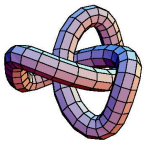
TECHNISCHE UNIVERSITÄT CHEMNITZ
FAKULTÄT FÜR
MATHEMATIK



Mathematische Grundlagen der Computergeometrie

(Vorlesung: Dr. M. Pester)

Inhalt:

6 Konvexe Hülle

6.1 Begriffe

Per Definition ist die *konvexe Hülle* für eine Menge S von endlich vielen Punkten die kleinste konvexe Menge, die S enthält (z.B. in der Ebene durch ein umspannendes Gummiband).

Konvexe Menge: Für zwei beliebige Punkte P_1, P_2 liegt auch die Verbindungsstrecke $\overline{P_1P_2}$ vollständig in der Menge: $\alpha P_1 + (1-\alpha)P_2, \quad \alpha \in [0,1]$

Anwendungen: Bildverarbeitung, Mustererkennung, ...

Definitionen:
Geg. $p_1, p_2, \ldots, p_k \in \mathbb{E}^d$ (verschiedene Punkte im d-dimensionalen Euklidischen Raum $\mathbb{E}^d$)

- Die Menge aller Punkte $\{p = \alpha_1 p_1 + \ldots + \alpha_k p_k,\ \alpha_j \in \mathbb{R}\ \sum \alpha_j = 1,\ \}$
  ist die durch $p_1, \ldots, p_k$ generierte *affine Menge.*
  p ist eine *affine Kombination* von $p_1, \ldots, p_k$.

  Bem.: für $k = 2$ eine Gerade durch p_1, p_2,
  i.a. Punkt, Gerade, Ebene, Hyperebene
  affine Mengen sind *verschobene* lineare Teilräume,
  lineare Teilräume enthalten immer den Ursprung

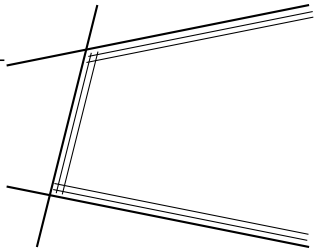
- Die Punktmenge $\{p = \alpha_1 p_1 + \ldots + \alpha_k p_k,\ \alpha_j \geq 0,\ \sum \alpha_j = 1\}$
  ist die durch $p_1, \ldots, p_k$ generierte *konvexe Menge.*
  p ist eine *konvexe Kombination* von $p_1, \ldots, p_k$.

  Bem.: für $k = 2$ die Strecke $\overline{p_1p_2}$

- Für eine beliebige Teilmenge L von Punkten aus $\mathbb{E}^d$ bezeichnet $\mathrm{conv}(L)$ die *konvexe Hülle* von L (die kleinste konvexe Menge, die L enthält)

  Betrachten hier nur endliche Mengen L (Punkte)

- Eine *polyedrale Menge* in $\mathbb{E}^d$ ist der Durchschnitt einer endlichen Menge von Halbräumen aus $\mathbb{E}^d$.
  Eine polyedrale Menge ist konvex;
  sie kann begrenzt oder unendlich sein.



- Ein *konvexes d-Polytop* ist eine begrenzte d-dimensionale polyedrale Menge.
  ($d = 3 \ \rightarrow$ Polyeder, $d = 2 \ \rightarrow$ Polygon)

Satz 1 [McMullen-Shephard, 1971]
Die konvexe Hülle einer endlichen Menge von Punkten in $\mathbb{E}^d$ ist ein konvexes Polytop; und umgekehrt: ein konvexes Polytop ist die konvexe Hülle einer endlichen Punktmenge.

Polytop wird durch seinen Rand (=Polytope niedriger Dimension, *Faces*) beschrieben.

k-face : k-dimensionales Rand-Polytop.

k	Bezeichnung	Speziell $d = 3$
$d-1$	*facets*	Flächen (Polygone)
$d-2$	*subfacets*	
...		
1	*edges*	Kanten
0	*vertices*	Ecken (Knoten)

d-Polytop	=:	d-face
leere Menge	=:	(-1)-face

Wenn P die konvexe Hülle von S in $\mathbb{E}^d$ ist, dann ist ein Face von P die konvexe Hülle einer Teilmenge $T \subseteq S$; aber: nicht jede Teilmenge von S bestimmt ein Face von P

Simplex ist ein Spezialfall eines d-Polytops, welches die konvexe Hülle von $d+1$ (affin unabhängigen) Punkten ist.

In diesem Fall ist jede Teilmenge der Punktmenge S ein Face von P und selbst wieder ein Simplex.

Beispiele: Punkt ($d = 0$), Kante ($d = 1$), Dreieck ($d = 2$), Tetraeder ($d = 3$)

Jedes k-face enthält 2^{k+1} Faces der Dimensionen $k, k-1, \ldots, 0, -1$

Beispiel Tetraeder (3-face):			
	1	3-face	(sich selbst)
	4	2-faces	(Dreiecke)
	6	1-faces	(Kanten)
	4	0-faces	(Ecken)
	1	-1-face	(leere Menge)

Anzahl der d-Faces ist von *kombinatorischer* Art, sie kann durch Binomialkoeffizienten ausgedrückt werden:

$$F(d,N) = \begin{cases} \frac{2N}{d} \begin{pmatrix} N - \frac{d}{2} - 1 \\ \frac{d}{2} - 1 \end{pmatrix} & d \text{ gerade} \\ 2 \begin{pmatrix} N - \lfloor \frac{d}{2} \rfloor - 1 \\ \lfloor \frac{d}{2} \rfloor \end{pmatrix} & d \text{ ungerade} \end{cases}$$

$$F(d,N) = O\left(N^{\lfloor \frac{d}{2} \rfloor}\right)$$

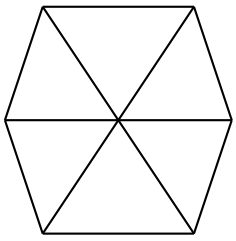
Exponentieller Anstieg (mit d) des zur Darstellung benötigten Speicherplatzes.

Wichtige Fälle $d = 2, 3$ einfacher:

$d = 2$: konvexes Polygon $\rightarrow$ geordnete Folge von Knoten

$d = 3$: Polyeder $\rightarrow$ Angabe der Knoten, Kanten, Faces

$v = 7, e = 12, f = 7$



Eulersche Formel für ebenen Graphen: mit v =Anzahl Knoten, e =Anzahl Kanten, f =Anzahl Faces (einschließlich des äußeren Bereichs):

$$v - e + f = 2$$

bzw. für einfach zusammenhängendes Gebiet (ohne *Löcher* und ohne Außenbereich:

$$v - e + f = 1$$

Es gilt:

$$\left.\begin{array}{ll} v \le \frac{2}{3}e, & e \le 3v - 6 \\ e \le 3f - 6, & f \le \frac{2}{3}e \\ v \le 2f - 4, & f \le 2v - 4 \end{array}\right\} \quad \Rightarrow \quad v \sim e \sim f$$

Speicherbedarf: $O(N)$

Bem.: Im 3D-Fall gilt zusätzlich mit s =Anzahl der Volumenelemente für einfach zusammenhängende Volumen-Netze:

$$v - e + f - s = 1$$

6.2 Zur Bestimmung der konvexen Hülle

Bezeichnungen: $\mathrm{conv}(L)$ – konvexe Hülle
$\mathrm{CH}(L)$ – Rand von $\mathrm{conv}(L)$
(meist wird beides als „konvexe Hülle“ bezeichnet.

Gegeben sei eine Menge S von N Punkten in $\mathbb{E}^d$.

Problem (1): Bestimme die konvexe Hülle $\mathrm{CH}(S)$

Problem (2): Bestimme die Punkte aus S, welche die Ecken von $\mathrm{conv}(S)$ sind.

Betrachtung des Aufwandes (Komplexität) nur für ebenes Problem (untere Grenze in 2D gilt auch für 3D).

Problem (1) ist von der Komplexität her vergleichbar mit Sortieralgorithmus: $O(N \log N)$, mit nur linearem Overhead.

Bsp.: Sortieren von $\{x_1, \ldots, x_N\}$ kann auf Problem (1) zurückgeführt werden: Punkte $\left(x_i, x_i^2\right)$ liegen auf der Parabel $y = x^2$; die konvexe Hülle (Polygon) entspricht der sortierten Folge der x_i.

Problem (2) entspricht einem Entscheidungsproblem (decision-tree technique von Ben-Or.

Problem(3): Sind N gegebene Punkte in der Ebene die Ecken ihrer konvexen Hülle?

Typische primitive Operation: Für drei Punkte $p, p^{(1)}, p^{(2)}$ ist zu entscheiden, ob p links oder rechts oder auf der gerichteten Strecke $\overline{p^{(1)}p^{(2)}}$ liegt. Benutzen dazu den vorzeichenbehafteten Flächeninhalt des Dreiecks $\triangle\left(p, p^{(1)}, p^{(2)}\right)$:

$$\triangle = \begin{vmatrix} x & y & 1 \\ x^{(1)} & y^{(1)} & 1 \\ x^{(2)} & y^{(2)} & 1 \end{vmatrix}$$

6.3 Algorithmen für die konvexe Hülle in der Ebene

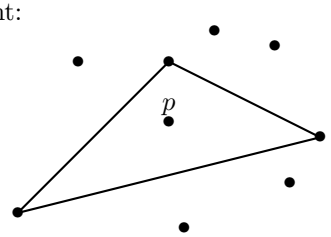
6.3.1 Herleitung eines solchen Algorithmus

Def. 1 *Der Punkt p einer konvexen Menge S heißt Extrempunkt, wenn keine zwei Punkte $a, b \in S$ existieren, so dass p auf der offenen Strecke $\overline{(a, b)}$ liegt.*

$E = \{$alle Extrempunkte von $S\}$
E ist kleinste Teilmenge von S mit: $\mathrm{conv}(E)$=$\mathrm{conv}(S)$.
E ist genau die Menge der Ecken von $\mathrm{conv}(S)$.

Damit wird die konvexe Hülle durch folgende Aktionen bestimmt:

1. Extrempunkte identifizieren
2. Diese Punkte ordnen, so dass sie ein konvexes Polygon bilden



Satz 2 *Ein Punkt p ist kein Extrempunkt von S nur dann, wenn er in einem Dreieck liegt, dessen Ecken Punkte von S sind und p nicht selbst Ecke dieses Dreiecks ist.*
Punkte lassen sich somit ausschließen mit einem Aufwand von $O\left(N^4\right)$ Operationen:
$O\left(N^3\right)$ Dreiecke gibt es und N Punkte (Test „innen/außen“ ist konstanter Aufwand).

Verständlicher Algorithmus, sehr uneffektiv, aber wenigstens endlich.

Satz 3 *(Sei F eine konvexe Figur.) Ein Strahl, von einem inneren Punkt ausgehend, schneidet den Rand in genau einem Punkt.*

Satz 4 *Aufeinanderfolgende Punkte eines konvexen Polygons treten in geordneter Folge bzgl. der von einem inneren Punkt gemessenen Winkel auf.*

Konstruktion der konvexen Hülle, wenn die Extrempunkte bekannt sind:

1. inneren Punkt bestimmen, z. B. Mittelpunkt als arithmetisches Mittel aller Extrempunkte
2. Sortieren nach Polarwinkel bzgl. des inneren Punktes (als Ursprung eines Polarkoordinatensystems).

Winkel müssen nicht exakt ausgerechnet werden, um zu entscheiden, welcher größer ist. Es reicht, den vorzeichenbehafteten Flächeninhalt des Dreiecks $\triangle OP_1P_2$ zu betrachten (positiver Flächeninhalt: P_2 hat größeren Polarwinkel als P_1).
Insgesamt: $O\left(N^4\right)$ Aufwand, nur arithmetische und Vergleichsoperationen.
mögliche Verbesserungen:

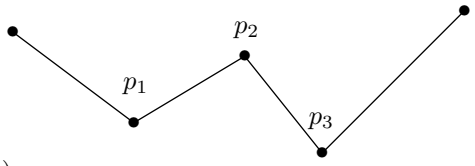
- redundante Berechnungen vermeiden
- anderer theoretischer Zugang

6.3.2 Graham's Scan

Um festzustellen, ob P in einem Dreick liegt, muss man nicht alle durchsuchen.
Graham, 1972: erst sortieren, dann Extrempunkte finden.

1. inneren Punkt finden $\rightarrow$ Ursprung O
2. alle Punkte sortieren nach Polarwinkel und Abstand $\overline{OP}$ (letzteres nur bei gleichem Winkel)
3. Sortierte Liste durchgehen und innere Punkte eliminieren. conv(S) bleibt übrig.

Bestimmung innerer Punkte:
$\angle(p_1, p_2, p_3) \geq \pi \Rightarrow p_2$ eliminieren
auch über Flächeninhalt: $\det(p_1p_2p_3) < 0$.

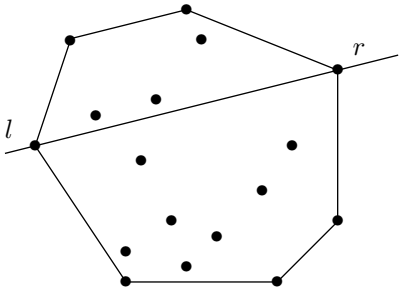


Der Scan-Algorithmus: (doppelt verkettete Liste liege vor)

```
begin
  v:=START; w:=PRED[v]; f:=false
  while (NEXT[v] <> START)  or  (not  f) do
    begin
      if NEXT[v] = w then f:=true;
      if Flaeche(v,NEXT[v],NEXT[NEXT[v]]) > 0
      then v:=NEXT[v]
      else begin DELETE(NEXT[v]); v:=PRED[v] end;
    end
end
```

Algorithmus ist optimal, bis auf den Nachteil der Transformation auf Polarkoordinaten
Modifikation (Andrew, 1976):

- Bestimme linken und rechten Extrempunkt l, r der gegebenen N Punkte.
- Linie durch l und r teilt Punkte in obere und untere Teilmenge.
- Sortieren der beiden Teilmengen nach x-Koordinate liefert zwei Polygone.
- Scan-Algorithmus für beide Polygone anwenden $\rightarrow$ untere/obere Hülle
- Vereinigung ergibt konvexe Hülle



6.3.3 Jarvi's March

Betrachten Beschreibung des Polygons über seine Kanten, statt über die Eckpunkte.

Satz 5 *Eine Strecke l, die durch zwei Punkte aus S bestimmt ist, ist genau dann eine Kante der konvexen Hülle, wenn alle anderen Punkte von S auf genau einer Seite von l oder auf l liegen.*

Für N Punkte existieren $\binom{N}{2} = O\left(N^2\right)$ Paare von Punkten (=Strecken); dafür sind dann noch $N-2$ Punkte zu testen.

$O\left(N^3\right)$ Operationen, um alle Kanten/Punktepaare der konvexen Hülle zu finden.

Jarvi's Verbesserung auf $O\left(N^2\right)$:

Wenn $\overline{pq}$ Kante der konvexen Hülle ist, dann gibt es eine weitere Kante der konvexen Hülle, die bei q beginnt.

Suche jeweils nächsten Punkt mit geringstem polaren Winkel bzg. aktueller Kantenrichtung.

Bem.: $O\left(N^2\right)$ ist „worst-case“, wenn alle N Punkte auch Ecken der konvexen Hülle sind.

Wenn die Anzahl h der Eckpunkte der konvexen Hülle wesentlich kleiner als N ist, wird der Algorithmus mit $O(N \cdot h)$ sehr effektiv.

6.3.4 Quickhull-Techniken

Beruhen auf dem Quicksort-Algorithmus.

- Bestimme linken und rechten Extrempunkt der N Punkte von S
- Die Gerade L durch l, r liefert zwei Teilmengen (nicht notwendig disjunkt, da l, r zu beiden gehören)
  $S^{(1)}$ Punkte auf und oberhalb L
  $S^{(2)}$ Punkte auf und unterhalb L
- Punkt $h \in S^{(1)}$ suchen, so dass $\triangle(hlr)$ maximalen Flächeninhalt hat (bei Gleichheit größeren Winkel $\angle hlr$)
  h gehört zur konvexen Hülle.
- Betrachten dann die Linien L_1 durch l, h und L_2 durch h, r (gerichtete Linien).
  Alle Punkte rechts von beiden Linien werden eliminiert (innere Punkte).
  Punkte links von L_1 und rechts von L_2 : $\Rightarrow$ $S^{(1,1)}$
  Punkte rechts von L_1 und links von L_2 : $\Rightarrow$ $S^{(1,2)}$
- rekursiv weiter ...

```
function QUICKHULL(S,l,r)
begin
  if S={l,r} then return({l,r}) else begin
  h:=FURTHEST(S,l,r);
  S1:={Punkte links von lh};
  S2:={Punkte rechts von hr};
  return(QUICKHULL(S1,l,h)*(QUICKHULL(S2,h,r)-h))
  end
end
```

Bem.: Es reicht mit $l_0 = (x_0, y_0)$ und $r_0 = (x_0, y_0 - \epsilon)$ zu beginnen und am Ende r_0 wieder zu löschen, da der Algorithmus als ersten Punkt h den Punkt r liefert (größter Abstand senkrecht zu $\overline{l_0 r_0}$).