Technische Universität Braunschweig

Einführung in algorithmische Geometrie

Arne Schmidt

Vorstellung

Interessen

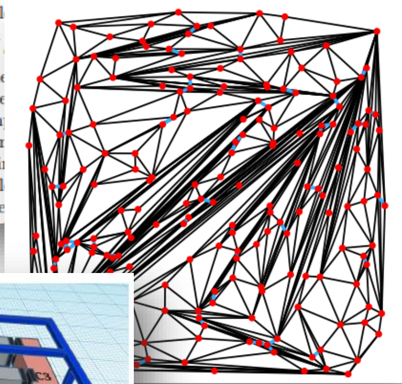
- Geometrische Optimierung,
- Programmierbare Materie,
- Komplexitätstheorie

#GernePerDu

Computing MaxMin Edge Length Triangulations

Sándor P. Fekete * Winfried Hellmann* Michael Hemmer* Arne Schmidt* Julian Troegel*

Abstract



Particle-Based Assembly Using Precise Global Control*

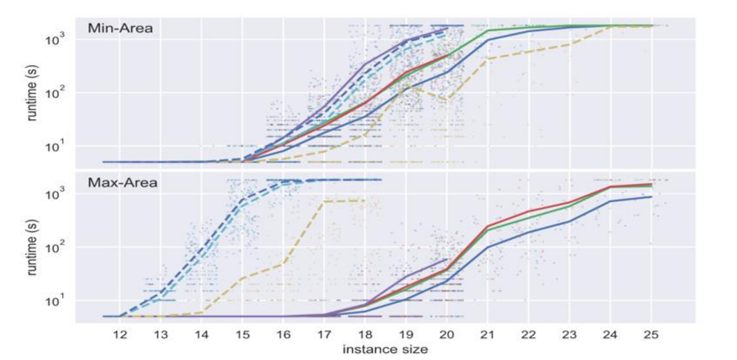
Keller[1], Christian Rieck[1], Scheffer[2], and Arne Schmidt[1]

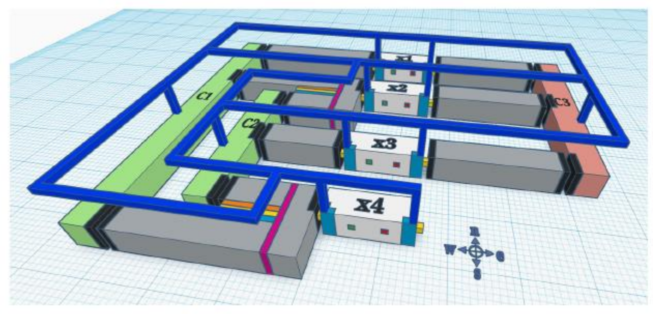
[1] Department of Computer Science, TU Braunschweig, Germany. {jkeller, rieck, aschmidt}@ibr.cs.tu-bs.de

Department of Computer Science, University of Münster, Germany.

Computing Area-Optimal Simple Polygonizations

SÁNDOR P. FEKETE, ANDREAS HAAS, PHILLIP KELDENICH, MICHAEL PERK, and ARNE SCHMIDT, Department of Computer Science, TU Braunschweig





Organisation

Arne Schmidt

Vorlesung + Gr. Übung + Kl. Übung

- Grundlagen
- Vertiefungen
- Besprechung Hausaufgaben

Fragen

Inhaltlich oder allgemeiner Ablauf	Vorlesung / große Übung
Zu Übungsblättern oder Korrektur	Kleine Übung
Individuelle Fragen	**Immer per Mail** an aschmidt@ibr.cs.tu-bs.de (Nicht über StudIP)
Sprechstunde	Montags, 09:45 Uhr im Raum IZ 333 (Am besten vorher per Mail ankündigen)

Semesterplan (vorläufig)

Datum (VL)	VL Nummer	VL / Ü Inhalt	Hausaufgabe (Ausgabe)	Hausaufgabe (Besprechung)
15. April	0	Einleitung		
22. April	1	Range Queries	HA1	
29. April	2 + Ü	KD-Trees		
06. Mai	3 + Ü	Sweep Line Algorithmen	HA2	HA1
13. Mai	4 + Ü	Overlays		
20. Mai	5 + Ü	Punktlokalisierung	HA3	HA2
27. Mai	Exkursionswoche			
03. Juni	6 + Ü	Trapezoidal Maps		
10. Juni	7 + Ü	Minkowski Summen	HA4	HA3
17. Juni	8 + Ü	PSPACE		
24. Juni	9 + Ü	Art Gallery	HA5	HA4
01. Juli	10 + Ü	Sichtbarkeitspolygone		
08. Juli	11 + Ü	Zusammenfassung		HA5
15. Juli	12 + Ü	Fragestunde / Puffer		

Hausaufgaben

- 5 Blätter
- Votierungssystem
- 50% der Aufgaben müssen votiert sein.
- Anmeldung per Studip (VL/Ü)

Klausur – Vorläufig

Datum:
28.08.

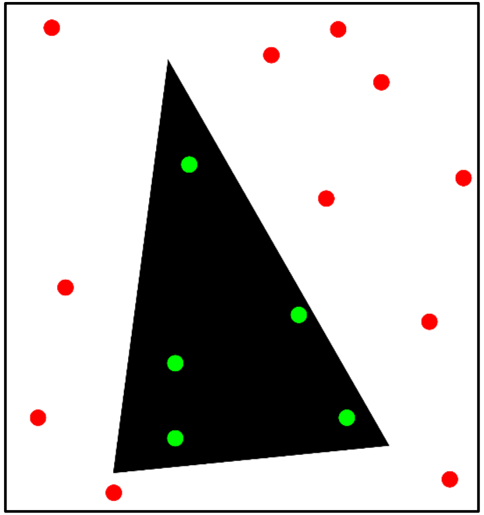
Uhrzeit:
09 Uhr – 11 Uhr

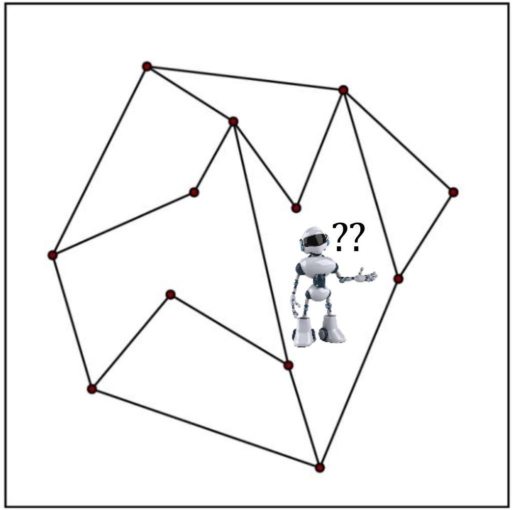
Ort:
SN 19.1

Inhalt:
Hauptsächlich Vorlesung

Kapitel 1 - Einführung

Probleme





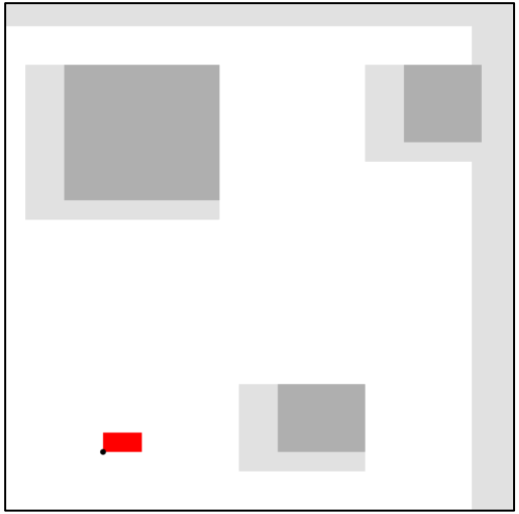
Welche Punkte liegen in dem Bereich?

→ Range Query

In welchem Bereich liegt ein Punkt?

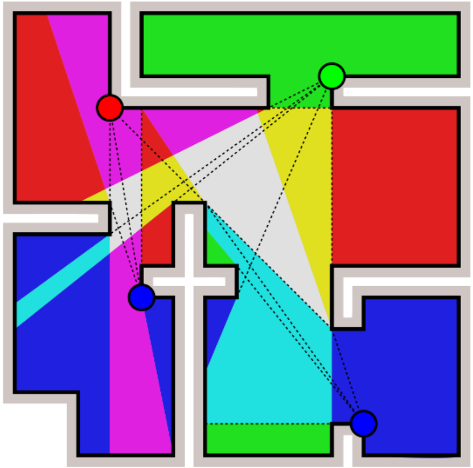
→ Punktlokalisierung

Probleme



Wohin und wie kann das rote Objekt bewegt werden?

→ Robot Motion Planning



Wie viele Wächter werden zur Abdeckung benötigt?

→ Art Gallery Problem

Kapitel 1.1 – Primitive

Primitive

Einige Operatoren werden immer wieder benötigt.

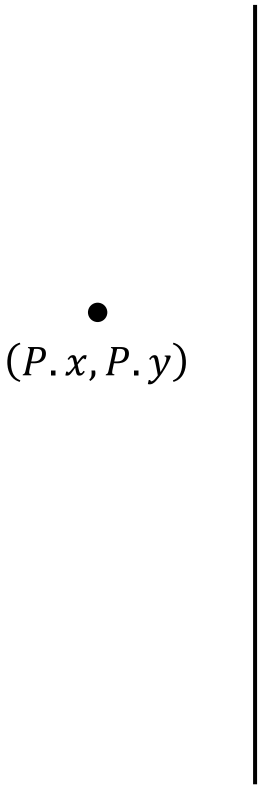
Wie können wir testen, ob ein Punkt „links“ von einer Linie liegt?

Wie können wir testen, ob zwei Segmente sich schneiden?

Wie testen wir, ob ein Punkt innerhalb einer Fläche liegt?

Kapitel 1.1.1 – Punkte und Linien

Punkt-Gerade-Test



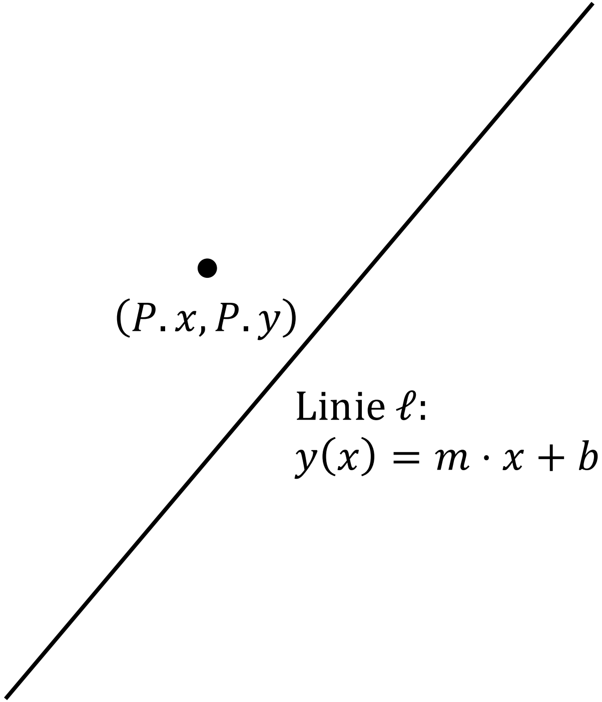
Linie ℓ:
$\ell.x$ = const

Wie testet man, ob der Punkt P links/auf/recht von ℓ liegt?

- $P.x < \ell.x \Rightarrow P$ liegt links von ℓ
- $P.x = \ell.x \Rightarrow P$ liegt auf ℓ
- $P.x > \ell.x \Rightarrow P$ liegt rechts von ℓ

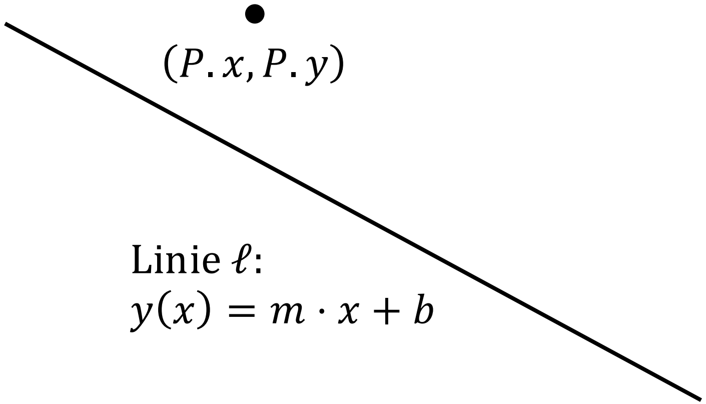
Punkt-Gerade-Test

Wie testet man, ob der Punkt P links/auf/recht von ℓ liegt?



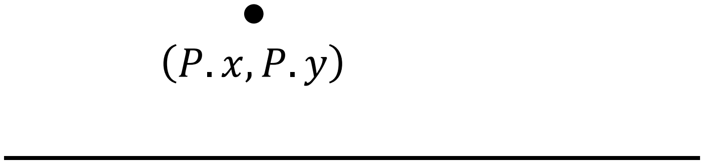
Punkt-Gerade-Test

Wie testet man, ob der Punkt P links/auf/recht von ℓ liegt?



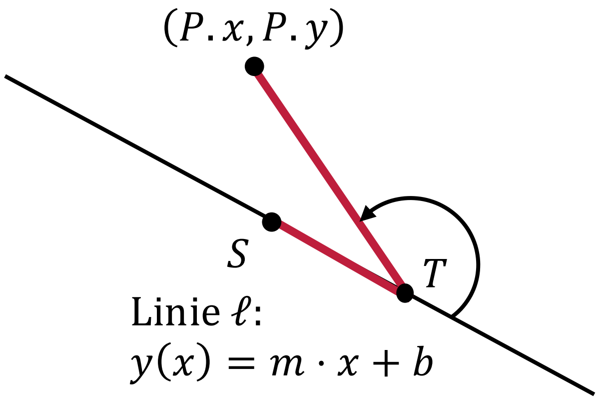
Punkt-Gerade-Test

Wie testet man, ob der Punkt P links/auf/recht von ℓ liegt?



Linie ℓ:
$y(x) = m \cdot x + b$

Punkt-Gerade-Test



Wie testet man, ob der Punkt P links/auf/rechts von ℓ liegt?

Definition 1.1:
Ein *Punkt* P ist ein Element von $\mathbb{R}^2$.
Ein *Segment* (Strecke) ST ist die Linie von einem Punkt S zu einem Punkt T.

Zwei Segmente ST und TP (kurz STP) bilden einen *links-Knick*, wenn der gebildete Winkel gegen den Uhrzeigersinn verläuft. Ansonsten bilden sie einen *rechts-Knick*.

Seien S und T zwei Punkte auf einer Linie ℓ. Dann gilt:
- STP bildet einen links-Knick, dann liegt P links von ℓ.
- STP bildet einen rechts-Knick, dann liegt P rechts von ℓ.
- S, T, P sind *kolinear*, wenn P auf ℓ liegt.

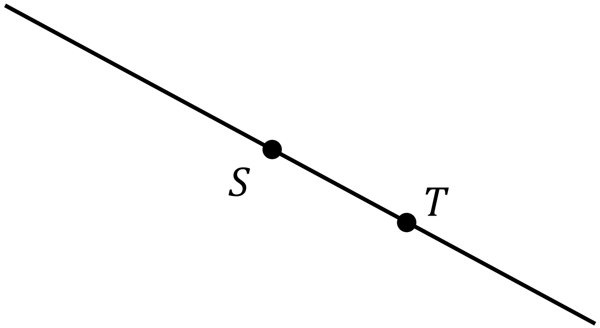
Punkt-Segment-Test

Das heißt, wir können uns auf Segmente und Punkte konzentrieren.

Betrachten wir den einfachen Fall: P liegt auf der Geraden durch S und T.

Wie beschreibt man die Gerade durch S und T?

Gerade durch zwei Punkte



Definition 1.2:
Jeder Punkt auf der Geraden durch S und T lässt sich beschreiben durch:

$$P(\lambda) = S + \lambda(T - S)$$

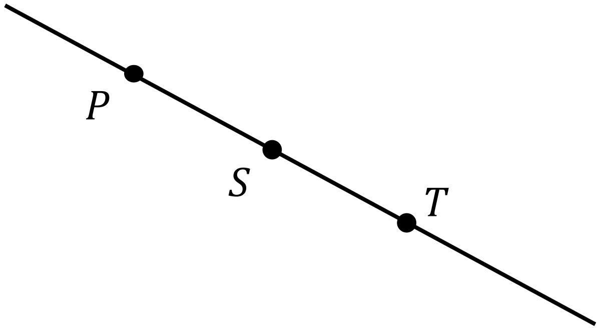
Bzw.

$$P(\lambda) = (1 - \lambda)S + \lambda T$$

Das ist eine ***Linearkombination***.
Ist $\lambda \in [0,1]$, spricht man von einer ***Konvexkombination***.

Lemma 1.3:
Ein Punkt P liegt auf der Geraden durch S und T, wenn ein $\lambda \in \mathbb{R}$ mit $P = (1 - \lambda)S + \lambda T$ existiert.

Gerade durch zwei Punkte



Lemma 1.3:
Ein Punkt P liegt auf der Geraden durch S und T, wenn ein $\lambda \in \mathbb{R}$ mit $P = (1-\lambda)S + \lambda T$ existiert.

Damit erhalten wir für einen Punkt P folgende Gleichungen:

$$P.x = (1-\lambda)S.x + \lambda T.x$$
$$P.y = (1-\lambda)S.y + \lambda T.y$$

Umgestellt nach λ:

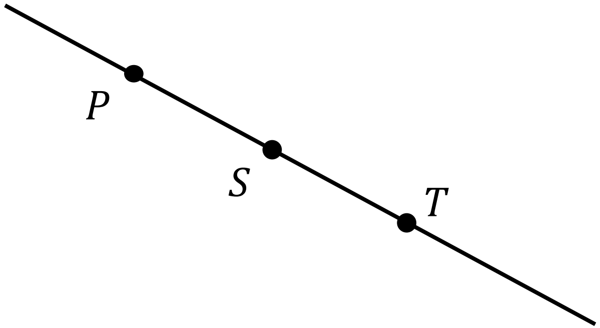
$$\lambda = \frac{P.x - S.x}{T.x - S.x}$$

$$\lambda = \frac{P.y - S.y}{T.y - S.y}$$

Also muss gelten:

$$(P.y - S.y)(T.x - S.x) - (P.x - S.x)(T.y - S.y) = 0$$

Gerade durch zwei Punkte



Lemma 1.3:
Ein Punkt P liegt auf der Geraden durch S und T, wenn ein $\lambda \in \mathbb{R}$ mit $P = (1-\lambda)S + \lambda T$ existiert.

Also muss gelten:

$$(P.y - S.y)(T.x - S.x) - (P.x - S.x)(T.y - S.y) = 0$$

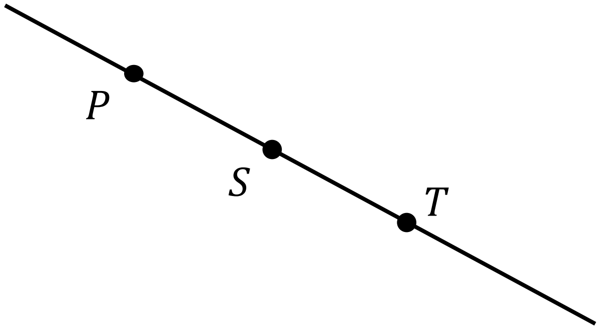
Das ist eine Determinante!

$$\det\begin{pmatrix} T.x - S.x & T.y - S.y \\ P.x - S.x & P.y - S.y \end{pmatrix}$$

Etwas anders dargestellt:

$$\det\begin{pmatrix} 1 & S.x & S.y \\ 0 & T.x - S.x & T.y - S.y \\ 0 & P.x - S.x & P.y - S.y \end{pmatrix}$$

Gerade durch zwei Punkte



Lemma 1.3:
Ein Punkt P liegt auf der Geraden durch S und T, wenn ein $\lambda \in \mathbb{R}$ mit $P = (1-\lambda)S + \lambda T$ existiert.

Lineare Algebra liefert:

$$\det\begin{pmatrix} 1 & S.x & S.y \\ 0 & T.x - S.x & T.y - S.y \\ 0 & P.x - S.x & P.y - S.y \end{pmatrix} = \det\begin{pmatrix} 1 & S.x & S.y \\ 1 & T.x & T.y \\ 1 & P.x & P.y \end{pmatrix}$$

Korollar 1.4:
Ein Punkt P liegt auf der Geraden durch S und T, wenn

$$\det\begin{pmatrix} 1 & S.x & S.y \\ 1 & T.x & T.y \\ 1 & P.x & P.y \end{pmatrix} = 0.$$

Beispiele

$P_1 = (-1, 2)$

$P_2 = (2, 2)$

$T = (1,1)$

$P_3 = (3, 1)$

$S = (0,0)$

$$\det\begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & -1 & 2 \end{pmatrix} = 2 + 1 = 3$$

$$\det\begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 2 & 2 \end{pmatrix} = 2 - 2 = 0$$

$$\det\begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 3 & 1 \end{pmatrix} = 1 - 3 = -2$$

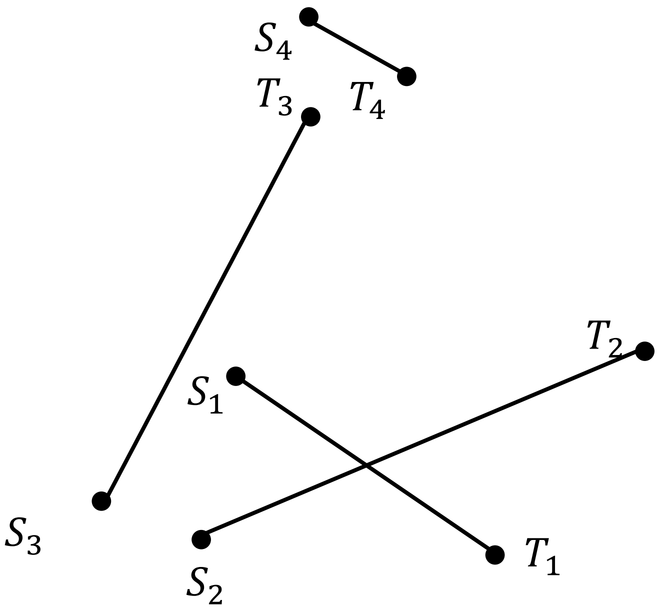
Punkte und Segmente

Definition 1.5:
Seien $P, S, T \in \mathbb{R}^2$. Wir definieren
$$ccw(S,T,P) := \det\begin{pmatrix} 1 & S.x & S.y \\ 1 & T.x & T.y \\ 1 & P.x & P.y \end{pmatrix}$$

Theorem 1.6:
Seien $P, S, T \in \mathbb{R}^2$. Dann ist STP
- ein links-Knick, wenn $ccw(S,T,P) > 0$
- ein rechts-Knick, wenn $ccw(S,T,P) < 0$
- kolinear, wenn $ccw(S,T,P) = 0$

Kapitel 1.1.2 – Schneiden von Segmenten

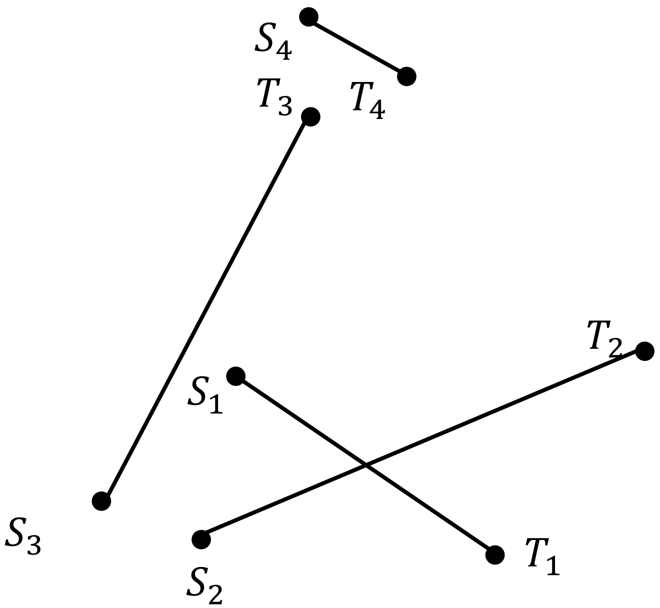
Segmente



Beobachtung 1:
Wenn sich zwei Segmente schneiden, dann müssen die Endpunkte der jeweils anderen Linie auf unterschiedlichen Seiten liegen!

Beobachtung 2:
Es reicht nicht, nur eine Linie zu prüfen!

Segmente



Theorem 1.7:
Zwei Segmente S_1T_1 und S_2T_2 schneiden sich genau dann, wenn:

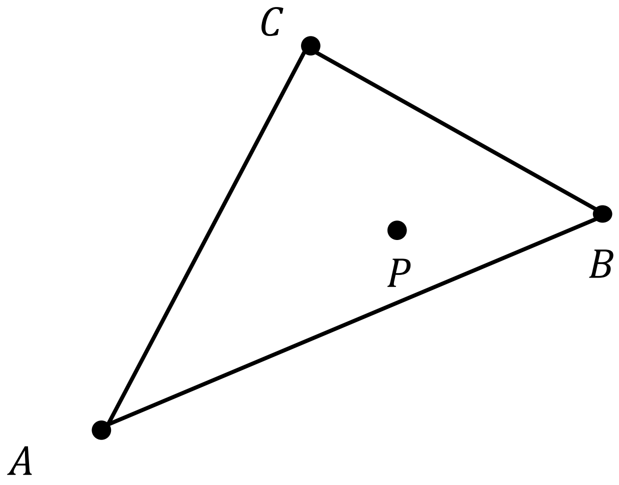
1. $ccw(S_1, T_1, S_2) \cdot ccw(S_1, T_1, T_2) \leq 0$ und
2. $ccw(S_2, T_2, S_1) \cdot ccw(S_2, T_2, T_1) \leq 0$

Etwas stärkere Bedingung fordert strikt kleiner 0, falls kolineare Punkte auftreten.

Auch zu berücksichtigen:
Degenerierte Segmente! ($S = T$)

Kapitel 1.1.3 – Container-Test

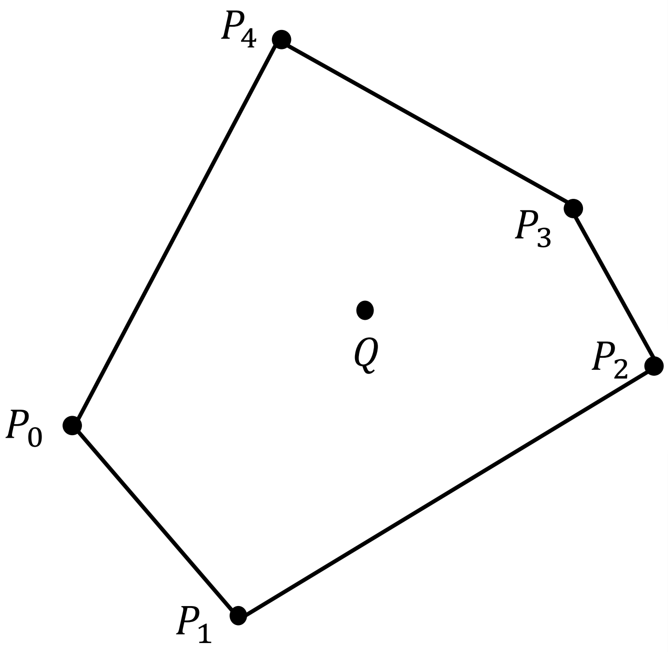
Dreiecke



Lemma 1.8:
Sei ABC ein Dreieck und P ein Punkt. P liegt genau dann innerhalb von ABC, wenn

1. $ccw(A, B, P) \geq 0$,
2. $ccw(B, C, P) \geq 0$ und
3. $ccw(C, A, P) \geq 0$

Konvexe Polygone



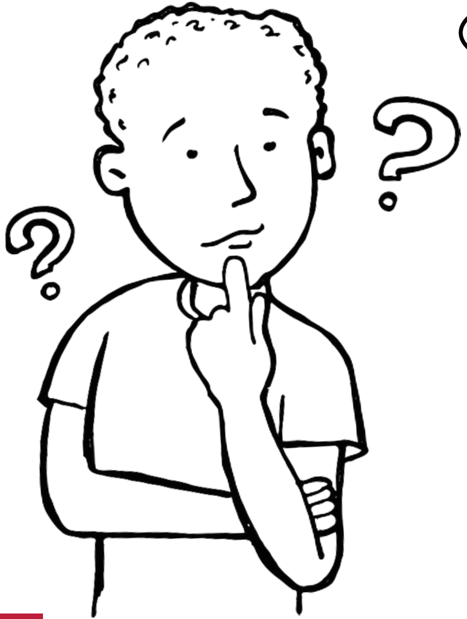
Definition 1.9 (Polygone):
Sei P $:= (P_0, \dots, P_{n-1})$ eine Folge von n Punkten. Dann heißt P *einfaches Polygon*, wenn für jedes Paar P_i, P_j mit $j \notin \{i-1,\ i,\ i+1\}$ die Segmente P_iP_{i+1} und P_jP_{j+1} sich nicht schneiden.

P heißt *konvexes* Polygon, wenn jedes Tripel P_i, P_{i+1}, P_{i+2} ein links-Knick bildet (jeder Innenwinkel ist als maximal 180°).

Achtung: Die Indizes sind immer mod n zu verstehen.

Lemma 1.10:
Sei $P := (P_0, \dots, P_{n-1})$ ein konvexes Polygon und Q ein Punkt. Q liegt genau dann innerhalb von P, wenn für alle Paare P_i, P_{i+1} gilt: $ccw(P_i, P_{i+1}, Q) \geq 0$

Nächste Kapitel



Wie funktioniert das bei beliebigen Formen?

Was passiert, wenn ich mehrere Punkte zu testen habe?

Was kann man tun, wenn ganz viele Flächen zur Verfügung stehen?