

# Grafisch-geometrische Algorithmen

Jan Keuchel

10. November 2025

# Inhaltsverzeichnis

<b>1</b>	<b>Grundlagen</b>	<b>2</b>
1.1	Orientierung dreier Punkte . . . . .	2
1.2	Schnitt zweier Segmente . . . . .	3
<b>2</b>	<b>Konvexe Hülle</b>	<b>5</b>
2.1	Definition . . . . .	5
2.2	Algorithmen . . . . .	6
2.2.1	Brute Force . . . . .	6
2.2.2	Monotone Ketten . . . . .	6
2.2.3	Jarvis' March . . . . .	6
2.3	Beweise . . . . .	6
<b>3</b>	<b>Schnitt von Linien-Segmenten</b>	<b>8</b>
<b>4</b>	<b>Mengenoperationen</b>	<b>9</b>
<b>5</b>	<b>Triangulation</b>	<b>10</b>

# Kapitel 1

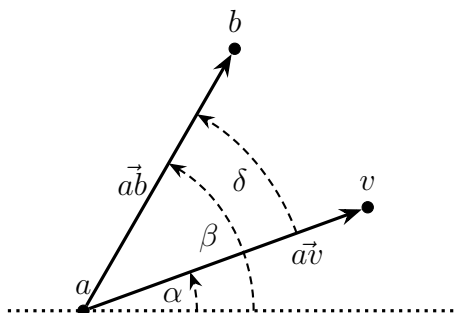
## Grundlagen

### 1.1 Orientierung dreier Punkte

Eine grundlegende und oft genutzte Operation, ist zu prüfen, ob sich ein Punkt rechts, links oder auf einer Strecke befindet. Ziel ist es also eine Funktion `RIGHTOF` zu definieren, welche 3 Punkte der Form  $p = (p_x, p_y) \in \mathbb{R}^2$  nimmt und diese auf  $\mathbb{R}$  abbildet. Also  $\text{RIGHTOF} : \mathbb{R}^6 \rightarrow \mathbb{R}$ , mit

$$\text{RIGHTOF}(a, b, v) \begin{cases} < 0 & \text{falls } v \text{ links der gerichteten Gerade } \overrightarrow{ab} \text{ liegt.} \\ = 0 & \text{falls } v \text{ auf der gerichteten Gerade } \overrightarrow{ab} \text{ liegt.} \\ > 0 & \text{falls } v \text{ rechts der gerichteten Gerade } \overrightarrow{ab} \text{ liegt.} \end{cases}$$

. Gegeben drei Punkte  $a, b, v \in \mathbb{R}^2$  hat man folgende Situation:



Man stellt fest, dass  $v$  genau dann rechts der gerichteten Gerade  $\overrightarrow{ab}$  liegt, wenn  $\sin \varphi > 0$ , links davon, wenn  $\sin \varphi < 0$  und darauf liegt, falls  $\sin \varphi = 0$  gilt. Somit kann man

$$\text{RIGHTOF}(a, b, v) := \sin \varphi = \sin(\beta - \alpha) = \sin \beta \cos \alpha - \cos \beta \sin \alpha$$

definieren. Da trigonometrische Funktionen aufgrund von Rechenleistung unerwünscht sind, kann `RIGHTOF` mit

$$\sin \beta = b.y - a.y$$

$$\cos \alpha = v.x - a.x$$

$$\cos \beta = b.x - a.x$$

$$\sin \alpha = v.y - a.y$$

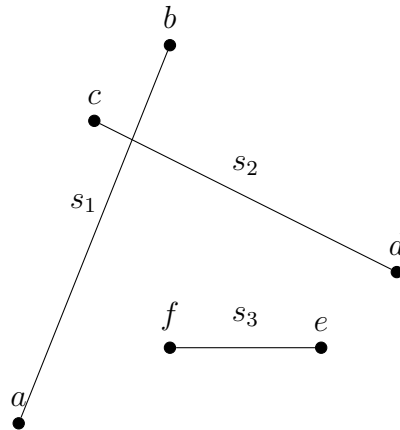
wie folgt dargestellt werden:

$$\text{RIGHTOF}(a, b, v) = (b.y - a.y)(v.x - a.x) - (b.x - a.x)(v.y - a.y)$$

.

## 1.2 Schnitt zweier Segmente

Eine weitere, oft genutzte Funktion ist der Test, ob sich zwei Segmente schneiden oder nicht. Ziel ist es nicht, zu berechnen, wo ein Schnittpunkt liegt, sondern ob er existiert oder nicht. Wir betrachten also mehrere Segmente  $s_1, s_2, s_3$  der Form  $s_i = (p_{i1}, p_{i2})$ , mit  $p \in \mathbb{R}^2$ :



Dabei stellt man fest, dass sich zwei Segmente  $s_i, s_j$  in genau einem Fall schneiden: Wenn sowohl die Punkte  $s_{j1}, s_{j2}$  auf unterschiedlichen Seiten der Geraden  $\overline{s_{i1}s_{i2}}$ , und die Punkte  $s_{i1}, s_{i2}$  auf unterschiedlichen Seiten der Geraden  $\overline{s_{j1}s_{j2}}$  liegen. Wird der Fall, dass ein Punkt  $p$  auf einer Geraden  $\overline{ab}$  liegt,

ebenfalls als Schnittpunkt angesehen, so reicht es aus, dass die beiden Punkt nicht auf der gleichen Seite liegen.

Die Funktion `CROSSINGEXISTS` kann also mit Hilfe der vorig beschriebenen Funktion `RIGHTOF` definiert werden (Hier wird ein Punkt, der auf einer Geraden liegt, bereits als Schnittpunkt angesehen):

$$\text{CROSSINGEXISTS}(a, b, c, d) = \left( \left( \text{RIGHTOF}(a, b, c) \cdot \text{RIGHTOF}(a, b, d) \right) \leq 0 \right) \textbf{ and } \left( \left( \text{RIGHTOF}(c, d, a) \cdot \text{RIGHTOF}(c, d, b) \right) \leq 0 \right)$$

# Kapitel 2

## Konvexe Hülle

Eine Untermenge  $C$  einer Ebene heißt genau dann konvex, wenn  $\forall p, q \in C : \overline{pq} \subseteq C$ .

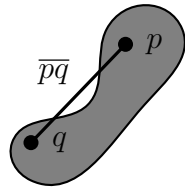


Abbildung 2.1: Nicht konvex

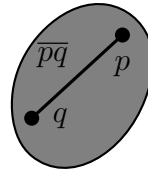


Abbildung 2.2: Konvex

Ein Polygon  $\mathcal{P}$  ist eine Fläche, die durch eine Folge an  $k > 2$  Punkten  $\langle p_1, p_2, \dots, p_k \rangle$  bestimmt wird.

### 2.1 Definition

Zu einer endlichen Menge  $P$  aus  $n$  Punkten, ist die konvexe Hülle  $\mathcal{CH}(P)$  ein konvexes Polygon, welches aus den Punkten von  $P$  besteht und alle Punkte von  $P$  einschließt.  $\mathcal{CH}(P)$  wird als eine von einem zufälligen Punkt ausgehende, im Uhrzeigersinn sortierte Liste aller Punkte  $p_i \in \mathcal{CH}(P)$  dargestellt.

Bei Abbildung 2.3 ist  $P = \{p_0, p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9\}$  und  $\mathcal{CH}(P) = \langle p_0, p_1, p_3, p_7, p_6, p_2 \rangle$

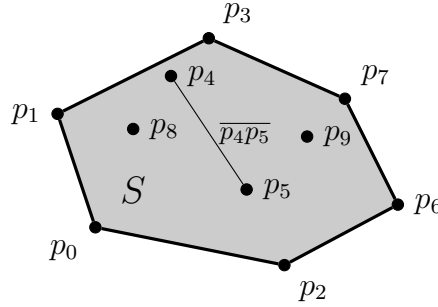


Abbildung 2.3: Konvexe Hülle

## 2.2 Algorithmen

Für die Berechnung der Konvexen Hülle werden nun drei Algorithmen betrachtet.

### 2.2.1 Brute Force

Bei der Betrachtung der Kanten  $e_i$  von  $\mathcal{CH}(P)$  fällt auf, dass diese eine gemeinsame Eigenschaft haben: All anderen Punkte  $p \in P$  befinden sich – da  $\mathcal{CH}(P)$  im Uhrzeigersinn sortiert ist – auf der rechten Seite von  $e_i$ . Die Kanten  $E$  von  $\mathcal{CH}(P)$  können also auch wie folgt beschrieben werden:

$$E = \{(p, q) \in P \times P : p \neq q \mid \forall r \in P : r \neq p \wedge r \neq q \wedge \text{RIGHTOF}(p, q, r) \geq 0\}$$

. Aus  $E$  müsste schließlich noch eine sortierte Folge gebildet werden.

### 2.2.2 Monotone Ketten

### 2.2.3 Jarvis' March

## 2.3 Beweise

**Theorem 1.** *Der Schnitt zweier konvexer Polygone ist ein konvexes Polygon.*

*Beweis.* Ein Polygon  $\mathcal{P}$  ist konvex, g.d.w.  $\forall p, q \in \mathcal{P} : \overline{pq} \subseteq \mathcal{P}$ . Seien  $\mathcal{A}, \mathcal{B}$  konvexe Polygone und  $\mathcal{J} = \mathcal{A} \cap \mathcal{B}$ . Mit  $(s, t \in \mathcal{J}) \Leftrightarrow (s, t \in \mathcal{A} \wedge s, t \in \mathcal{B})$  gilt  $(\overline{st} \subseteq \mathcal{A} \wedge \overline{st} \subseteq \mathcal{B}) \Leftrightarrow (\overline{st} \subseteq \mathcal{J})$ .  $\square$

**Theorem 2.** *Unter allen Polygonen, welche eine Punktmenge  $P$  enthalten, ist dasjenige mit dem kleinsten Umfang konvex.*

*Beweis.* Sei  $\mathcal{P} = \langle p_0, p_1, \dots, p_{n-1} \rangle$  das Polygon mit dem kleinsten Umfang, welches  $P$  enthält. Ferner sei  $p_i^+ = p_{(i+1) \bmod n}$  der Nachfolger von  $p_i$  und  $U(\mathcal{A})$  der Umfang eines Polygons  $\mathcal{A}$ . Für drei Punkte  $A, B, C$  ist  $\overline{AC} \leq \overline{ABC}$ . Somit gilt für  $a_1, \dots, a_n$ , dass  $\overline{a_1 a_n} \leq \overline{a_1 \dots a_n}$ . Sei  $\mathcal{P}$  nicht konvex, dann

$$\exists p, q \in \mathcal{P} : \overline{pq} \not\subseteq \mathcal{P} \Rightarrow \exists j, k < n \wedge j < k : \exists s, t \in \left( \overline{p_j p_j^+} \cup \overline{p_k p_k^+} \right) \cap \overline{pq}$$

. Mit  $\mathcal{P}' = \langle p_1, \dots, p_j, s, t, p_k^+, \dots, p_n \rangle$  ist  $U(\mathcal{P}') < U(\mathcal{P})$ .

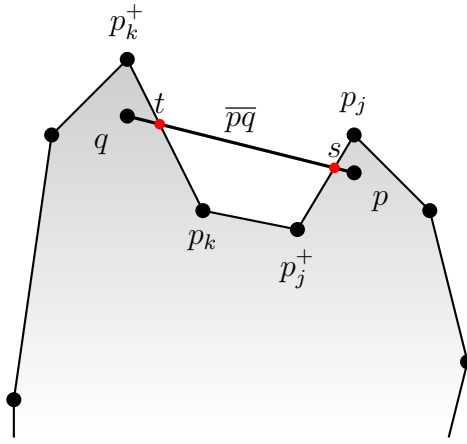


Abbildung 2.4: Ist  $\mathcal{P}$  nicht konvex, gibt es ein Segment, zwischen zwei Punkten  $p, q$ , welches nicht in  $\mathcal{P}$  liegt.

□



## Kapitel 3

# Schnitt von Linien-Segmenten

# Kapitel 4

## Mengenoperationen

# Kapitel 5

## Triangulation