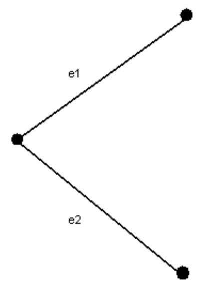
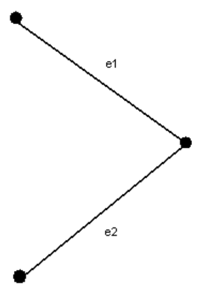
Konstruktion der „inneren“ Traingulierung mit dem Sweepline-Verfahren

Zuerst wird das EPS mit den x-Koordinaten der Ecken initialisiert. Dann startet man die Sweepline links des Graphens. Die Sweepline durchläuft den Graphen und schneidet dabei die Ecken des Polygons. Dabei unterscheidet man drei verschiedene Sorten von Ecken.

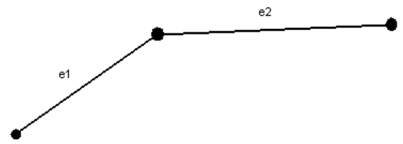
1. Start-Ecke:
Die Ecke startet ein neues Intervall I



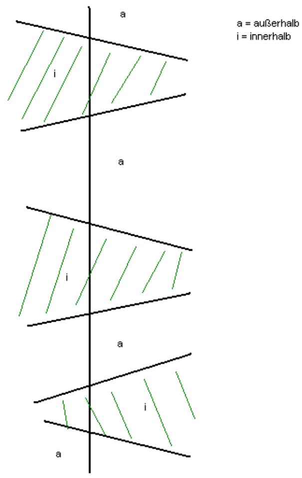
2. End-Ecke:
Die Ecke beendet ein Intervall I



3. Durchgangs-Ecke:
Die Ecke ändert das Intervall I nicht



Das SLS enthält alle „aktiven“ Kanten.
Die Sweepline schneidet immer eine gradzahlige Anzahl von Kanten, da die Sweepline Polygone schneidet.

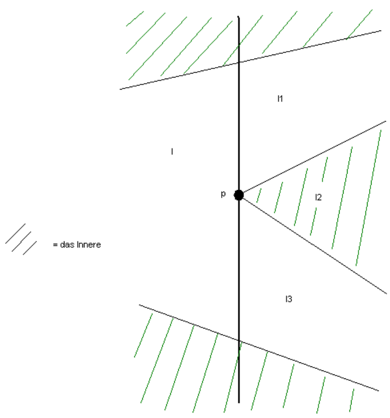


Zu jedem Intervall I der Sweepline wird ein Zeiger auf einen Polygonzug V1,...,Vk gespeichert. Entweder ist $\overline{V_i V_{i+1}}$ schon eine Kante von p oder bereits eine konstruierte Kante der Triangulierung. Der Zeiger zeigt von Intervall I zur rechtesten Kannte des Polygonzugs V1,...,Vk (d.h. auf den Punkt mit der grten x-Koordinate)

Veränderungen der Ereignispunkte p:

<u>1. Fall:</u> p ist Startpunkt
Intervall I ist Teil der Sweepline und Intervall I enthält p

1.1. Intervall I ist ein a-Intervall. (a-Intervall heisst, dass das Intervall „ausserhalb“ eines Polygons liegt)
Das Intervall I wird in drei Intervalle I1, I2 und I3 gespalten.
I2 zeigt auf p (I2 ist ein Polygonzug der Lnge 0)



1.2. Intervall I ist ein i-Intervall (i-Intervall heisst, dass das Intervall „innerhalb“ eines Polygons liegt)
V_l ist rechtester Knoten des zu Intervall I gehenden Polygonzugs. $\overline{PV_l}$ wird Traingulierungskante, sowie die Kanten $\overline{PV_{l-1}}$,$\overline{PV_{l-2}}$,....,Kante soweit wie möglich bzw. $\overline{PV_{l+1}}$,$\overline{PV_{l+2}}$...,Kante soweit wie möglich ⇒ soweit wie möglich heisst das Ende des Polygonzugs ist erreicht wird oder der Winkel zwischen V_{i-1},V_i,p $>\pi$ bzw. Winkel zwischen p,V_j,V_{j+1} $>\pi$
(π bedeutet das der Winkel größer 180°ist.)
Spalte I in I_1, I_2, und I_3 auf.
I_1: V_i,V_{i-1},p mit p rechtester Punkt
I_3: p,V_j,....,V_k mit p rechtester Punkt

2. Fall: p ist Durchgangs-Kante

Unterhalb von p liegt das „Innere“des Polygons. Man fügt eine Triangulierungskante von p aus nach V_l ein, dann weitere Kanten von V_l ausgehend nach unten bzw. oben und soweit wie möglich, siehe dazu 1.2.
Daraus resultiert ein neuer Kantenzug fr I: p,V_j,....,V_k mit p rechtester Knoten

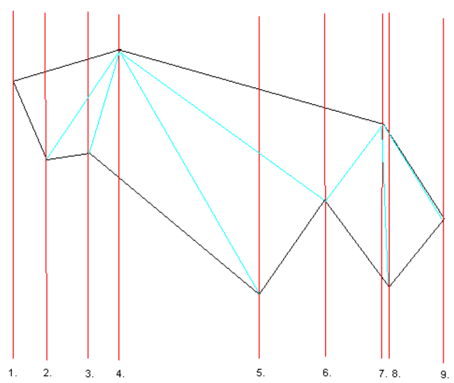
Analog dazu, wenn das „Innere“des Polygons oberhalb von p liegt.

3. Fall: p ist End-Ecke

3.1. I ist i-Intervall
Füge Triangulierungskanten nach allen Ecken des Kantezuges von I ein.
I wird entfernt und I_1 und I_2 zu I vereinigt.

3.2. I ist a-Intervall
V_1,....,V_k ist der Polygonzug zu I_1
W_1,....,W_l ist der Polygonzug zu I_2
ziehe Traingulierungskante von p zu den Polygonzügen $\overline{PV_{k-1}}$,....,soweit wie möglich bzw. $\overline{PW_2}$,.....,soweit wie möglich (zu „soweit wie möglich“ siehe 1.2.)
vereinige I_1 und I_2 zu I,
neuer Kantenzug zu dem neuen vereinigten Intervall I: V_1,...,V_k,P,w_1,....,w_l

Beispiel:



Das Beispiel zeigt die verschiedenen Sweeplines (1.-9.) und die daraus entstandene Triangulierung des Polygons.
Der Algorithmus trianguliert das Innere des Polygons.

Laufzeit des Algorithmus für ein n-Eck P:

- Initialisierung des EPS: O (n log n)
- bei Veränderungen des EPS und des SLS: O (log n)
  (bei geeigneter Wahl der Datenstruktur: z.B. Heap (EPS) und balanvierter Suchbaum (SLS))
- O(1) Zeit pro eingefügte triangulierte Kante ⇒ O(n) insg. für den gesamten Graph, da O(n) viele triangulierte Kanten
  (Polygonzüge als verkettete Liste speichern)

⇒ insg. O (n log n) Zeit