeit $O(n^2)$ inkrementell berechnet werden.

Verallgemeinerung auf höhere Dimensionen
(ohne Beweis; s. O'Rourke 1998, S. 201):

Die Anzahl der Facetten beliebiger Dimension in einem Arrangement von *n* Hyperebenen im *d*-dimensionalen Raum ist $O(n^d)$, die Zone jeder Hyperebene hat dort die Komplexität $O(n^{d-1})$, und ein solches Arrangement kann mit Zeit- und Speicheraufwand $O(n^d)$ konstruiert werden.

Anwendungen:

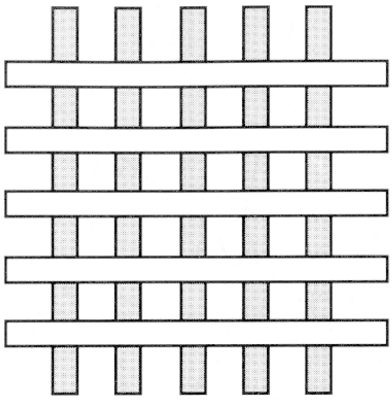
Hidden Surface Removal

- Basis für 3D-Computergrafik
- gegenwärtig am häufigsten angewandte geometr. Berechnung

gegeben: Menge flacher, undurchsichtiger, unterschiedlich gefärbter Polygone im R^3; Beobachterstandpunkt (virtuelle Kamera)
gesucht: 2D-Bild ihres Aussehens vom gegebenen Standpunkt

n = Gesamtzahl der Kanten aller Polygone

worst case-Komplexität des Output-Bildes: $O(n^2)$



- viele Algorithmen erreichten $O(n^2 \log n)$
- optimaler Algorithmus wurde lange Zeit nicht gefunden
- McKenna 1987: $O(n^2)$-Algorithmus auf der Grundlage von Arrangements

Annahmen:
- Polygone schneiden sich nicht in ihrem Inneren
- Beobachterstandpunkt unendlich weit entfernt $(0; 0; +\infty)$: Parallelprojektion (andernfalls wird Szene transformiert; s. Computergrafik, **`http://www-gs.informatik.tu-cottbus.de/~wwwgs/cg2_v07a.pdf`**)
- es gibt ein "Hintergrundpolygon", das hinter allen anderen liegt und den gesamten Szenen-Hintergrund füllt

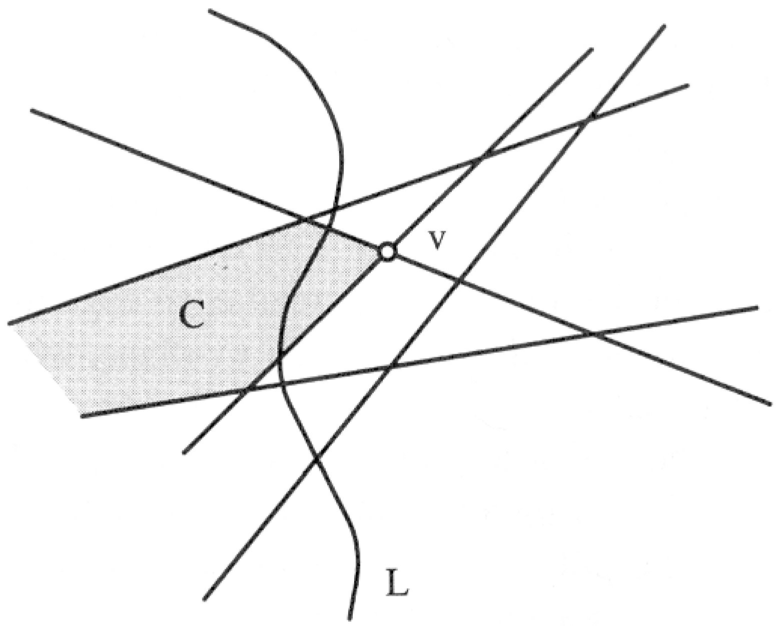
Skizze des McKenna-Algorithmus:

- Projiziere alle Kanten der Input-Polygone orthografisch in die *xy*-Ebene
- erweitere jede Kante zu Gerade $\Rightarrow$ Arrangement von n Geraden
- Konstruiere dieses Arrangement als Datenstruktur (Aufwand $O(n^2)$)
- jetzt liegt jede Zelle des Arrangements in der Projektion von mindestens einem Polygon
- gesucht: dasjenige dieser Polygone, das jeweils dem Betrachter am nächsten liegt (d.h. die größte z-Koordinate hat) (dieses ist eind. bestimmt)

$O(n)$ Polygone zu jeder Zelle $\Rightarrow$ naiver Ansatz: für jede der $O(n^2)$ Zellen diese alle überprüfen; Gesamtzeit $O(n^3)$.

- Grundidee für Verbesserung: "*topologischer Sweep*" mit "Pseudo-Gerade" L

L ist eine einfache, offene, orientierte Kurve, die jede Gerade des Arrangements genau einmal schneidet, und zwar für alle Geraden in einheitlicher Richtung (von unten nach oben).



(aus O'Rourke 1998)

Vorteil der "Biegsamkeit" von L:
Sortierung, welcher Schnittpunkt v als nächster erreicht wird, entfällt – benutze ungeordnete Menge von "sweepable" Knoten

v "sweepable" ⇔ *v* inzidiert mit 2 Kanten, die von *L* in benachbarten Punkten geschnitten werden.
Es kommt nicht darauf an, welcher der "sweepablen" Punkte von *L* als nächstes überschritten wird.

Sweep-Status-Struktur enthält:

- Liste der n+1 *aktiven Zellen* (geschnitten von *L*)
- Liste der *aktiven Kanten* (dito)
- für jede aktive Zelle *C* eine Liste der Polygone, deren Projektion *C* enthält, sortiert nach *z*-Koordinate
- diese Liste ändert sich bei Überstreichen eines Knotens nur wenig (Ausnutzung von Szenen-Kohärenz)
- ⇒ konstante Update-Zeit
- ⇒ insgesamt worst case-Zeitaufwand $O(n^2)$

McKenna-Algorithmus ist in der Praxis nicht optimal:

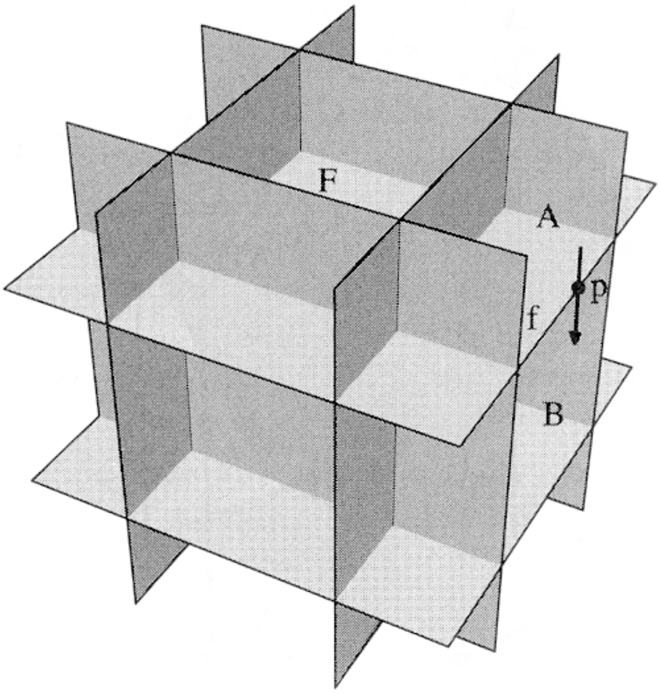
- benötigt *immer* $\Omega(n^2)$ Zeit und Speicherplatz
- die meisten "realistischen" Szenen haben viel geringere Komplexität
- gesucht: *output-sensitive* Hidden Surface-Algorithmen (aktives Forschungsgebiet)

Visual Space Partition und Aspekt-Graph

gegeben: Polyeder im R^3
Aufgabe: Erkennen, ob sich dieses Polyeder in einer gegebenen Szene wiederfindet (Mustererkennung)
zur Unterstützung der Lösung: alle Ansichten, die ein Betrachter von dem Polyeder haben kann, speichern
– Äquivalenz von Ansichten: nur kombinatorisch verschiedene Ansichten werden gespeichert ("Aspekte" des Polyeders)

VSP (visual space partition) eines Polyeders *P*: Partition des Äußeren von *P* in Zellen mit konstantem Aspekt von *P*.

Beispiel "Würfel":



VSP des Würfels

der Beobachter p kann F sehen, wenn p in Zelle A ist, aber nicht von Zelle B aus

der *Aspekt-Graph* von P: zur VSP dualer Graph.

Es gilt:
P beschränktes, konvexes Polyeder $\Rightarrow$ die VSP von P ist genau das Arrangement aus den Ebenen, die die Seitenflächen von P enthalten (Plantinga & Dyer 1990).

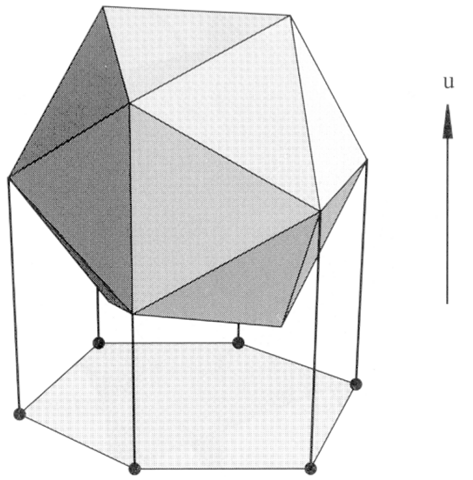
$\Rightarrow$ die VSP eines beschränkten, konvexen Polyeders mit n Ecken hat Größe $O(n^3)$ und kann in Zeit $O(n^3)$ konstruiert werden.

Beachte: für nichtkonvexe Polyeder steigt die Komplexität des Aspekt-Graphen auf $O(n^9)$.

(s. O'Rourke 1998, S. 212)

verwandtes Problem:

Minimierung der Fläche des Schattens, den ein Polyeder bei parallelen Lichtstrahlen auf eine Ebene wirft



- kombinatorische Struktur des Schattens ist für verschiedene Projektionsrichtungen *u* wieder unterschiedlich
- Partitionierung der Richtungsvektoren: verschiebe alle Ebenen, die eine Seitenfläche enthalten, in den Nullpunkt $\Rightarrow$ dadurch entsteht ein Ebenen-Arrangement im R^3, dessen Ebenen alle durch 0 gehen $\Rightarrow$ vollst. festgelegt durch deren Schnitte mit fester Ebene z=1 (dort: Arrangement *A* von *Geraden*)

Man kann zeigen:
der kleinste Schatten wird angenommen, wenn der Projektionsrichtungsvektor *u* einem Knoten von *A* entspricht (McKenna & Seidel 1985).

damit Algorithmus:
Bestimme *A* (Zeit: $O(n^2)$); für jeden der $O(n^2)$ Knoten *v* von *A* bestimme den Schatten von *P* in einer Ebene senkrecht zur durch *v* bestimmten Richtung;
Ausgabe: dasjenige *v* mit der minimalen Schattenfläche.

Durch Ausnutzung von Kohärenz bei der Neuberechnung der Schattenfläche Reduktion des Gesamt-Zeitaufwandes auf $O(n^2)$.