AG Übung vom 15.11.2010

Übungsblatt 0

Konvention: – Entartete Fälle werden nicht betrachtet (Polygone haben mind. 3 Ecken, etc.)

– Polygone sind zusammenhängend

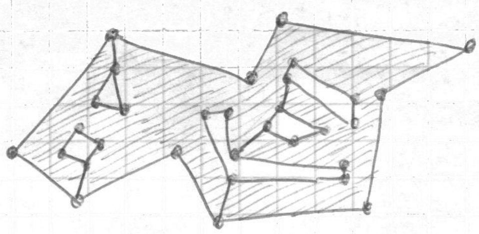
(1) Beweise den Satz aus der VL: „Jedes Polygon mit n Ecken und h Löchern lässt sich triangulieren. Die Zahl der Δe ist $n+2h-2$."

Induktion über n

Start: $n=3 \rightarrow$ klar

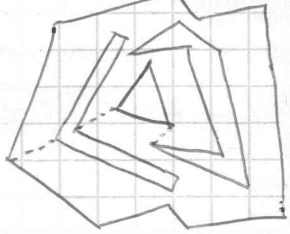
Induktionsschritt: Nehme an, die Beh. gilt für $n-1$ (und kleinere), Zeige: Die Beh. gilt auch für n.

1. Verbinde jedes Loch mit dem Rand oder einem anderen Loch (durch Einfügen einer Diagonale).

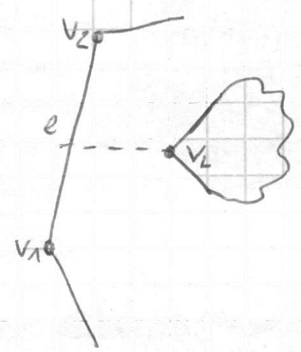


(Und zwar so, dass jedes Loch über andere Löcher oder direkt genau 1x mit dem Rand verbunden ist)

2. Das geht immer:



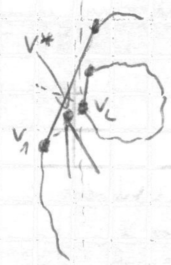
Betrachte z.B. die untere von den linkesten Ecken eines Lochs (v_L). Diese muss konvex sein. Links daneben liegt die Kante e des Polygons. ($e=v_1v_2$)



~~Wenn~~ v_1, v_2 oder beide liegen links von v_L. Sei das OBdA v_1.

Wenn $v_2 v_1$ in P liegt, füge diese ein.

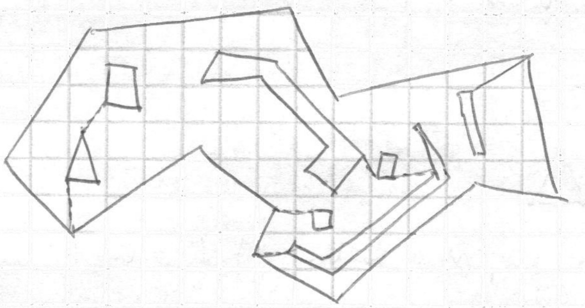
Sonst:



Ansonsten muss es ein v^* geben, das näher an v_2 liegt, beim nächsten muss $v_2 v^*$ in P liegen (analog zur VL).

Beachte: v^* kann nicht zum selben Loch wie v_2 gehören, ob es zum Außenrand oder einem anderen Loch gehört, spielt keine Rolle!

3.



Erhalte ein einfaches Polygon mit $n+2h$ Ecken (2 neue Ecken für jede Loch-Diagonale), das sich nach VL triangulieren lässt, die Zahl der Δe ist dabei $n+2h-2$

(2) Wie viele Diagonalen enthält eine Triangulierung eines Polygons mit n Ecken?

Es gibt $n+2h-2$ Δe, jedes hat 3 Kanten, also insgesamt $3n+6h-6$.

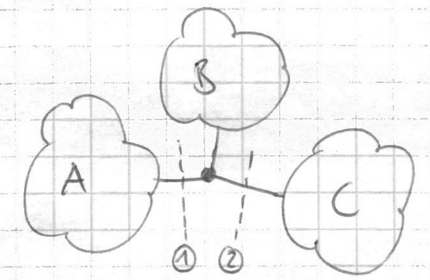
Davon sind n Stück „Außenkanten", Innere gibt es also $2n+6h-6$.

Da jede Diagonale 2 Dreieckskanten zur Verfügung stellt, hat eine Triangulierung $n+3h-3$ Diagonalen.

(3) Zeige, dass in jeder Triangulierung eines einfachen Polygons mit n Ecken eine Diagonale existiert, so dass auf jeder Seite mindestens $\lceil \frac{n}{3}-1 \rceil$ Dreiecke liegen

- Betrachte den dualen Graphen der Triangulierung (Baum)
- Starte bei einem Blatt (= Ohr der Triangulierung)
- Wandere den Baum entlang und wähle immer den größten Teilbaum bis mindestens $\lceil \frac{n}{3}-1 \rceil$ Knoten hinter dir liegen.

mit $n-2$ Knoten



Betrachte dabei den letzten Schritt (① → ②)

Bei ①:	$A < \lceil \frac{n}{3}-1 \rceil$	I
oBdA:	$C \geq B$	II
Bei ②:	$A+B+1 \geq \lceil \frac{n}{3}-1 \rceil$	III
	$n-2 = A+B+C+1$	IV

$$\text{IV} \Rightarrow C = n-A-B-3$$

$$\underset{(\text{II})}{\Leftrightarrow} C \geq n-A-C-3$$

$$\Leftrightarrow 2C \geq n-A-3$$

$$\underset{(\text{I})}{\Rightarrow} 2C > n-3-\lceil \tfrac{n}{3}-1 \rceil$$

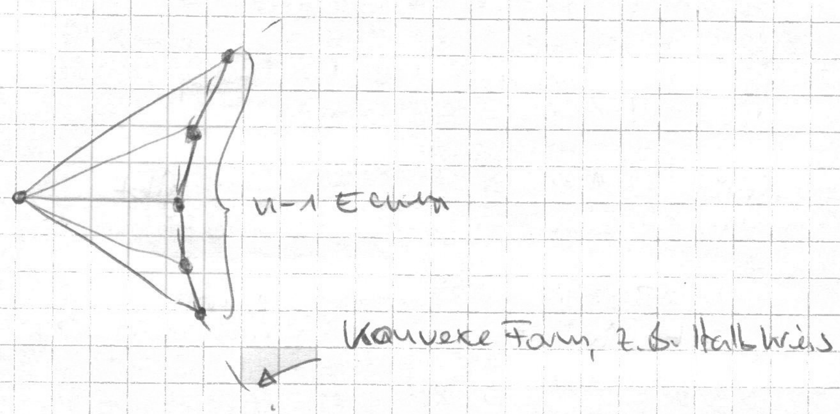
$$\Leftrightarrow 2C \geq n-2-\underbrace{\lceil \tfrac{n}{3}-1 \rceil}_{\leq \frac{n}{3}}$$

$$\Leftrightarrow C \geq \tfrac{1}{2}n-1-\tfrac{1}{6}n = \tfrac{n}{3}-1$$

$$C \text{ ganzzahlig} \Rightarrow C \geq \lceil \tfrac{n}{3}-1 \rceil$$

d.h. auf beiden Seiten der Diagonale bei ② liegen je mindestens $\lceil \frac{n}{3}-1 \rceil$ Dreiecke.

(4) Zeige, dass es für jedes n ein Polygon mit mindestens n Ecken gibt, das genau eine Triangulierung besitzt:



Falls gewähltes $n \leq 2$, Polygon für höheres n erzeugen.

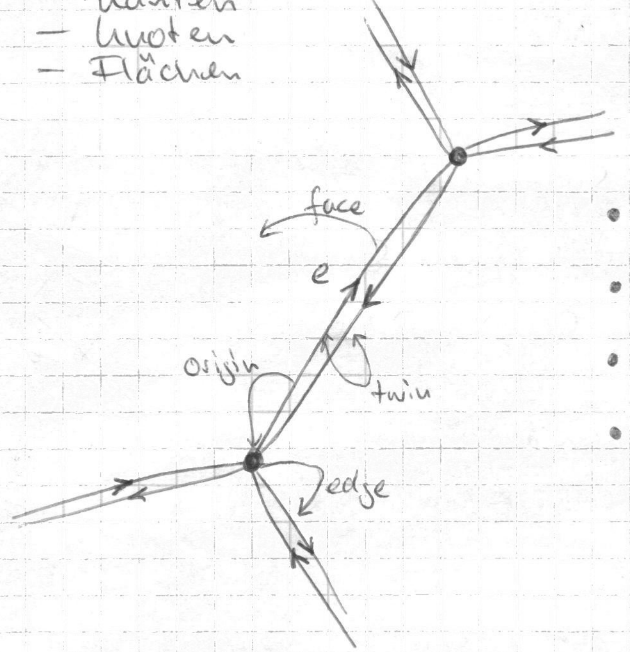
Doubly-Connected Edge Lists

Wie verarbeitet man Polygone, Triangulierungen und andere geometrische Graphen im Rechner?

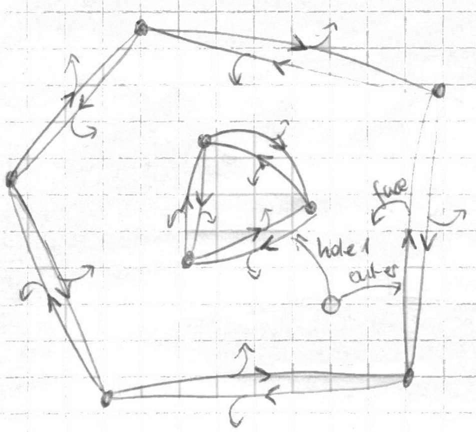
→ DCEL

Alternativen: QEDS, Winged Edge

Bestandteile:
- Halbkanten
- Knoten
- Flächen



- Kante → Halbkanten-Paar („twins")
- e.origin = Knoten, von dem die Kante kommt
- e.destination = e.twin.origin
- Knoten verweisen auf irgendeine ausgehende Kante (v.edge)

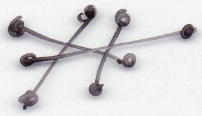


- Halbkanten verweisen auf Flächen links von ihnen
- Halbkanten verweisen auf Nachfolger um ihre Flächen
- Flächen verweisen auf eine Kante pro Komponente

Schnittpunkte von Liniensegmenten

Problem: Gegeben n Liniensegmente in der Ebene, finde alle Schnittpunkte (in kurzer Zeit)

Wie lange dauert das?



Im schlimmsten Fall $O(n^2)$

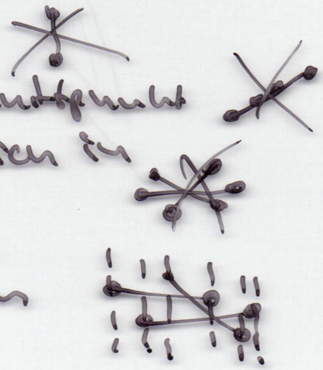
ABER: Das geht schneller!

$O(n \log n + k \log n)$ k = Anzahl der Schnittpunkte

Sweep-Line-Algorithmus für Schnittpunkte

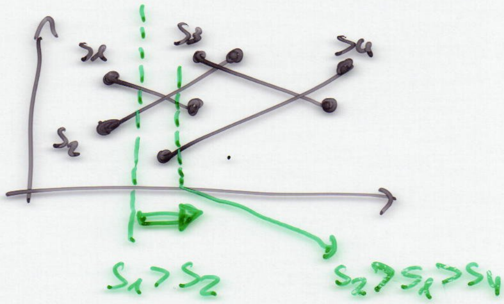
Annahmen

- keine Segmente verlaufen senkrecht
- 2 Segmente haben höchstens einen Schnittpunkt
- Höchstens 2 Segmente schneiden sich in einem Punkt.
- Segment Start- und Endpunkte haben paarweise verschiedene X-Koordinaten



Idee

- Sweep-Line bewegt sich von links nach rechts
- Betrachte Y-Ordnung der Segmente an der Sweep-Line



Betrachte folgende Ereignisse

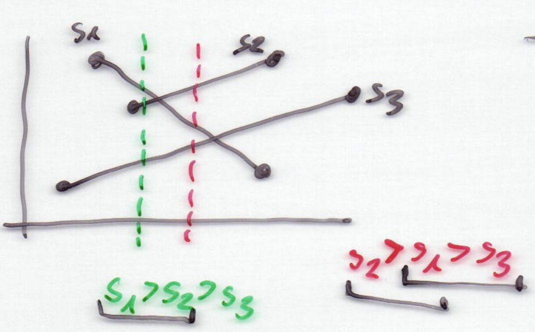
(1) SL erreicht linken Endpunkt eines Segments

(2) SL — " — rechten Endpunkt — " —

(3) SL erreicht einen Schnittpunkt von 2 Segmenten

(1), (2) → Sortiere Endpunkte der Segmente nach X-Koord.

ABER: Was ist mit den Schnittpunkten?



→ Teste Nachbarn in der Y-Ordnung auf Schnitt sobald sich die Nachbarschaft ändert

→ Das passiert nur an Ereignis-Punkten!

Datenstrukturen

- Y-Ordnung
- Ereignisse (Priority Queue)

Ereignisse

(1) linker Endpunkt → Füge in Ordnung ein $\boxed{O(\log n)}$
→ Teste auf Schnitt (max. 2x)

(2) rechter Endpunkt → Entferne aus der Ordnung $\boxed{O(\log n)}$
→ Teste Schnitt der ehem. Nachbarn (1x)

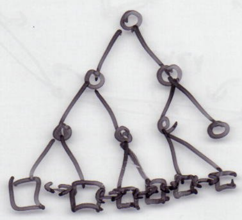
(1)+(2) Gefundene Schnittpunkte in Ereignis-Queue einfügen

$$\boxed{O(\log k) = O(\log n^2) = O(2 \log n) = O(\log n)}$$

(3) Schnittpunkt → Segmente in Ordnung tauschen $O(1)$
→ Teste neue Nachbarn auf Schnitt (max 2x) $O(1)$

Struktur für Y-Ordnung

Balancierter Binärbaum mit verketteten Blättern



Laufzeit

n Segmente
k Schnittpunkte

- $2n + k$ Ereignisse
- Jedes wird in $O(\log n)$ abgearbeitet

$\Rightarrow O((n+k) \log n)$