Triangulierung einfacher Polygone (Fortsetzung)

Wir können mit dem Sweepline-Verfahren aus der letzten Vorlesung das Innere eines Polygons in O (n log n) Zeit triangulieren.

gesamte Triangulierung:

- konstruiere konvexe Hülle von P
- trianguliere mit vorigem Verfahren das Innere von P und alle zwischen dem Rand der konvexen Hülle und P entstehenden „Buchten“
- ziehe Strahlen von den Extrempunkten der konvexen Hülle nach ∞

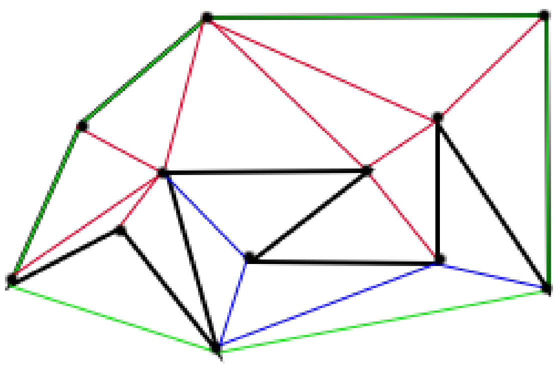


Abbildung 1: Triangulierung des Inneren (rot) und einer „Bucht“ (blau) zwischen dem Polygon und dem Rand der konvexen Hülle (grün)

Laufzeit:
Sei P' ein zu bearbeitendes Teilpolygon (das Innere oder eine Bucht) mit kp' Ecken, dann brauchen wir dafür $O\ (kp'\ log\ kp')$ Zeit.

also insgesamt:

$$\sum_{P'\ Teilpolygon} O\ (kp'\ log\ kp')\ , \quad wobei \sum_{P'} kp' \leq 3\ n \quad \textit{(höchstens zwei Buchten und das Innere liegen an einerEcke an)}$$

$$= \quad O\left(\sum_{P'}(kp'\ log\ kp')\right)$$

$$= \quad O\left(\sum_{P'}(kp'\ log\ n)\right)$$

$$
\begin{aligned}
&= \quad O\left((\sum_{P'} kp')\ log\ n \right) \\
&= \quad O\ (n\ log\ n)
\end{aligned}
$$

Satz: *Obiger Sweepline-Algorithmus trianguliert ein einfaches n - Eck in O (n log n) Zeit.*

Anwendungen:

- *Numerische Mathematik*: Methode der finiten Elemente zur numerischen Lösung von Differentialgleichungen
- *Computer-Graphik*: Viele Probleme sind mit konvexen Polygonen leichter lösbar; zum Beispiel: Gartenzaun und Hintergrund → Schnitt zweier beliebiger Polygone → Zerlegung in konvexe Polygone

Satz: *Sei P ein konvexes n - Eck, Q ein einfaches m - Eck und eine Triangulierung von Q gegeben. Dann kann P ∩ Q in O (n + m) Zeit berechnet werden.*

Beweis: → *Übung*

Da Triangulierungen meist sehr aufwändig zu handhaben sind, will man eigentlich eine Zerlegung in konvexe Polygone.

Zerlegung eines einfachen Polygons in möglichst wenige konvexe Teile

gegeben sei ein einfaches Polygon P

„greedy" Algorithmus:

- trianguliere Inneres von P → Triangulierung T
- durchlaufe Triangulierungskanten in beliebiger Reihenfolge, entferne eine Kante immer dann, wenn dadurch keine konkave Ecke entsteht

Behauptung: *Falls n_{opt} die minimale Anzahl von konvexen Polygonen ist, in die sich P zerlegen lässt, so liefert obige Methode höchstens $4n_{opt}$ - 3 Teile.*

Anmerkung: Dies ist also ein „Approximationsalgorithmus", er liefert eine Lösung, die höchstens um einen konstanten Faktor schlechter als das Optimum ist. Approximationsalgorithmen werden meist für NP-vollständige Probleme untersucht. Für dieses Problem existiert allerdings auch ein Optimierungsalgorithmus der Laufzeit $O(n^2c^2)$, wobei c die Anzahl der nichtkonvexen Ecken von P ist.

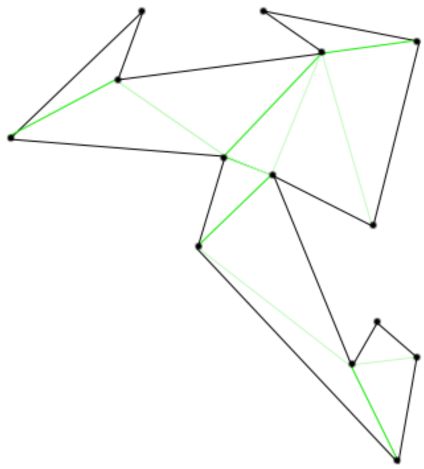
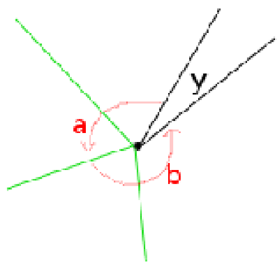


Abbildung 2: Triangulierungskanten nacheinander entfernen

Beweis: *Ordne jede noch verbliebene Triangulierungskante einer (nichtkonvexen) inzidenten Ecke zu, um derentwillen sie geblieben ist. Dann werden jeder nichtkonvexen Ecke* ≤ 2 *Kanten zugeordnet.*



$$\begin{aligned} & \ldots \textit{denn angenommen} \geq 3 \\ \Rightarrow \quad & \alpha \geq \pi,\ \beta \geq \pi,\ \textit{weil sonst eine} \\ & \textit{Kante herausgenommen werden} \\ & \textit{könnte} \\ \Rightarrow \quad & \alpha + \beta \geq 2\pi \textit{ also } \gamma \leq 0 \Rightarrow \lightning \end{aligned}$$

. . . also gibt es höchstens *2c Zerlegungskanten, also*

$$\begin{aligned} n_{app} &\leq 2c + 1 \\ c &\geq \frac{n_{app} - 1}{2} \end{aligned}$$

Jede Zerlegung muss mindestens $\frac{c}{2}$ *Zerlegungskanten haben, also*

$$\begin{aligned} n_{opt} &\geq \frac{c}{2} + 1 \\ n_{opt} &\geq \frac{1}{2}\left(\frac{n_{app} - 1}{2}\right) + 1 \\ n_{app} &\leq 4n_{opt} - 3 \end{aligned}$$

$\square$

Sichtbarkeit

Sei P ein einfaches Polygon und q ein Punkt im Inneren.

Definition: *$Sicht(q, P) = \{ r \mid \overline{qr}$ schneidet keine Kante von $P\}$ ist der* Sichtbarkeitsbereich.

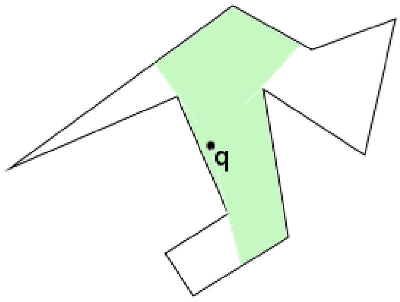


Abbildung 3: Sichtbarkeitsbereich

Definition: *Die Menge $\{ q \mid Sicht(q, P) = Inneres$ von $P\}$ ist der* Kern *von P.*

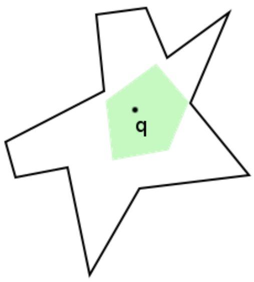


Abbildung 4: Kern eines sternförmigen Polygons

<u>Problem</u>: gegeben q und P, berechne S = Sicht(q, P)

<u>Lösung</u>:

1. trianguliere das Polygon P und betrachte den Dualen Graphen B zur Triangulierung
2. bestimme das Dreieck d, das q enthält
3. von d ausgehend mit Breitensuche auf B, bestimme für jedes Dreieck welche Teile von q aus sichtbar sind

Die Vereinigung der Teile ist Sicht(q, P).

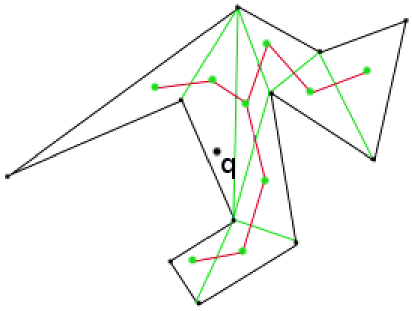


Abbildung 5: Dualer Graph B zur Triangulierung

Behauptung: *B ist ein Baum.*

Beweis: $\rightarrow$ *Übung*