Algorithmische Geometrie

Vorlesung im Sommersemester 2005
Prof. Dr. Helmut Alt
Institut für Informatik der FU Berlin

Skript zum 15. April

1 Konvexe Hüllen

1.1 Grundlegende Definitonen

Es seien $p, q \in \mathbb{R}^n$ Punkte im n-dimensionalen Raum. Eine **Strecke** $\overline{pq}$ zwischen p und q kann mit einer Punktmenge identifiziert werden und ist definiert als:

$$\overline{pq} = \{(1-\lambda)p + \lambda q \mid \lambda \in [0,1]\}$$

Eine **Gerade** g durch p und q kann analog zur Strecke dargestellt werden durch:

$$g = \{(1-\lambda)p + \lambda q \mid \lambda \in \mathbb{R}\}$$

Oft wird eine Gerade als **orientiert** oder gerichtet betrachtet. Eine Gerade durch p in Richtung q ist dann verschieden von einer Geraden durch q in Richtung p.

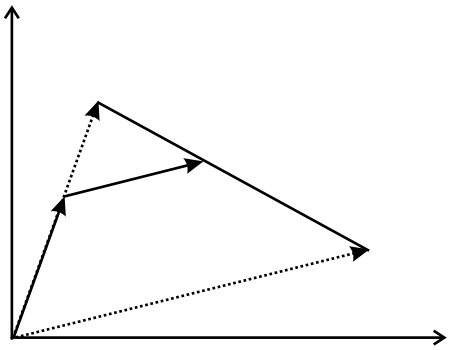

Abbildung 1: Zur Definition der Strecke durch skalierte Ortsvektoren.

In der Ebene können wir eine Gerade g durch die Punkte p, q auch mit Hilfe ihres Normalenvektors darstellen. Wir notieren dazu im folgenden mit $\vec{p}$ den Ortsvektor eines Punktes p und definieren $\vec{d} = \vec{q} - \vec{p} = (d_x\ d_y)^T$. Der Vektor $\vec{n} = (-d_y\ d_x)^T$ wird der **Normalenvektor** der Geraden gennant. Er steht senkrecht auf der Geraden g. Es gilt:

$$\vec{x} \in \mathbb{R}^2 \text{ liegt auf } g \quad \Leftrightarrow \quad \vec{p} - \vec{x} \perp \vec{n} \quad \Leftrightarrow \quad (\vec{p} - \vec{x}) \cdot \vec{n} = 0$$

Betrachten wir g als gerichtete Gerade, so können wir die Mengen aller Punkte, die links bzw. rechts von der Geraden liegen, wenn wir in ihre Richtung blicken, definieren als:

$$\text{Punkte links von } g: \quad \{\vec{x} \mid (\vec{p} - \vec{x}) \cdot \vec{n} > 0\}$$
$$\text{Punkte rechts von } g: \quad \{\vec{x} \mid (\vec{p} - \vec{x}) \cdot \vec{n} < 0\}$$

Diese Mengen sind *offene Halbebenen*. Sie heißen offen, weil sie ihre „Grenze" g nicht enthalten. Dementsprechend können wir die geschlossenen Halbebenen der Punkte links bzw. rechts der Geraden g beschreiben, indem wir in der obigen Definition ein Skalarprodukt ≥ 0 bzw. ≤ 0 fordern.

1.1.1 Algorithmische Aspekte

Wir betrachten im folgenden Geraden und Strecken, die durch je zwei Punkte definiert sind. Wir können in konstanter Zeit folgende Fragen entscheiden:

- Haben zwei gegebene Strecken bzw. Geraden einen Schnittpunkt? Wenn ja, welchen?
  Im Falle von zwei Strecken $\overline{p_1q_1}$ und $\overline{p_2q_2}$ gilt z.B: $\overline{p_1q_1}$ schneidet $\overline{p_2q_2}$ genau dann, wenn p_1 links und q_1 rechts von der Geraden durch p_2 und q_2 liegt und umgekehrt.
- Gegeben seien drei orientierte Geraden g, g_1, g_2 so dass g_1 und g_2 jeweils g schneiden. Schneidet g_1 die Gerade g früher (im Sinne der Orientierung von g) als g_2? Es wird sich im folgenden als nützlich erweisen, hier eine Relation zu definieren: $g_1 <_g g_2 \Leftrightarrow g_1$ schneidet g früher als g_2

Damit können wir auch folgendes Problem lösen:

Schnittpunktsortierung Gegeben sei eine orientierte Gerade g sowie eine Folge von Geraden $g_1, g_2, ..., g_n$, die g schneiden. In welcher Reihenfolge schneiden die g_i die Gerade g?

Zur Lösung können wir einen beliebigen Sortieralgorithmus verwenden, wenn als Vergleichsoperation die oben definierte Relation $<_g$ verwenden. Damit kann die Schnittpunktsortierung mit einer Laufzeit von $O(n \log n)$ bewältigt werden. Auch den Binärsuche-Algorithmus können wir für Geradenschnittpunkte einsetzen, indem wir als Vergleichoperation $<_g$ verwenden.

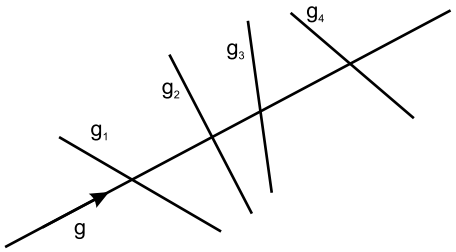

Abbildung 2: Schnittpunktsortierung

1.1.2 Polygone

Eine endliche Folge von Strecken der Form $\overline{p_1p_2}$, $\overline{p_2p_3}$, ..., $\overline{p_np_{n+1}}$ heißt **Streckenzug** (auch **Kantenzug** oder **Polygonzug**) der Länge n. Die p_i werden die Ecken genannt, die Strecken $\overline{p_ip_{i+1}}$ sind die Kanten des Polygonzugs.

Ein Polygonzug heißt **einfach** genau dann, wenn für $i < j$ gilt:

$$\overline{p_ip_{i+1}} \cap \overline{p_jp_{j+1}} = \begin{cases} p_j & \text{für } j = i+1 \\ \varnothing & \text{sonst} \end{cases}$$

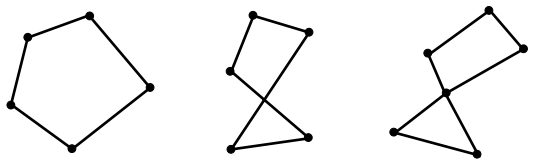

Abbildung 3: Einfaches Polygon (links), nicht einfache Polygone (mitte, rechts).

Ein Polygonzug heißt **Polygon** genau dann, wenn $p_{n+1} = p_1$. Ein Polygon ist **einfach**, genau dann wenn gilt:

$$\overline{p_i p_{i+1}} \cap \overline{p_j p_{j+1}} = \begin{cases} p_j & \text{für } j = i+1 \\ p_j & \text{für } i = 1 \ \wedge \ j = n \\ \varnothing & \text{sonst} \end{cases}$$