Übung 2

Aufwärmproblem

Gegeben: n Segmente im $\mathbb{R}^2$ $s_1, \dots, s_n =: S$

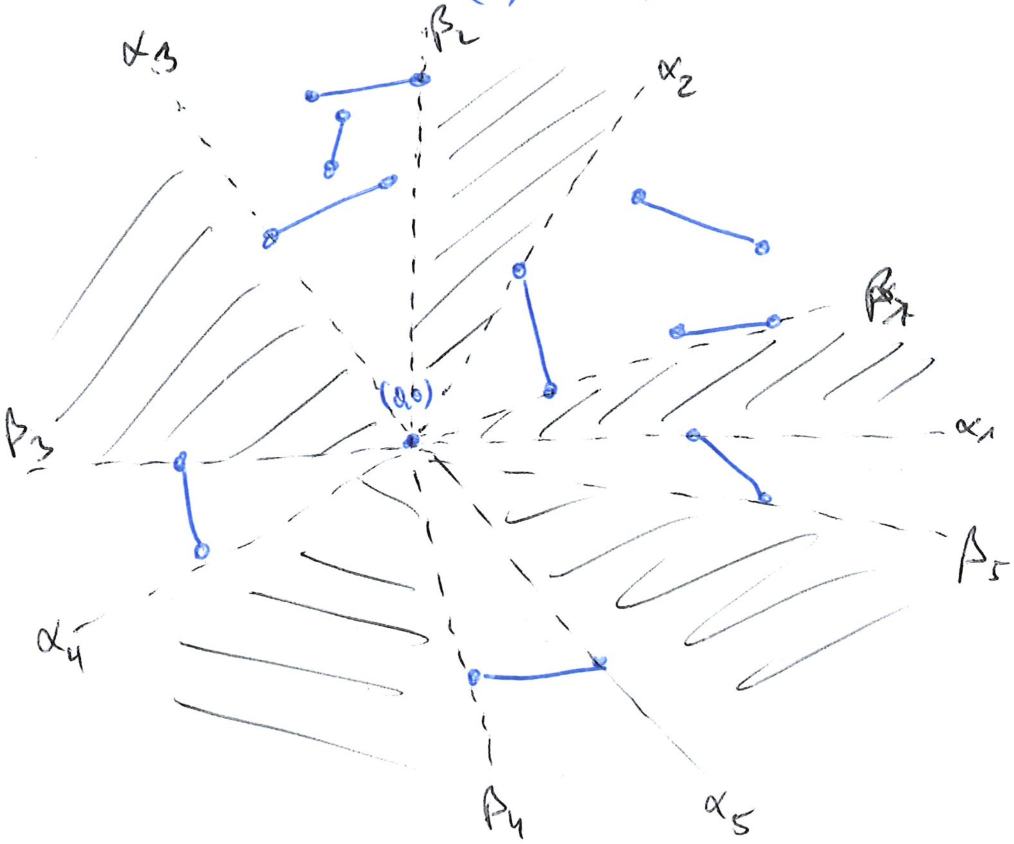
Gesucht: Intervalle $[\alpha_i, \beta_i]$ mit

$I = \bigcup [\alpha_i, \beta_i]$ maximal

und $K(I) \cap S = \emptyset$

S_i sind hier als offene Mengen zu betrachten, d.h. Endpunkte dürfen auf dem Rand eines Intervals liegen.

$K(I) \doteq$ Kreiskegel aller Intervalle mit Ursprung $(0,0)$



Wie löst man das mit einem Sweep-Line-Algorithmus?

- Plane-sweep (also eine gerade Linie von links nach rechts) scheint nicht sinnvoll.
- Betrachte Angular-sweep: Ein Strahl, der sich um einen Punkt dreht.

Idee:

- Sortiere Endpunkte der Segmente bzgl. ihres ccw-Winkels zur x-Achse (→ Eventpoints)

- Für jeden Eventpoint:
  - zähle einen Zähler hoch, wenn ein neues Segment beginnt.
  - zähle einen Zähler herunter, wenn ein Segment endet

- Gebe ein Intervall aus, wenn Zähler auf 0 fällt bis er wieder auf 1 geht

Laufzeit

- Sortieren $O(n \log n)$
- $2n$ Event points, je $O(1)$ Bearbeitungszeit
  $\Rightarrow O(n)$

$\Rightarrow O(n \log n)$

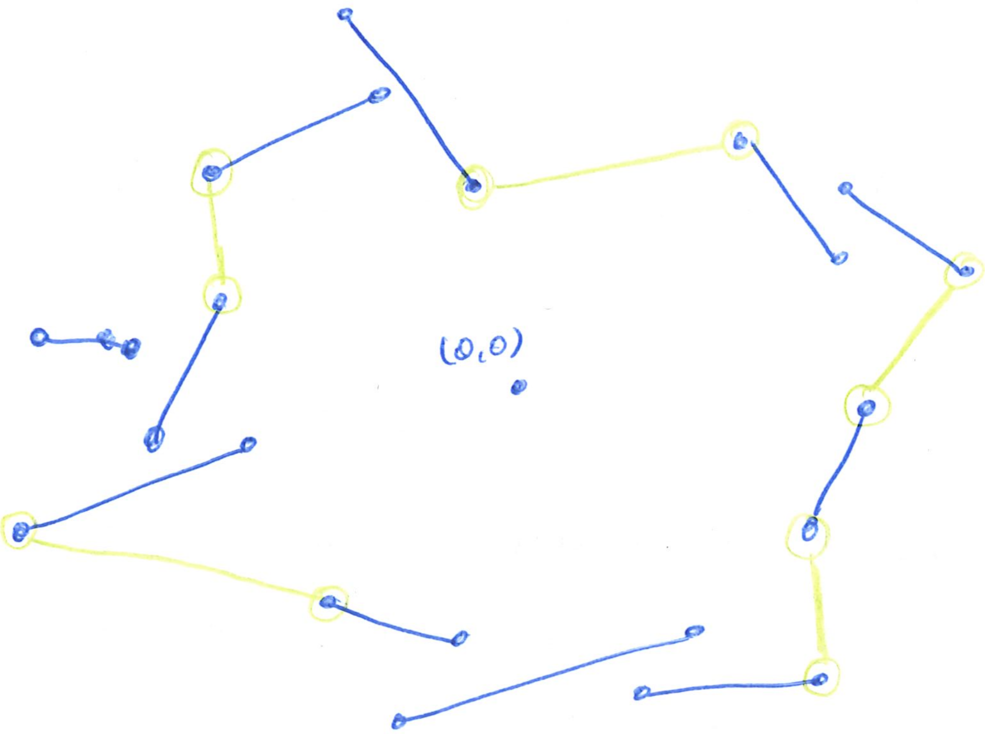
EAG 2.2

Zu beachten:

Wie codieren wir das Intervall?

Problem: Die Winkel sind ggf. nicht exakt darstellbar (irrational), auch wenn die Inputdaten/Segmentenden rationalen Zahlen entsprechen.

Wie umgeht man das Problem?



Anstatt die Winkel zuspeichern, beschreibe die Intervalle über Segmente.

Problem 2

Gegeben: n Punkte $P_1, \dots, P_n =: P$

Gesucht: ~~Geordnete Menge~~ konvexes Polygon $CH = (P'_1, \dots, P'_k)$ mit
$P'_i \in P$
und ~~Polygon CH enthält~~ jeder Punkt aus P
liegt innerhalb oder auf CH.

(CH wird auch Convex Hull genannt)

Entwirf einen Angular Sweep, um das Problem zu lösen.

Was nimmt man als Drehpunkt für den Angular Sweep?

Wichtig: Die Punkte, die später auf CH liegen, müssen bzgl. des Sweeps entlang von CH sortiert sein.
→ Genau dann der Fall, wenn der Drehpunkt in oder auf CH liegt; Am besten auf!

Können wir schnell entscheiden, welcher Punkt der Eingabe auf CH liegen muss?

Sei $p \in P$ der Punkt mit der kleinsten y-Koordinate
(ist p nicht eindeutig, wähle darunter denjenigen mit der kleinsten x-Koordinate)

→ p wird auf CH liegen. (Beweis selbst)

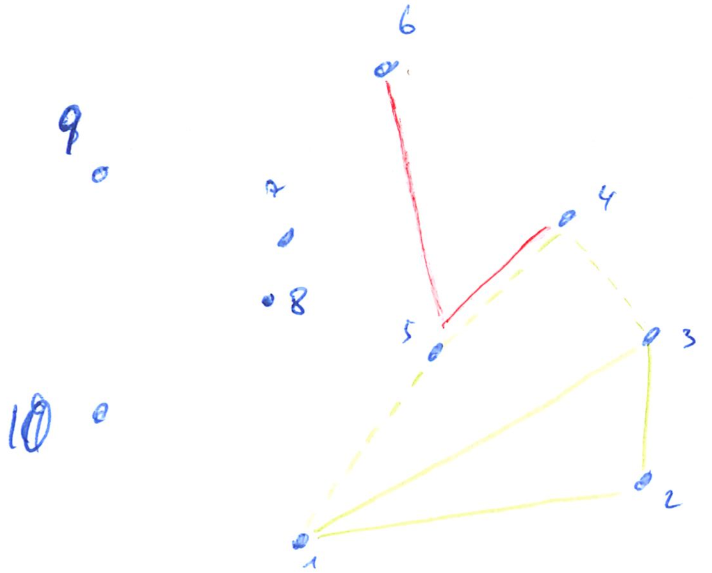
→ Sortiere P bzgl. des ccw Winkels zwischen der hor. Linie durch p ~~und~~ zu $\overline{pp'}$, $p' \in P$.

Idee: Gehe die sortierte Liste durch und baue CH nach und nach auf.

Benutze als Invariante:

Nach Abarbeiten des i-ten Punktes haben wir ~~die~~ CH bzgl. der ersten i Punkte.

Die ersten drei Punkte der Reihenfolge bilden ein korrekt orientiertes, konvexes Polygon (1, 2, 3)

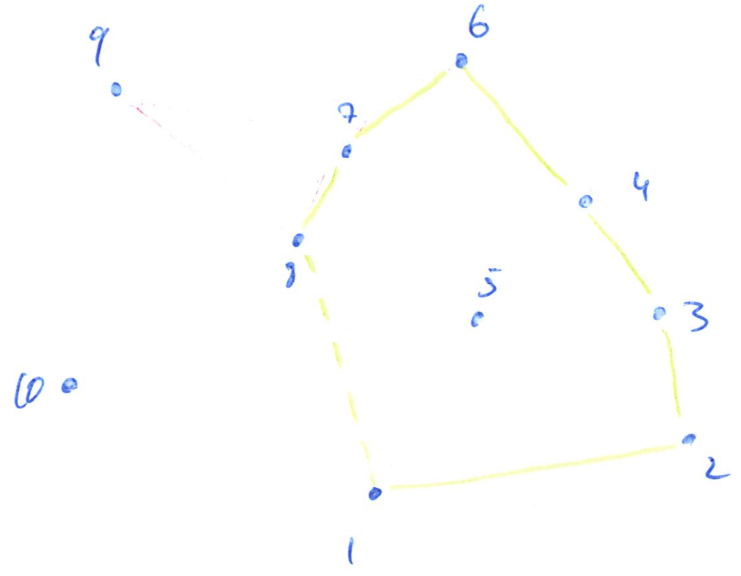


Im Beispiel können wir 4 und 5 aufnehmen, ohne die Invariante zu verletzen.

Sind wir bei Punkt 6, sehen wir, dass (4, 5, 6) einen ungültigen Winkel für ein konvexes Polygon bilden (alle ~~…~~ Innenwinkel < 180°) ⇒ 5 kann nicht auf CH liegen!

→Entferne 5 und fahre fort.

→~~Knote~~ Punkt 6 kann nun aufgenommen werden.

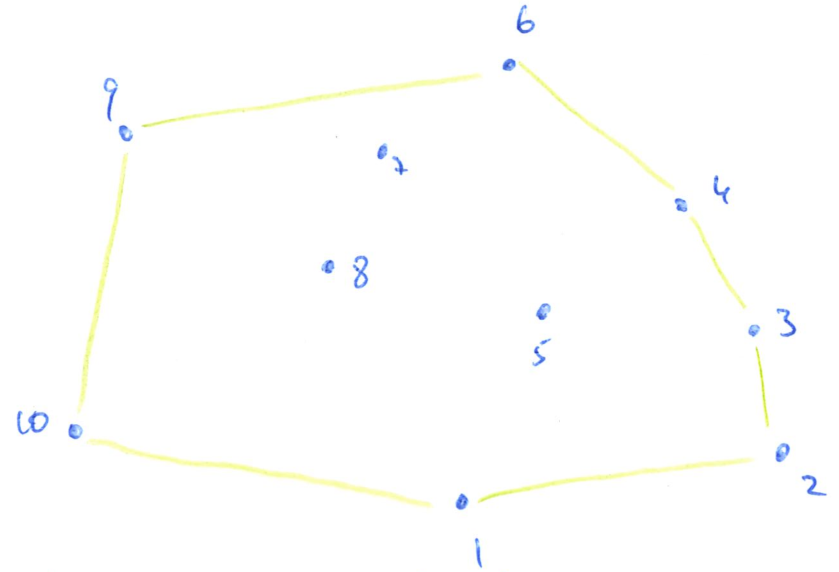


Bis 8 nehmen wir alles auf.

Für Punkt 9 bildet (7, 8, 9) einen falschen Winkel!

→entferne 8.

Nun erkennen wir, auch (6, 7, 9) bildet einen falschen Winkel ⇒ entferne auch 7.



→Fahre normal fort.

Also, für jeden Knoten p_i in der RF:

- Betrachte letzte beiden Knoten p_{k-2}, p_{k-1} auf CH
- Ist (p_{k-1}, p_{k-2}, p_i) ein ungültiger Winkel, lösche p_k und beginne erneut für p_i
- Andernfalls nimm p_i in CH auf.

Das lässt sich bspw. mit einem Stack implementieren!

Wie ist die Laufzeit?

- $O(n)$ Startpunkt suchen
- $O(n \log n)$ Sortieren
- $O(n)$ Eventpoints
  ↳ maximal $O(n)$-Mal etwas löschen

$\Rightarrow O(n^2)$? Nein!

Jeder Knoten kann maximal 1-Mal gelöscht werden
$\Rightarrow$ ALLE Löschoperationen zusammen sind $O(n)$
$\rightarrow$ im Schnitt also $O(1)$ pro Eventpoint

$\Rightarrow$ Laufzeit ist $O(n \log n)$