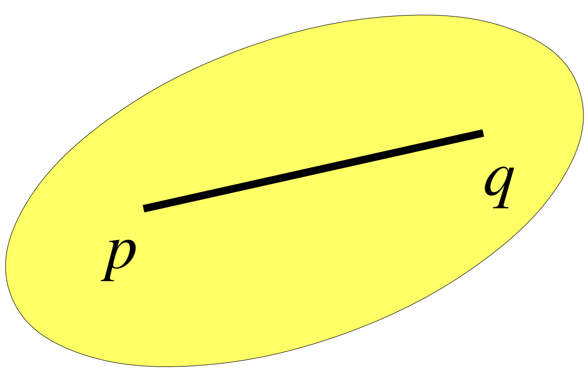
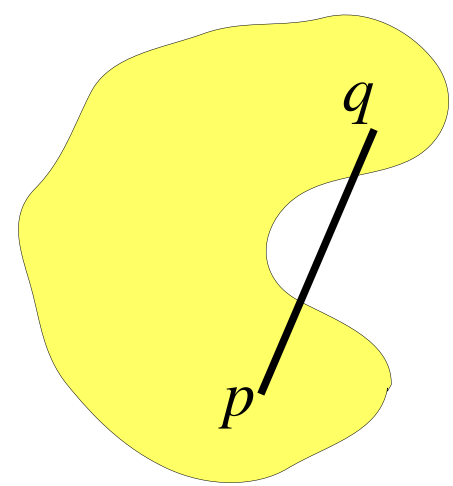
Berechnung der konvexen Hülle

1. Grundbegriffe

Eine Menge M von Punkten heißt **konvex** wenn für alle Punkte p und q aus M auch die Strecke mit Endpunkten p und q in M enthalten ist.



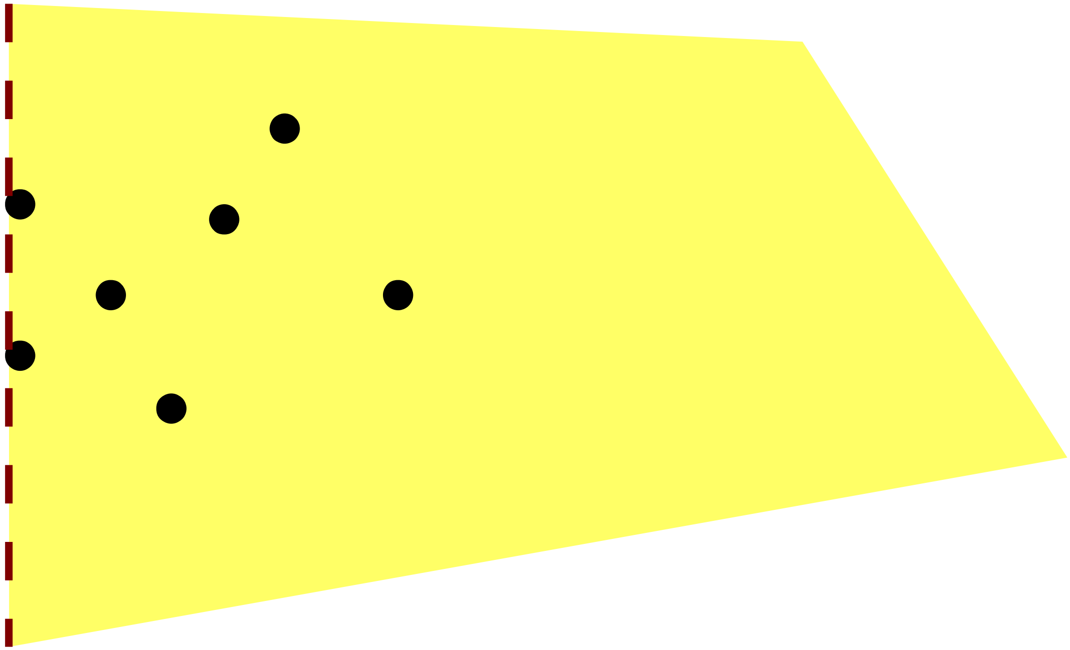
konvexe Menge



nicht konvexe Menge

Für eine Menge P von Punkten ist die **konvexe Hülle** von P gleich dem Durchschnitt aller konvexen Mengen, die P enthalten.

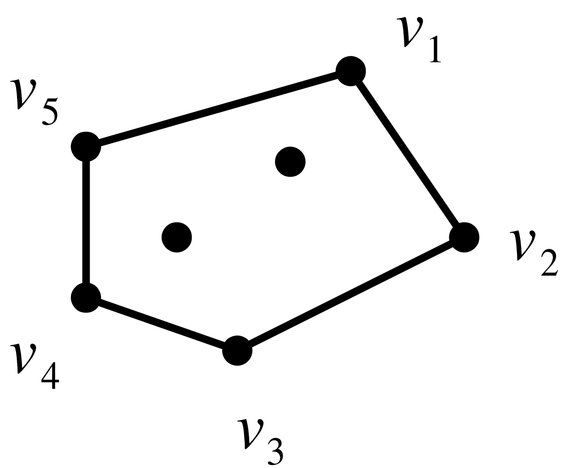
Sie wird mit **CH(*P*)** bezeichnet.



Bei uns wird P immer eine endliche nicht leere Menge mit n Punkten sein.

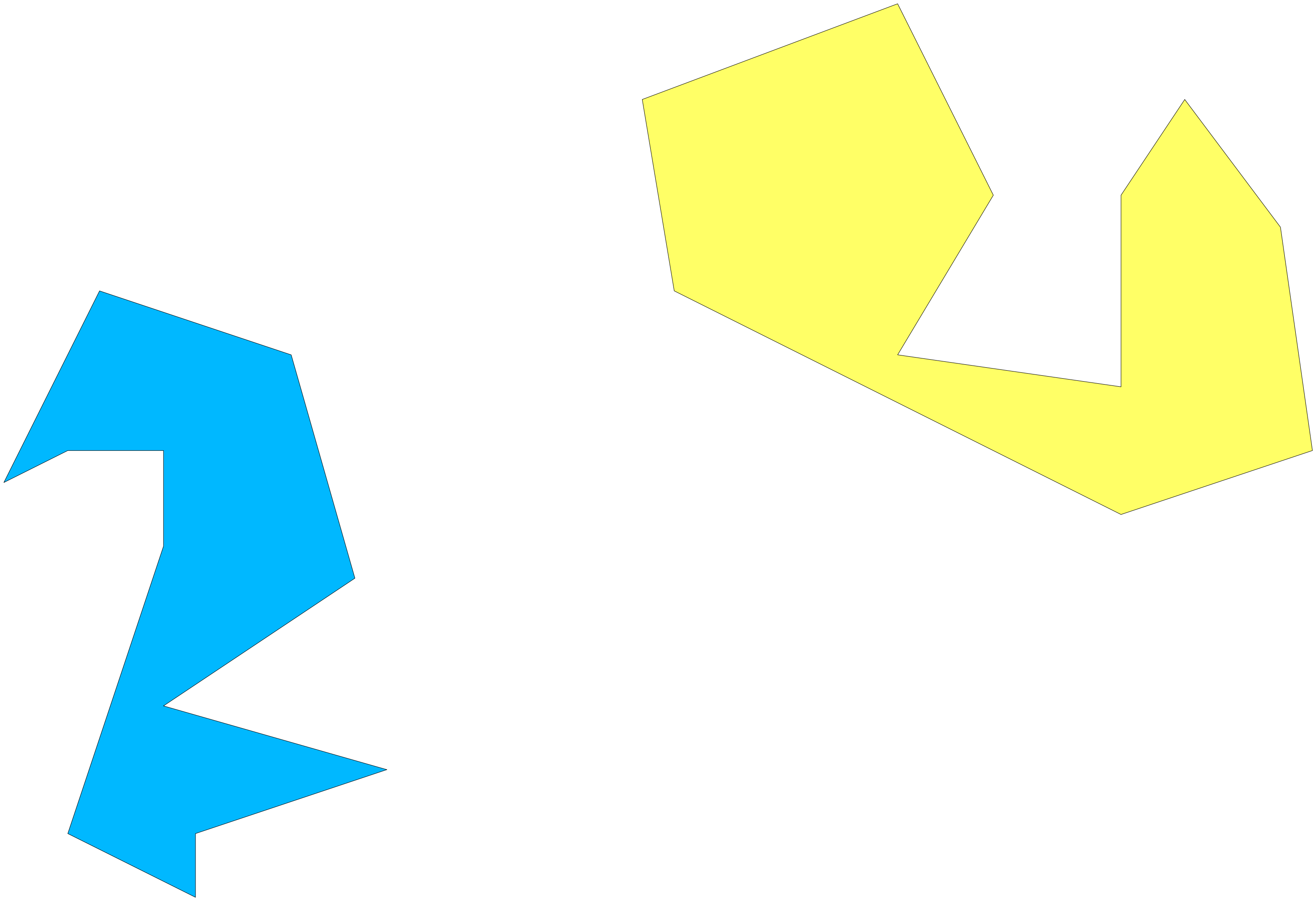
Wir wollen eine **endliche Beschreibung** der Menge CH(P) berechnen.

Da CH(P) ein konvexes Polygon mit höchstens n Ecken ist, liegt es nahe, die **Folge der Eckpunkte** dieses Polygons zu berechnen.

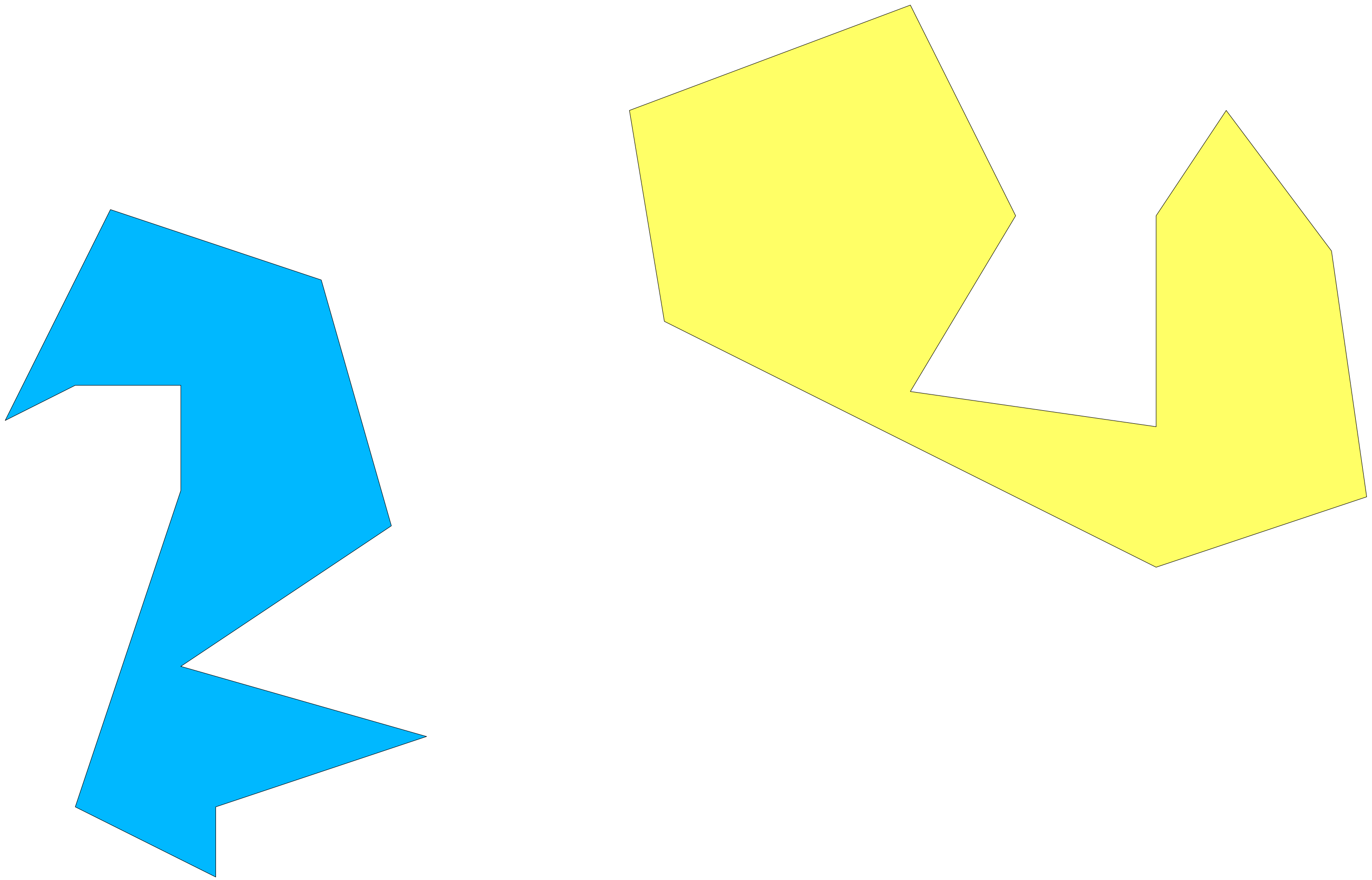


2. Eine Anwendung

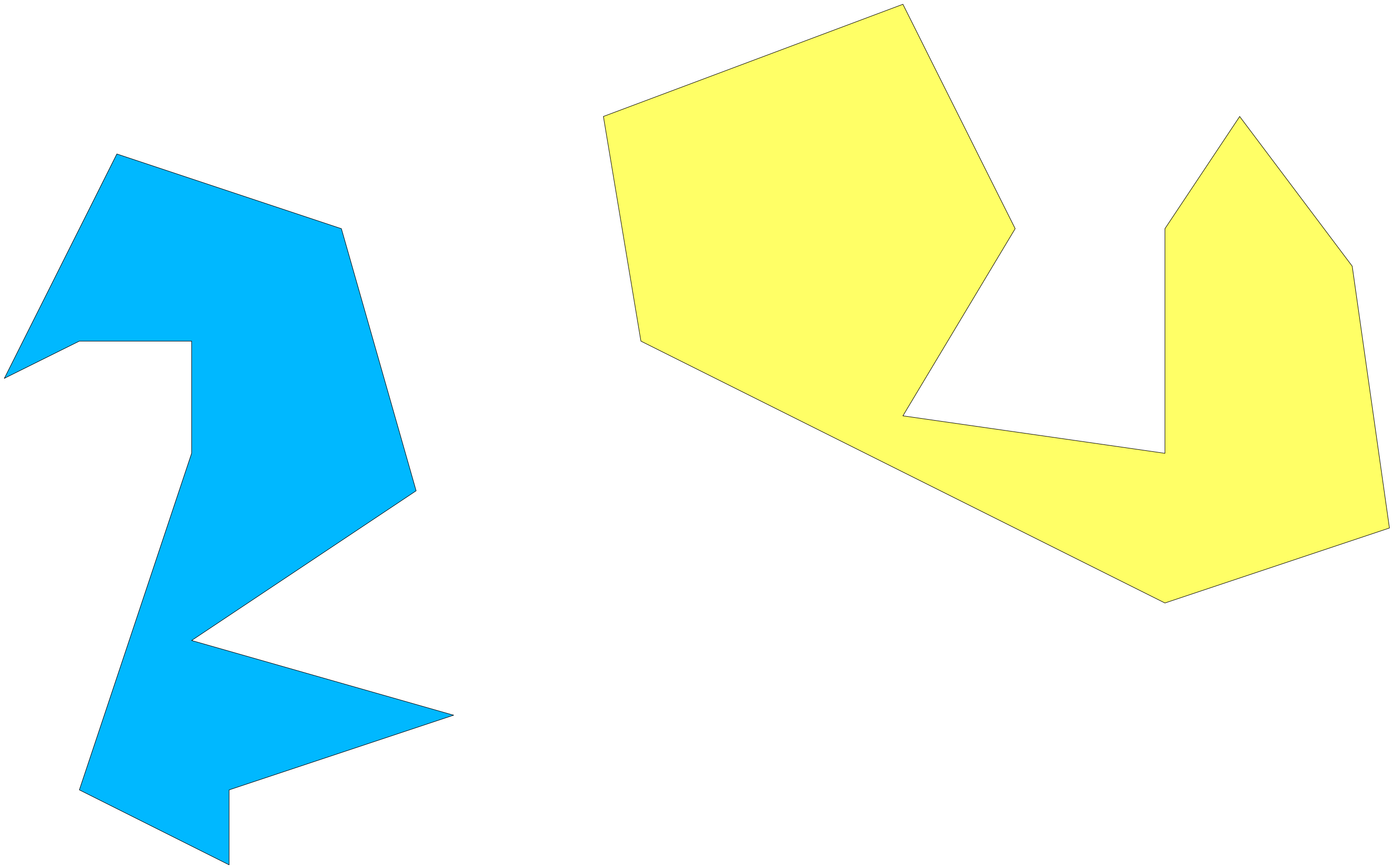
Erkennung von Kollisionen



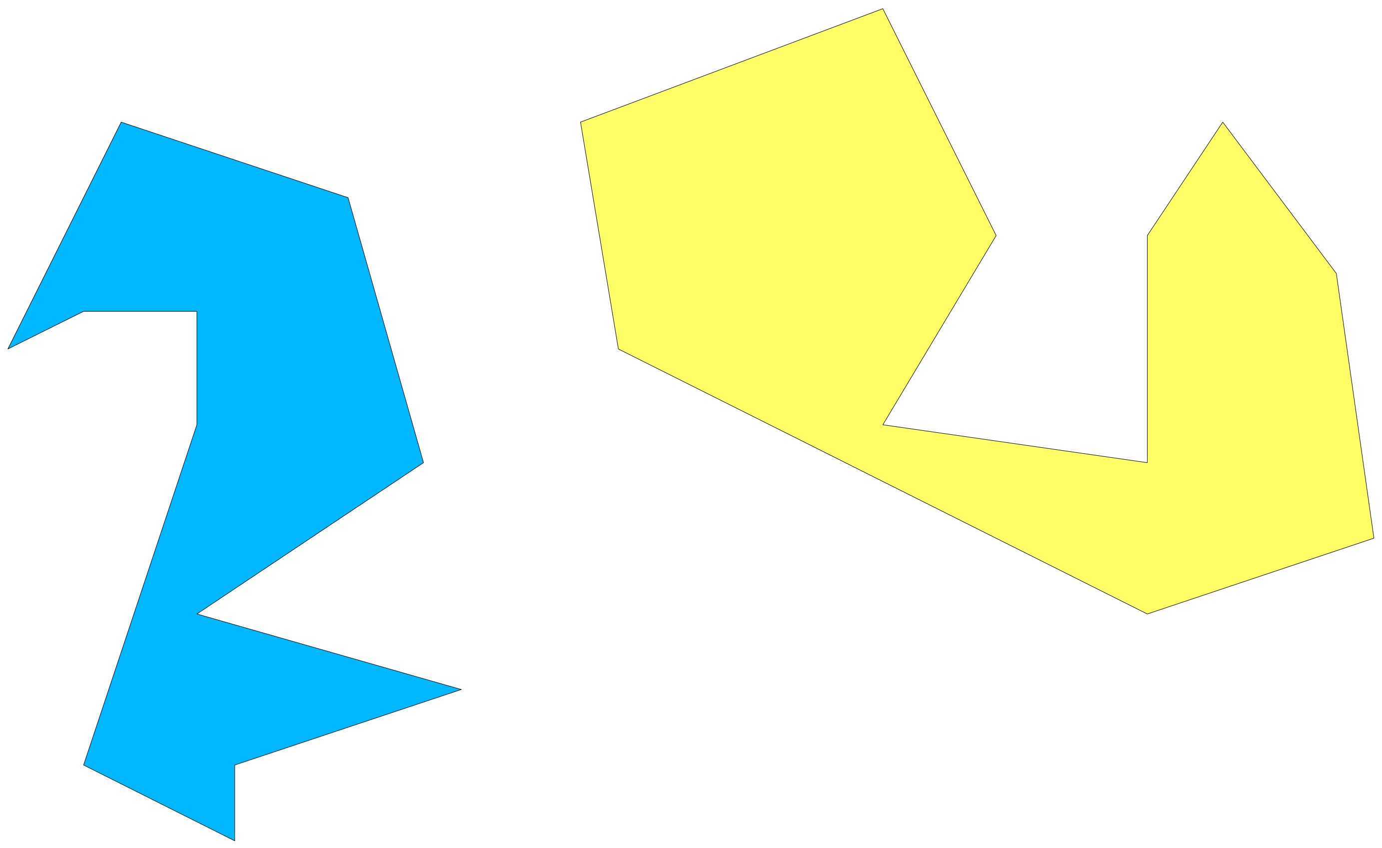
Erkennung von Kollisionen



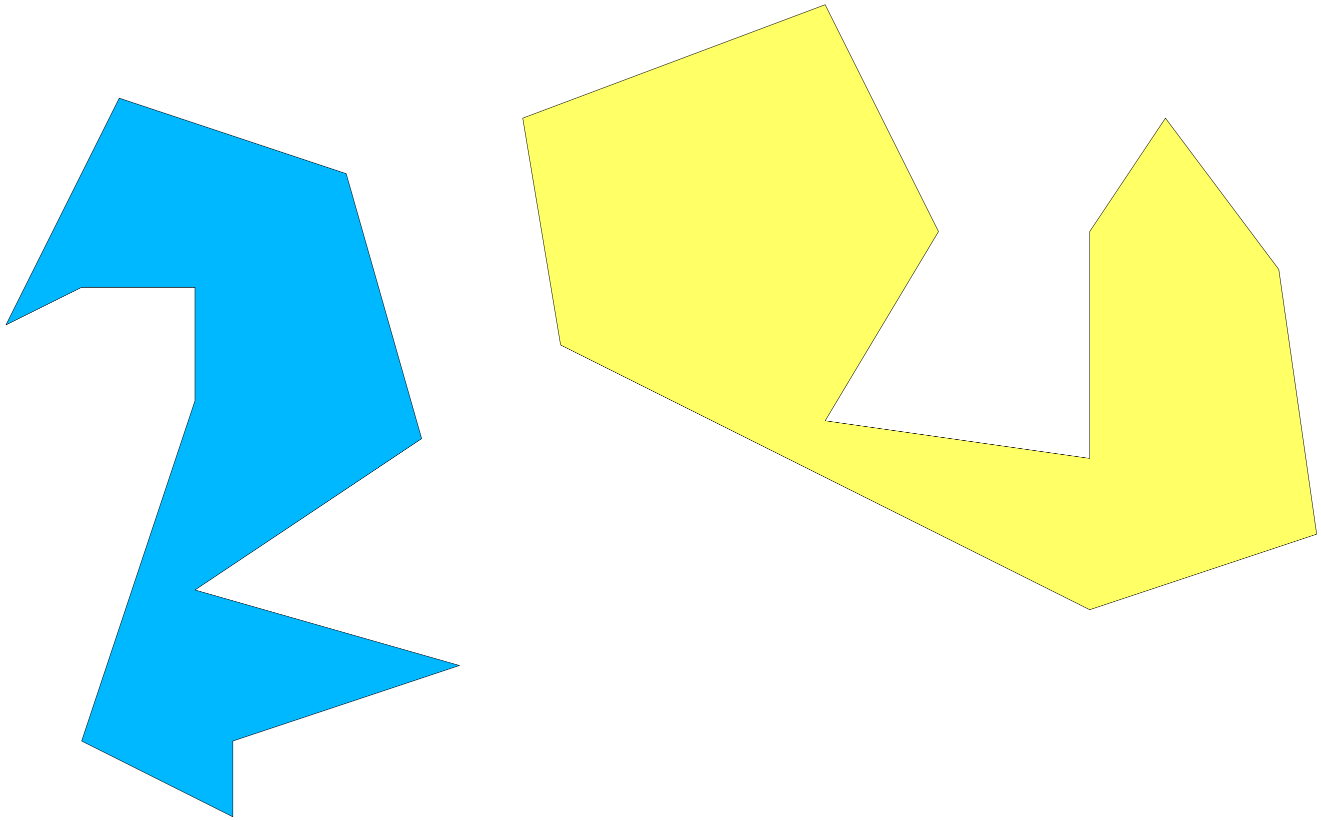
Erkennung von Kollisionen



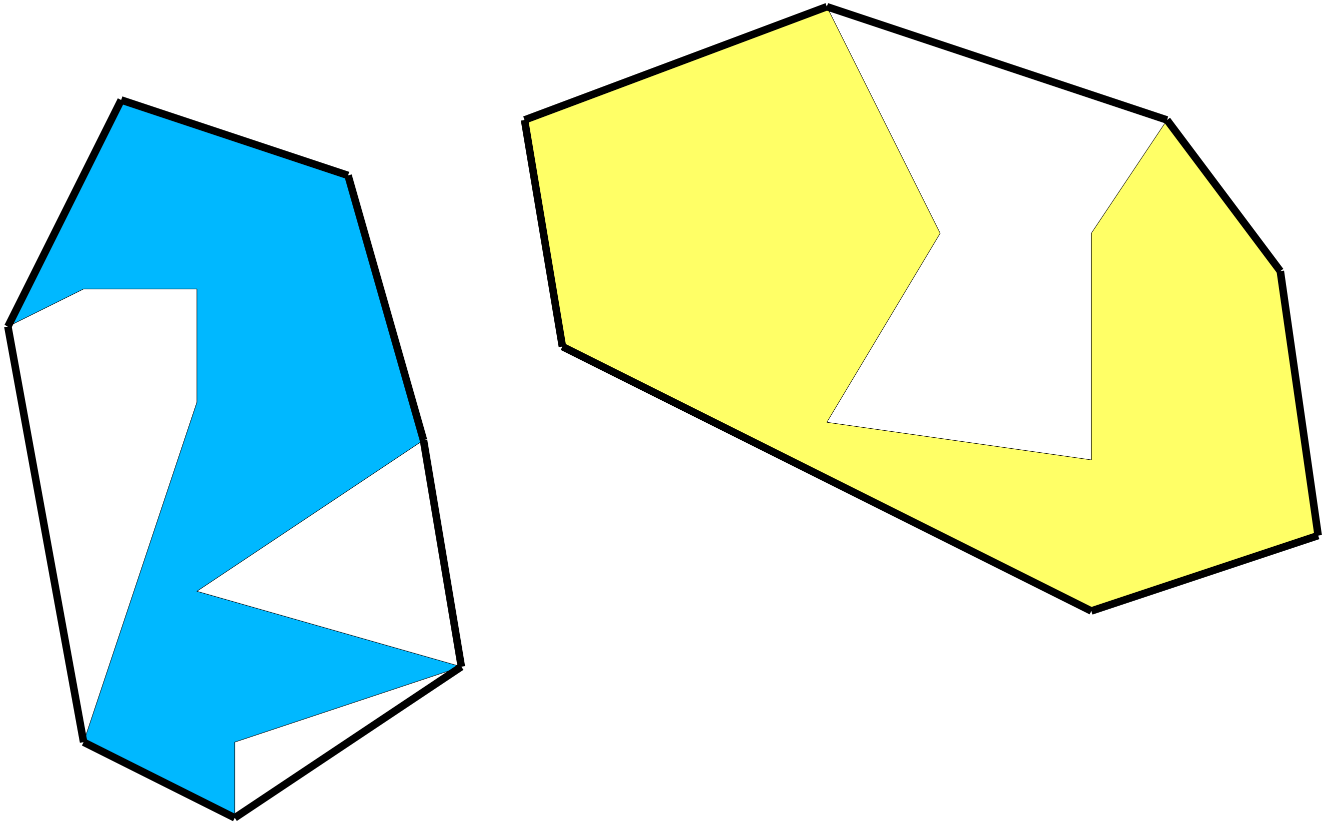
Erkennung von Kollisionen



Erkennung von Kollisionen



Erkennung von Kollisionen



3. Ein inkrementeller Algorithmus

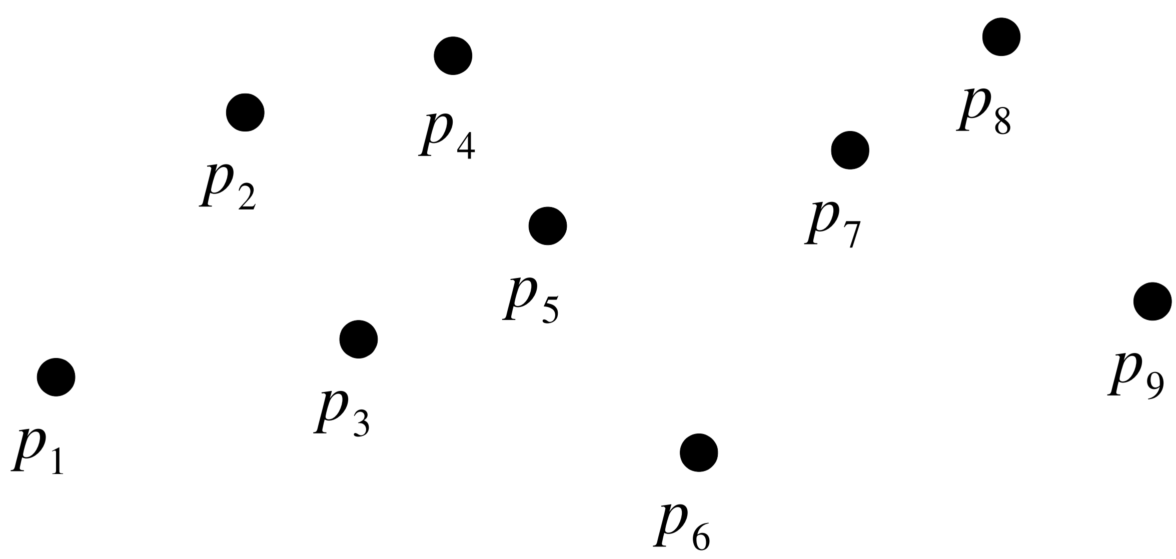
Zunächst einige vereinfachende Annahmen

P enthält mindestens drei Punkte.

Keine drei Punkte aus P liegen auf einer Geraden.

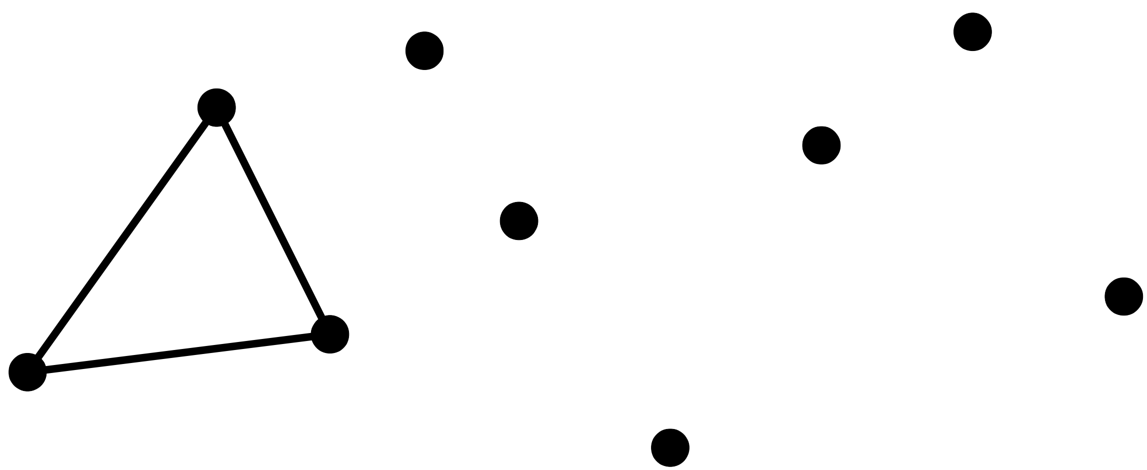
Die Punkte aus P haben paarweise verschiedene x-Koordinaten.

1. Schritt:
Wir sortieren die Punkte in P nach aufsteigenden x-Koordinaten.



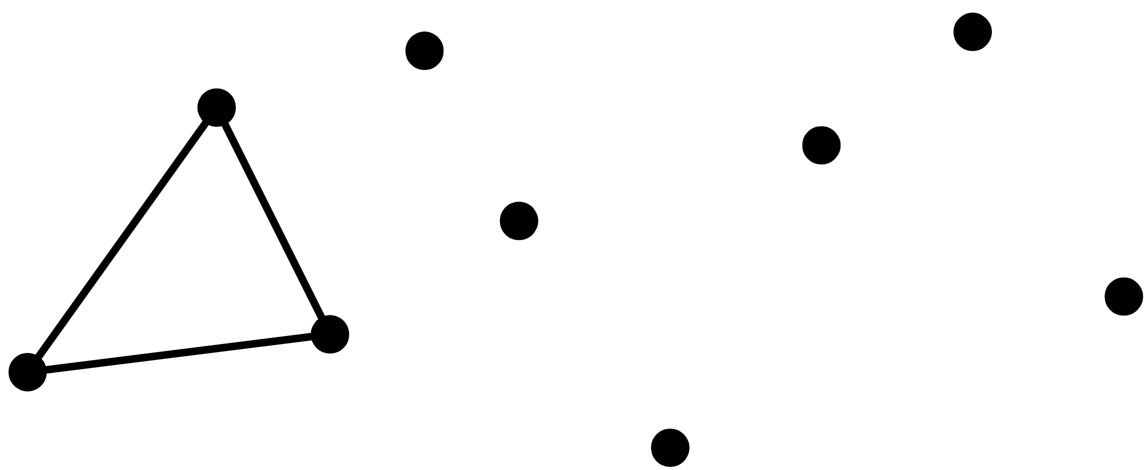
2. Schritt:

Wir bestimmen nun die konvexe Hülle der ersten drei Punkte.

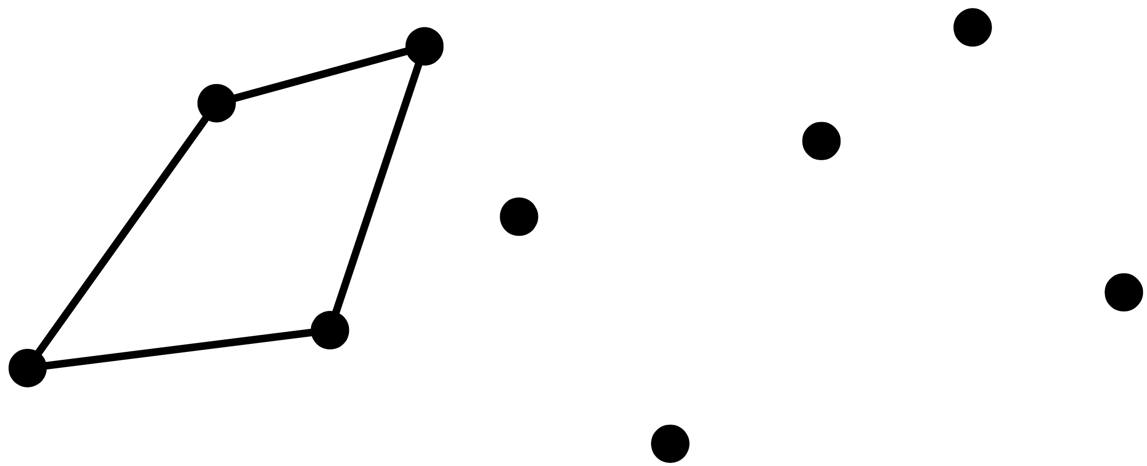


Weiterer Ablauf:

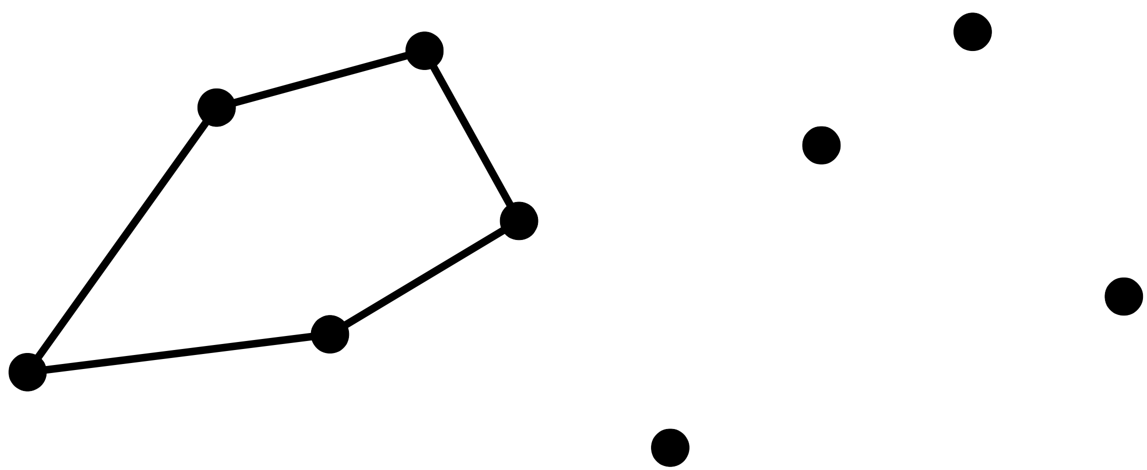
In jedem Schritt nehmen wir einen weiteren Punkt hinzu.



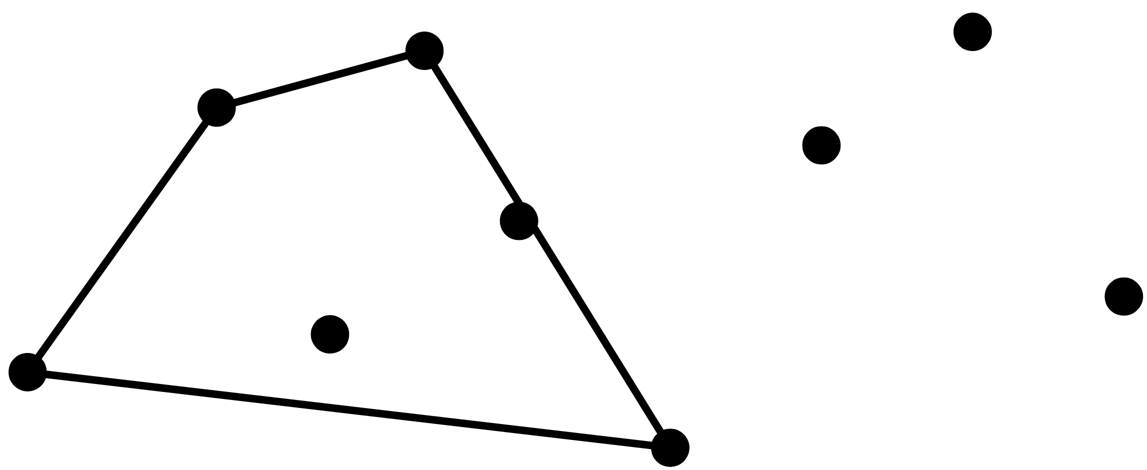
Wir nehmen den vierten Punkt hinzu.

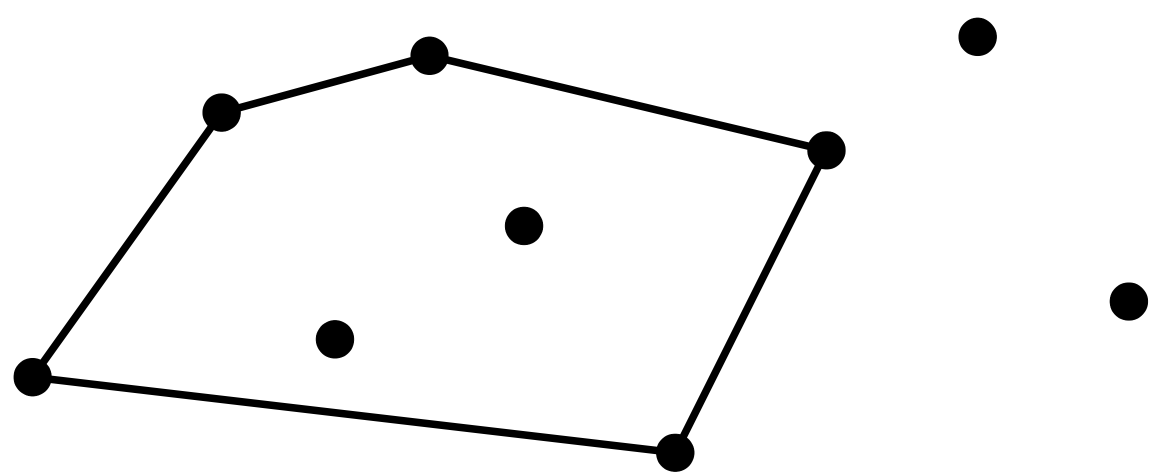


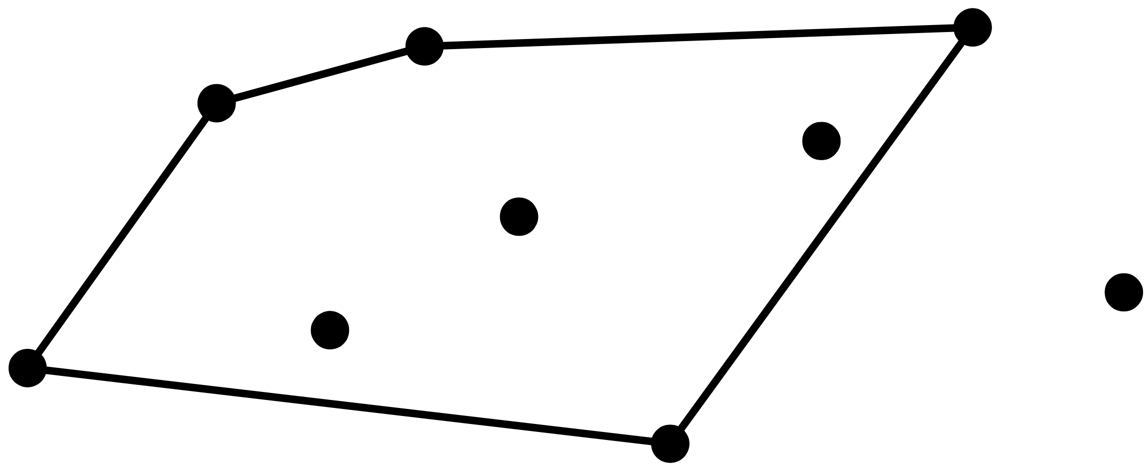
Wir nehmen den fünften Punkt hinzu.



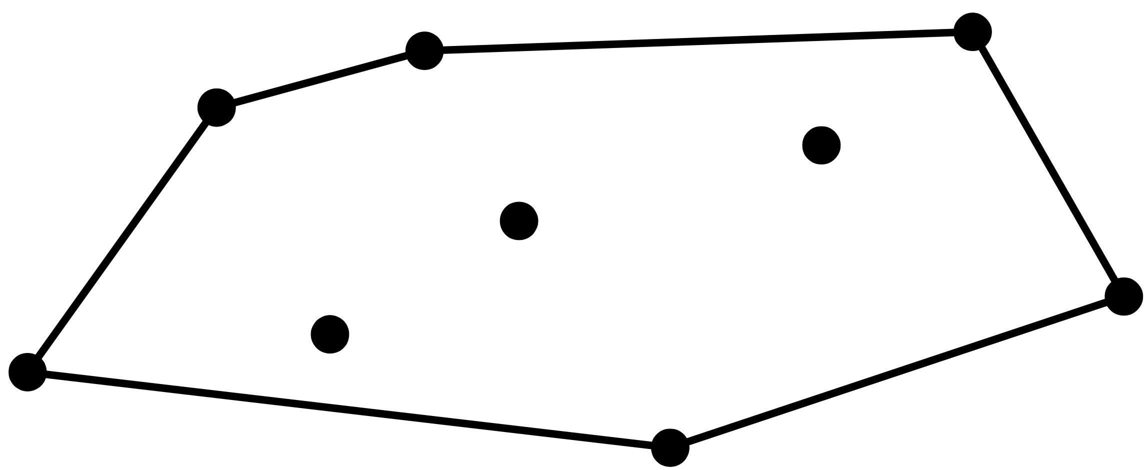
Wir nehmen den sechsten Punkt hinzu usw.



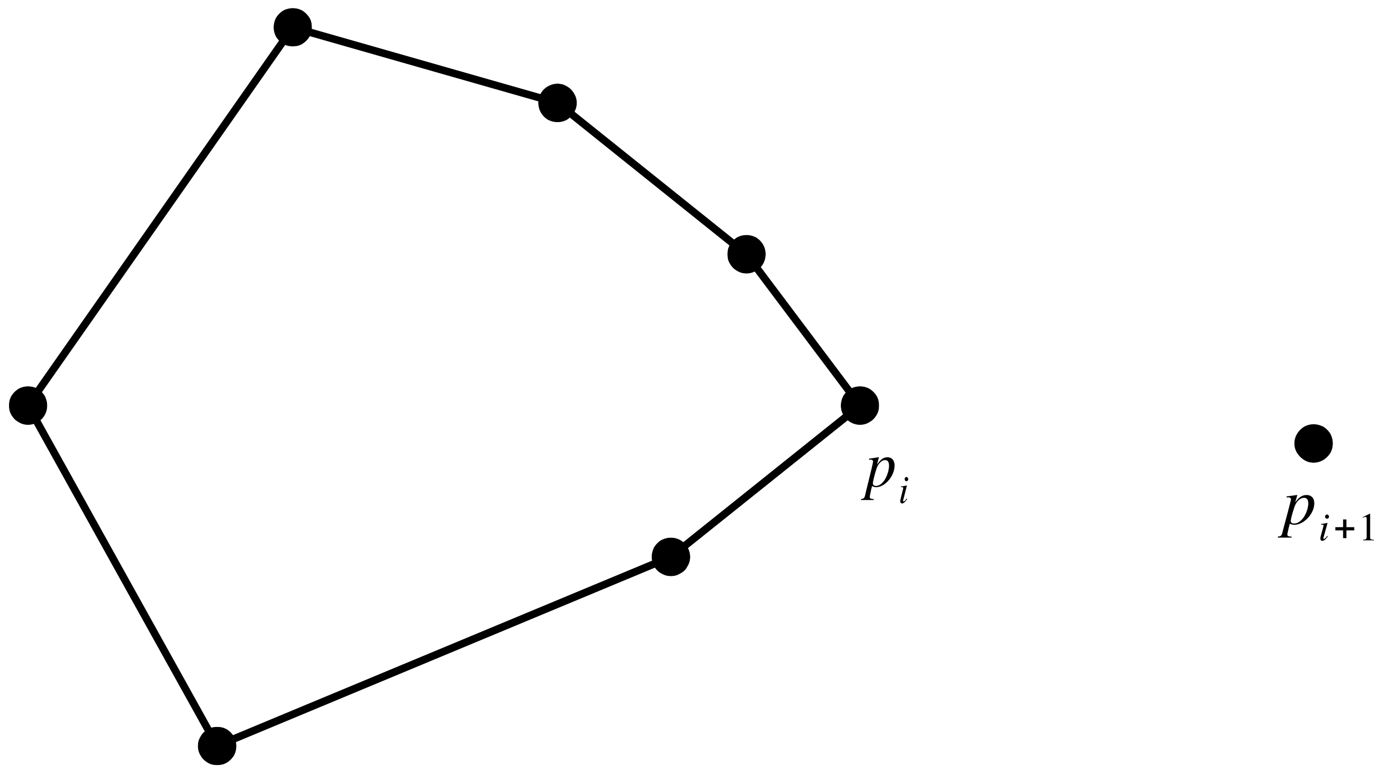


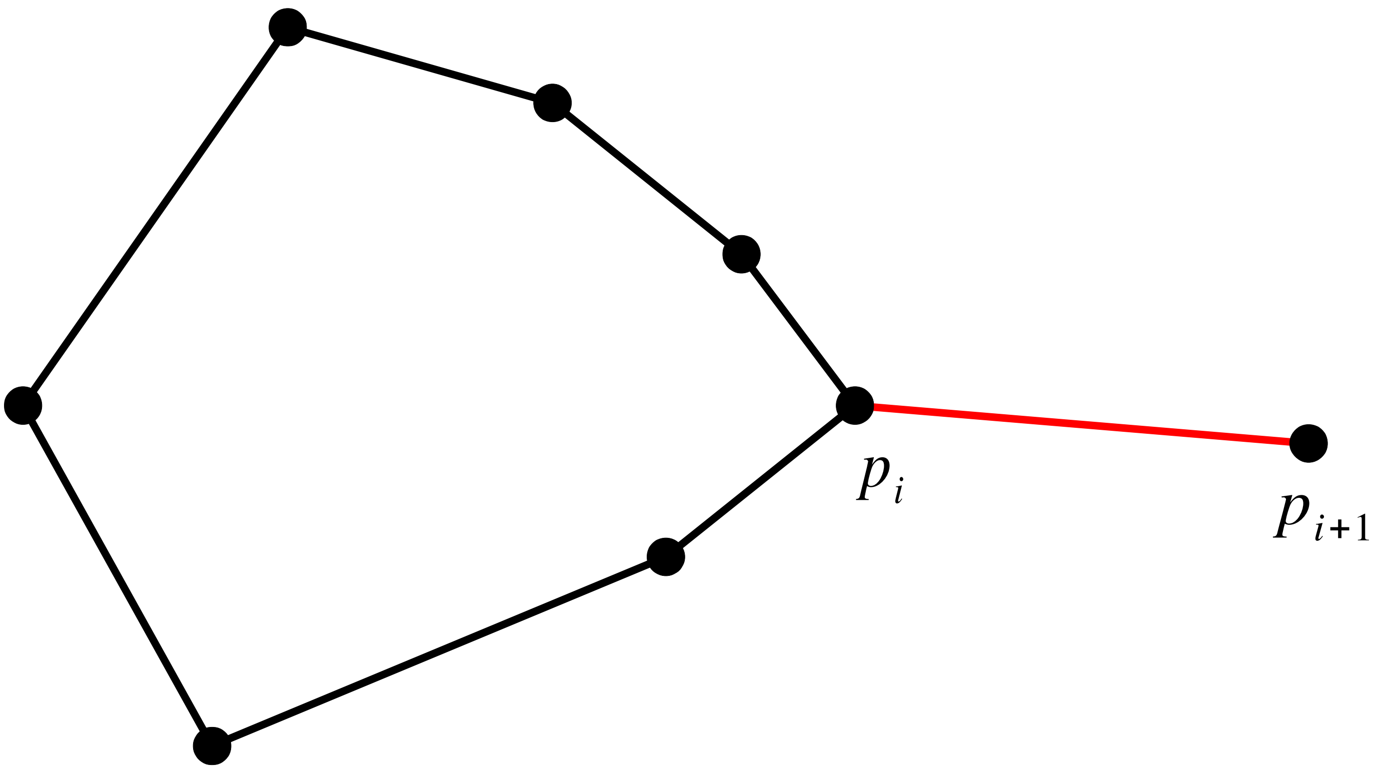


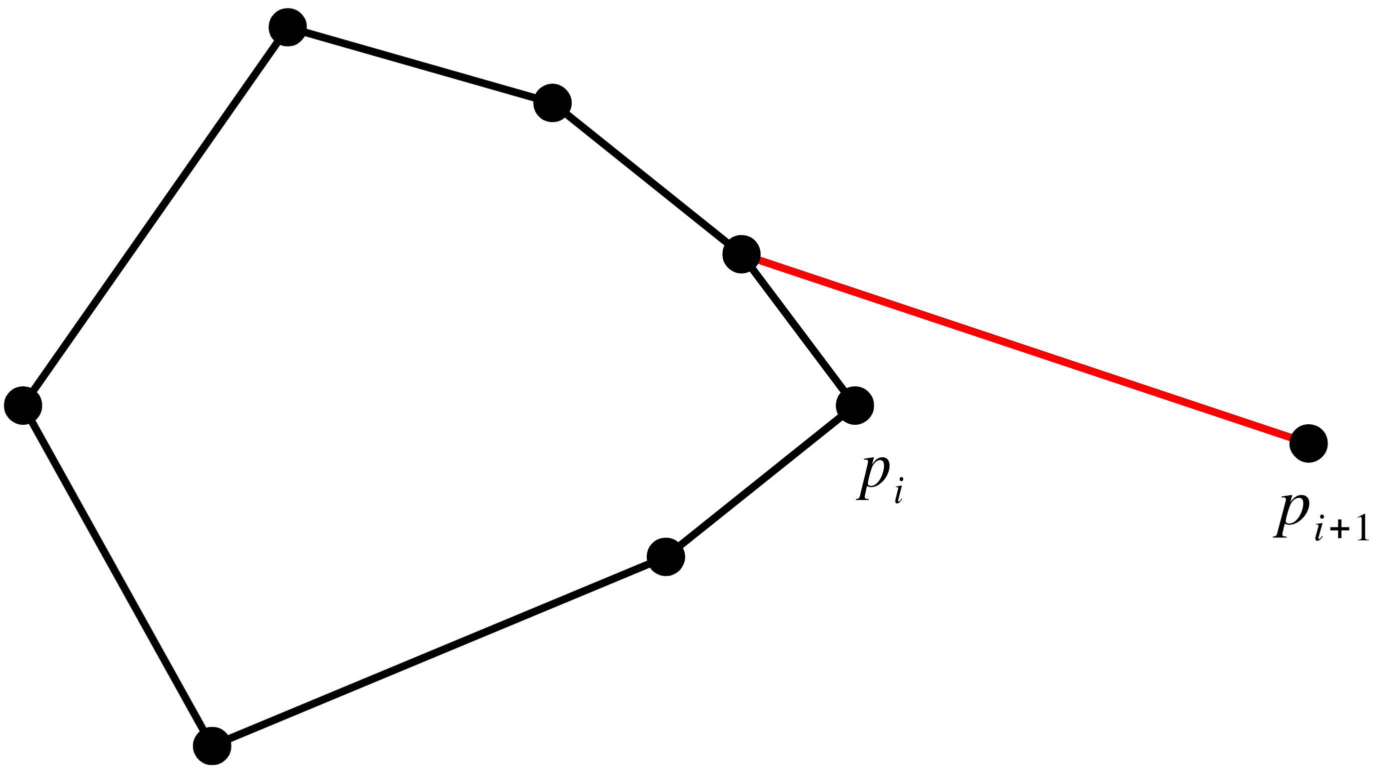
Wenn wir den letzten Punkt hinzugenommen haben, sind wir fertig.



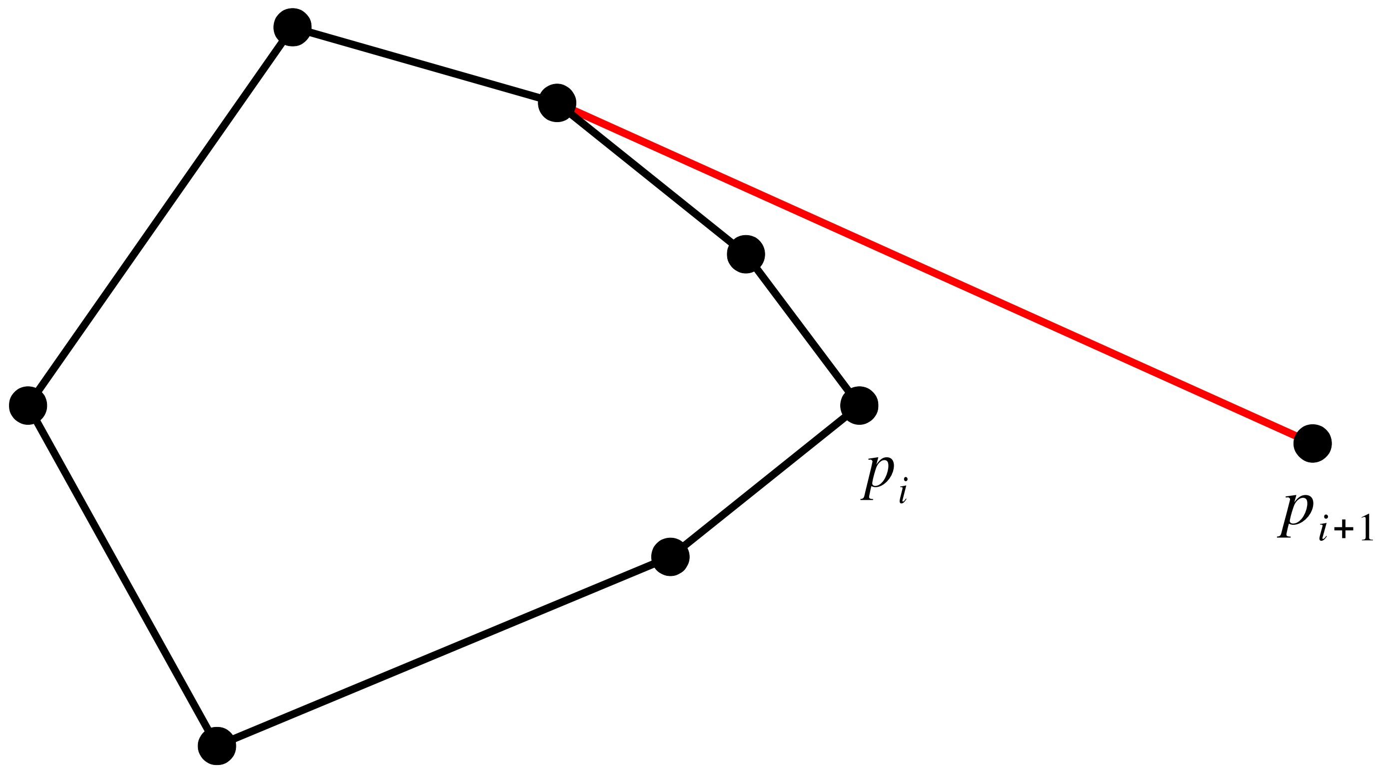
Es bleibt noch zu beschreiben, wie wir im Detail vorgehen beim Übergang von $\mathrm{CH}(p_1, p_2, \ldots, p_i)$ zu $\mathrm{CH}(p_1, p_2, \ldots, p_{i+1})$.



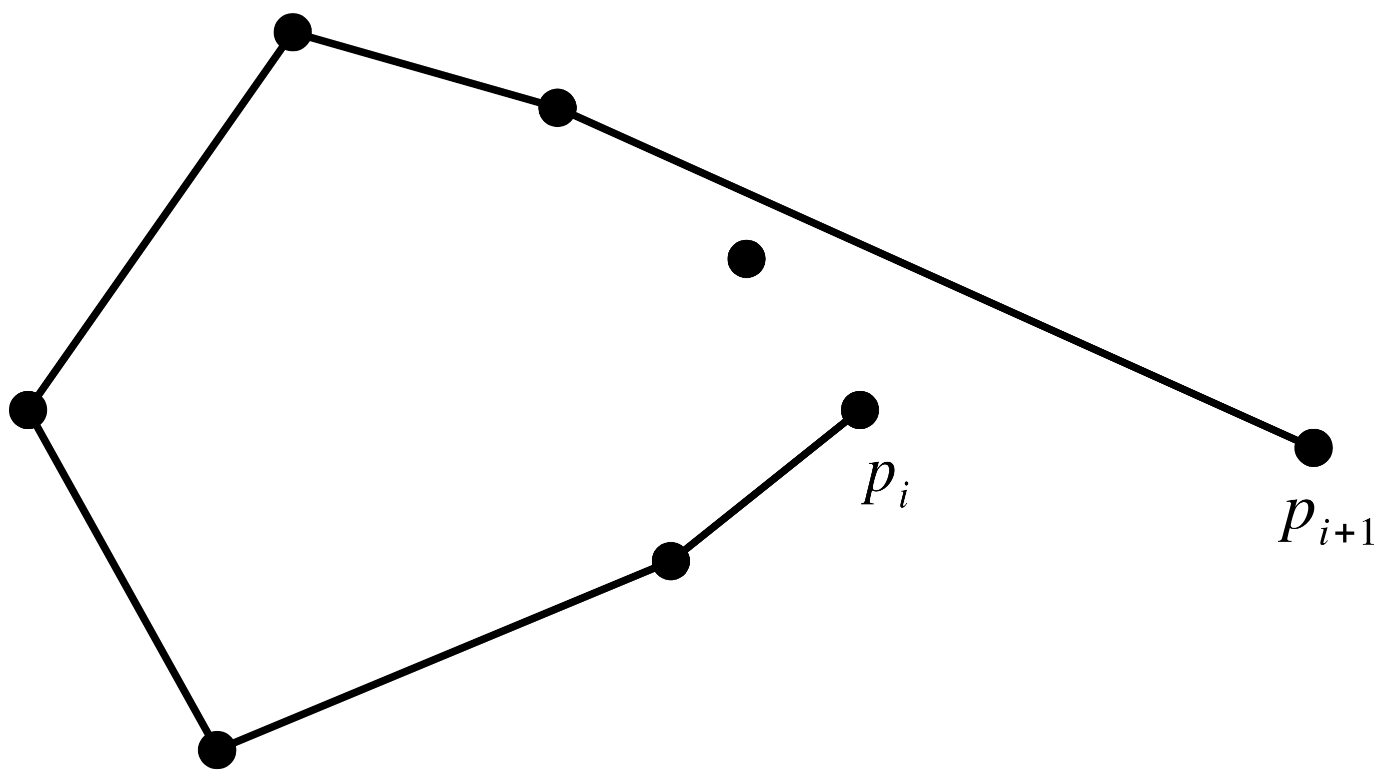

p_i
p_{i+1}

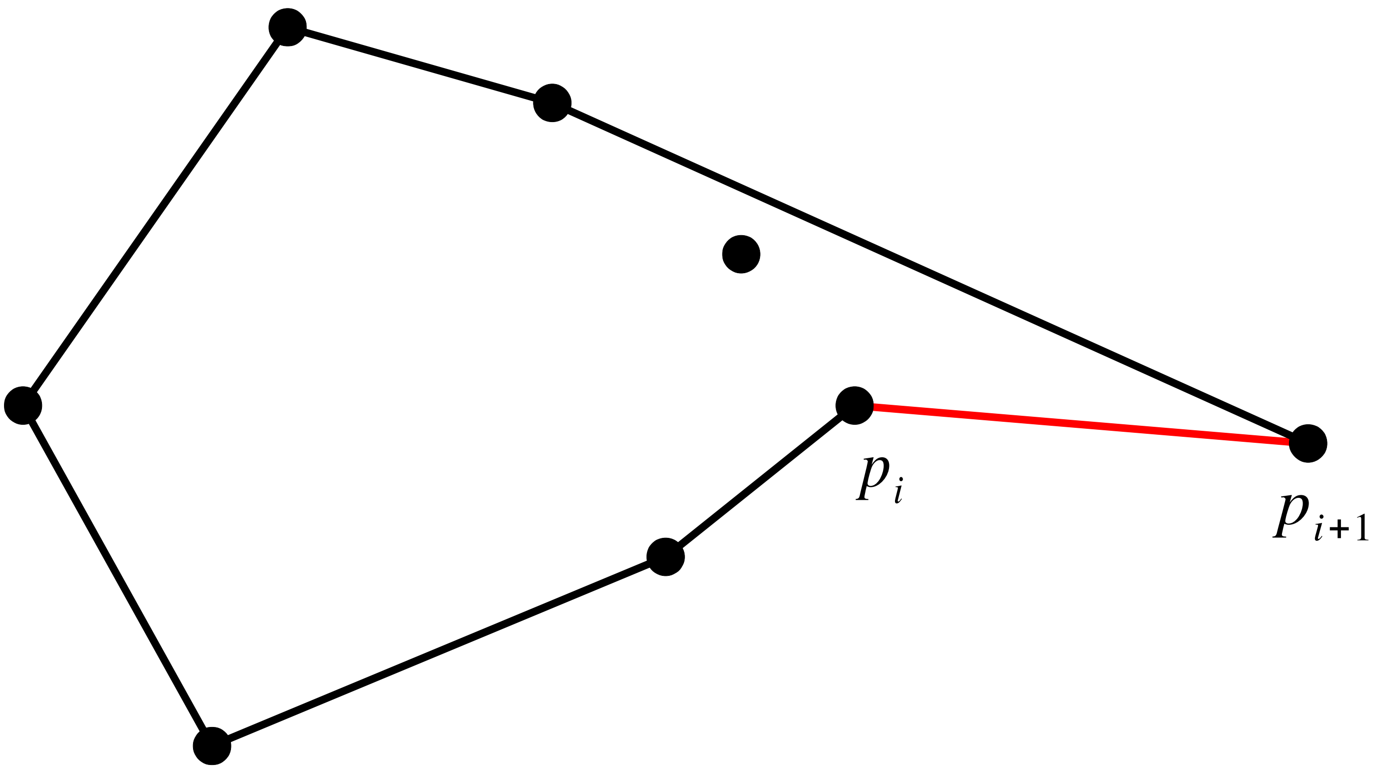

p_i
p_{i+1}

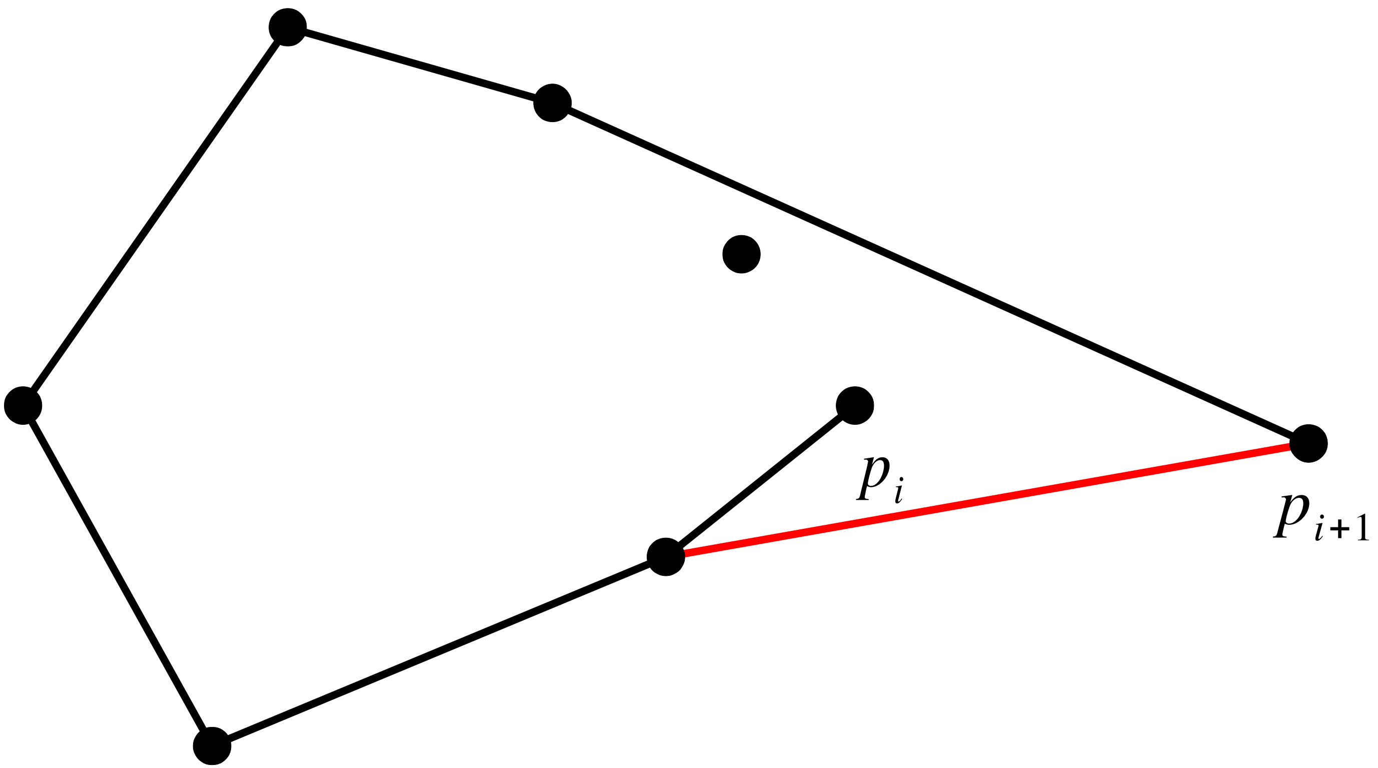
Jetzt haben wir den Punkt gefunden, wo die obere Tangente aufliegt.



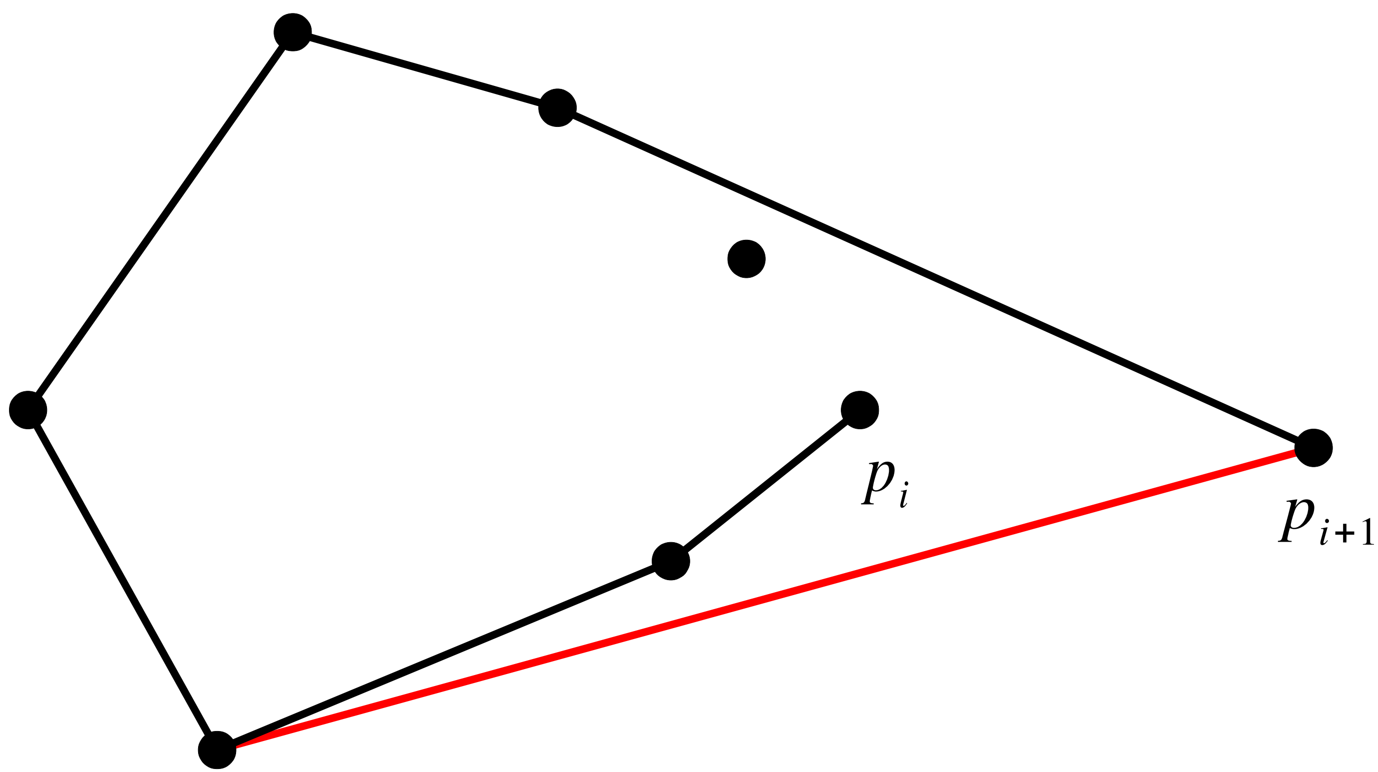
Damit können wir den oberen Teil der konvexen Hülle aktualisieren.



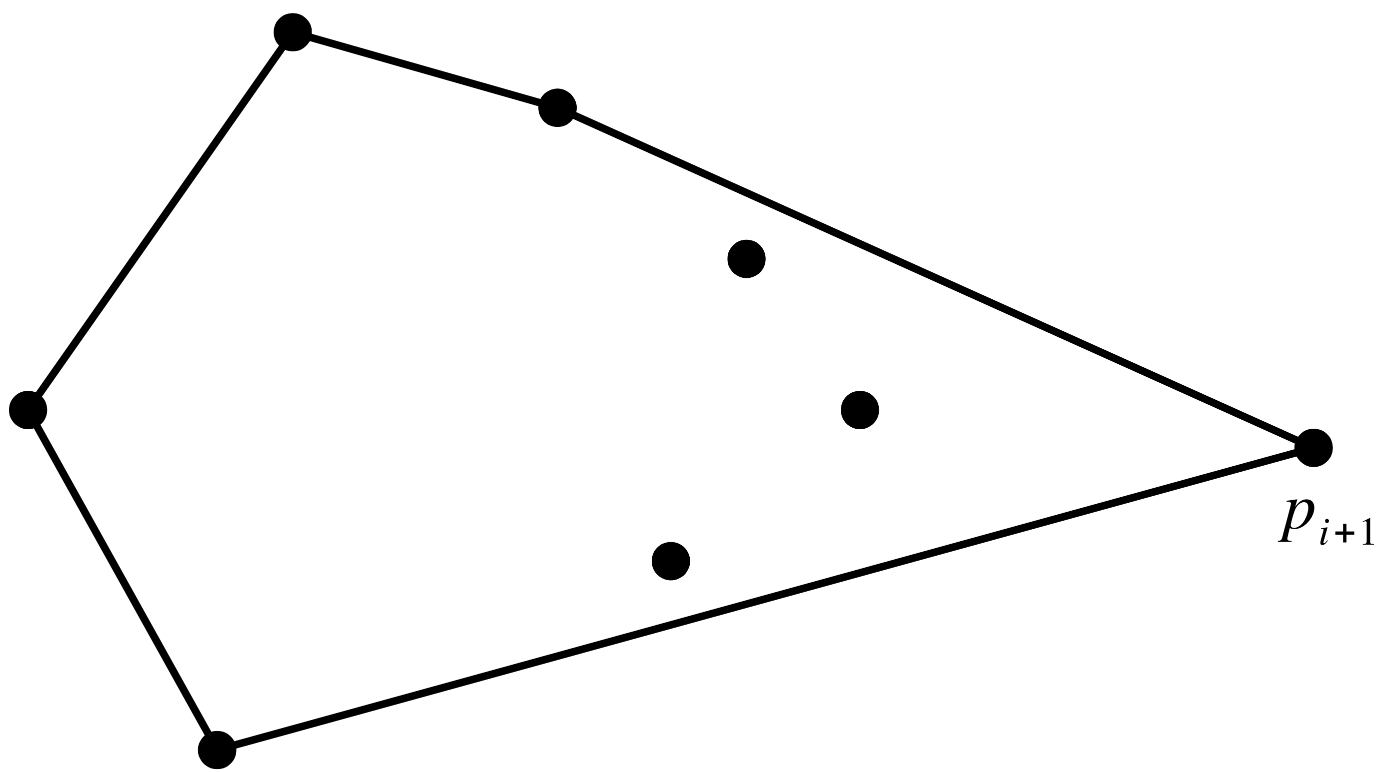

p_i
p_{i+1}


p_i
p_{i+1}

Jetzt haben wir auch den Punkt gefunden, wo die untere Tangente aufliegt.



Damit können wir auch noch den unteren Teil der konvexen Hülle aktualisieren und sind fertig mit der Verarbeitung von p_{i+1}.



Analyse des deterministischen inkrementellen Algorithmus

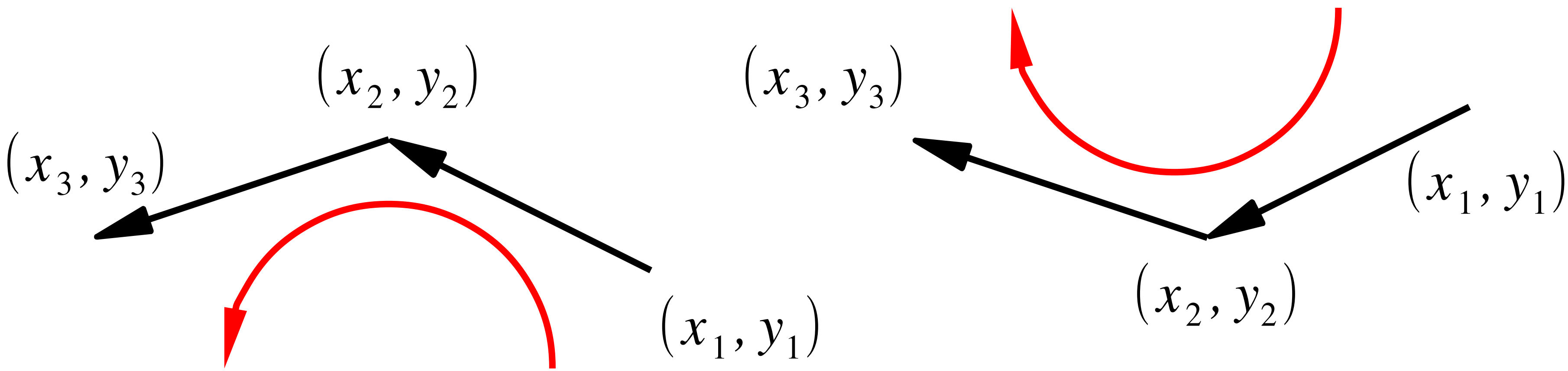
Übergang von $\mathrm{CH}(p_1, p_2, \dots, p_i)$ zu $\mathrm{CH}(p_1, p_2, \dots, p_{i+1})$

Amortisierte Kosten: $O(1)$

Laufzeit ohne Sortieren: $O(n)$

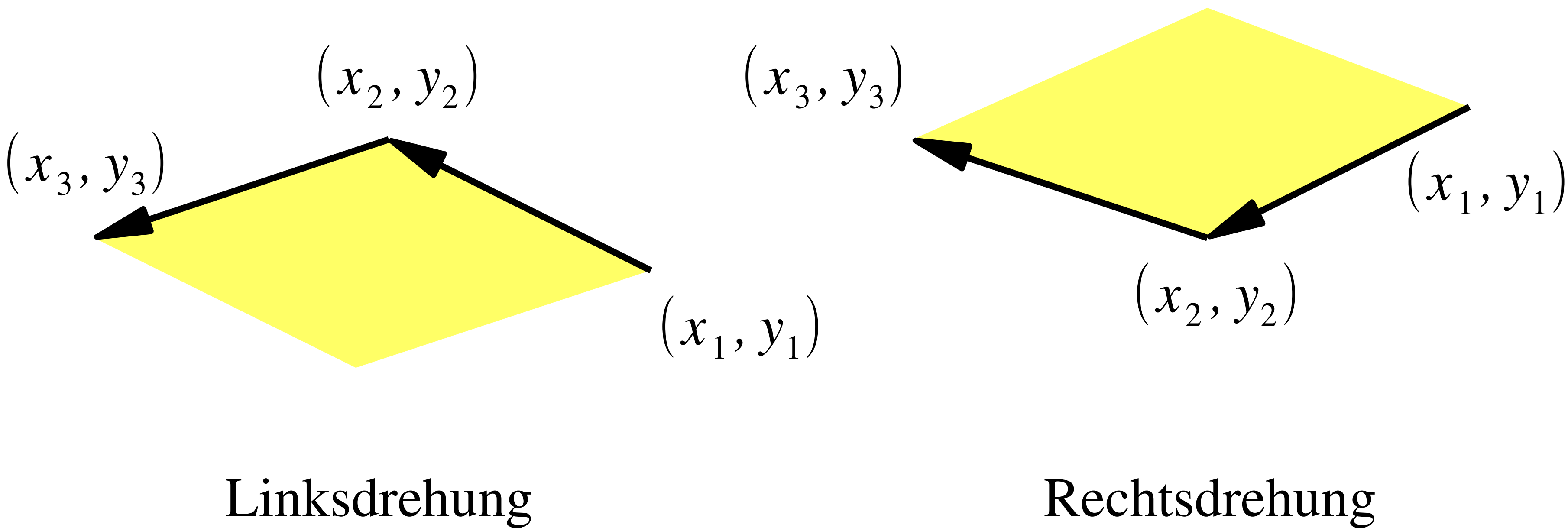
Laufzeit mit Sortieren: $O(n \log n)$

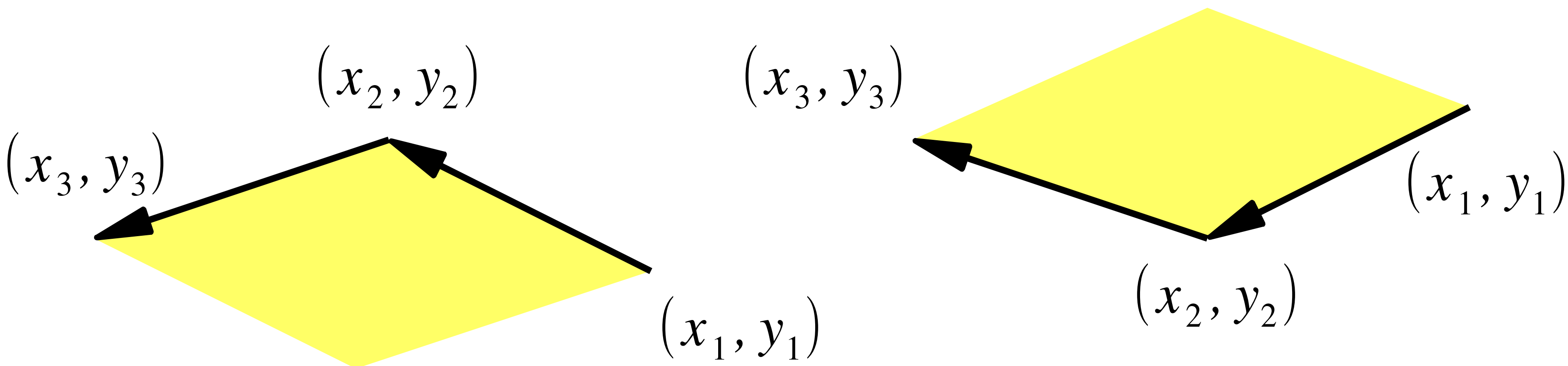
4. Test der relativen Lage von Punkten



Linksdrehung

Rechtsdrehung


(x_2, y_2)
(x_3, y_3)
(x_1, y_1)
Linksdrehung
(x_3, y_3)
(x_1, y_1)
(x_2, y_2)
Rechtsdrehung



Linksdrehung

$$\begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix} > 0$$

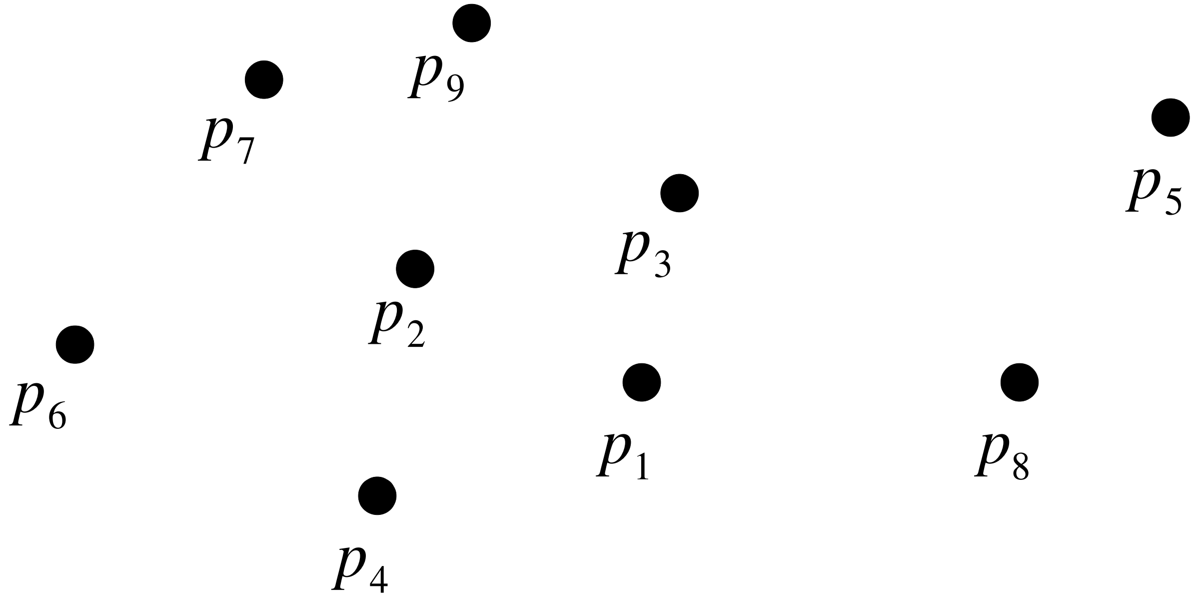
Rechtsdrehung

$$\begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix} < 0$$

5. Randomisierung des inkrementellen Algorithmus

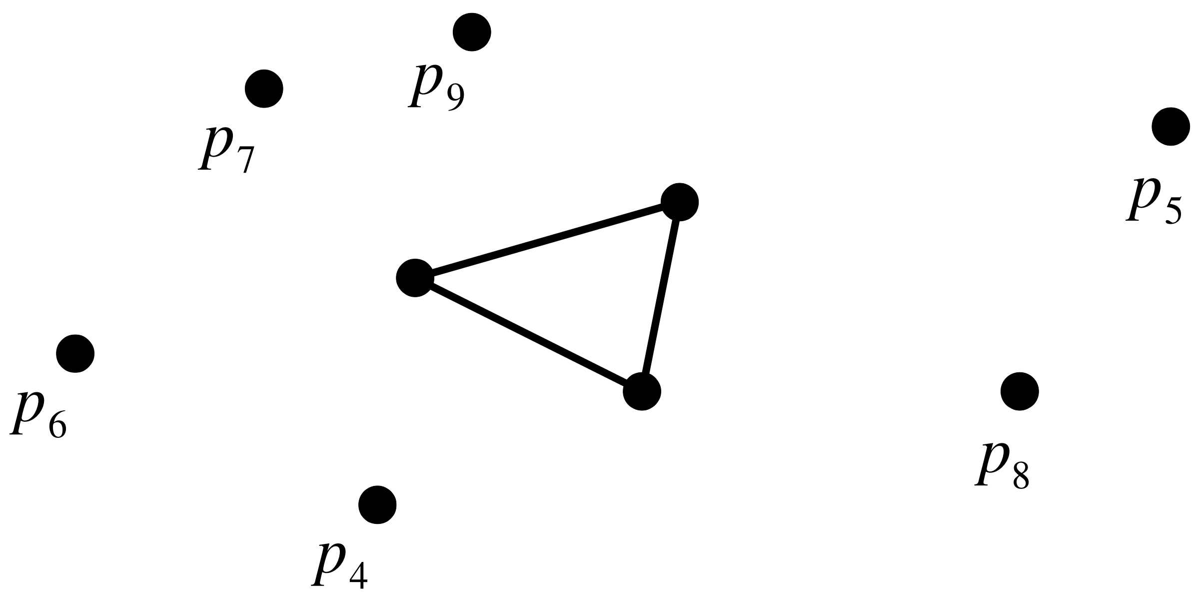
Grundidee:

Wir verarbeiten die Punkte in P in einer zufälligen Reihenfolge.



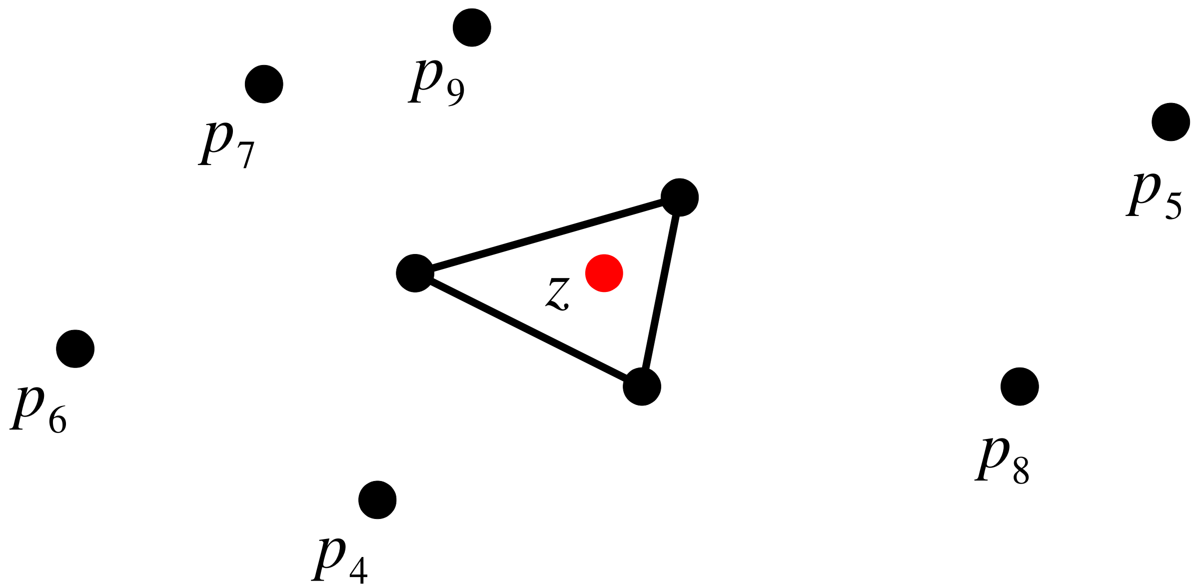
1. Schritt:

Wir bestimmen das Dreieck mit den Ecken p_1, p_2 und p_3.



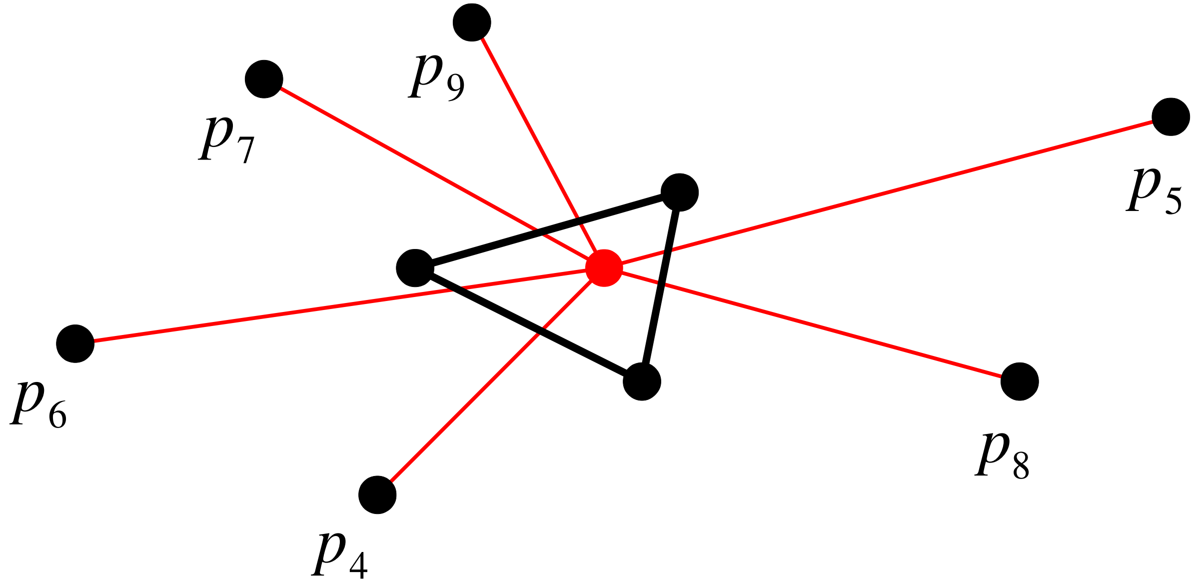
2. Schritt:

Wir bestimmen einen Punkt z im Inneren des Dreiecks.



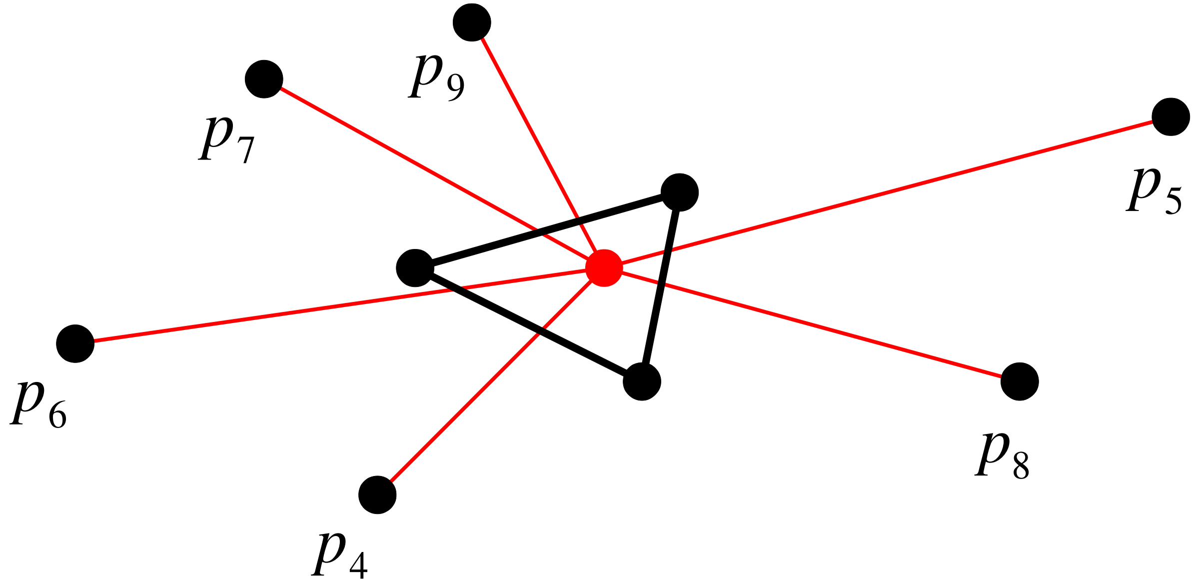
3. Schritt:

Für jeden Punkt p_i bestimmen wir die Seite des Dreiecks, die der Strahl mit Anfangspunkt z durch p_i schneidet.



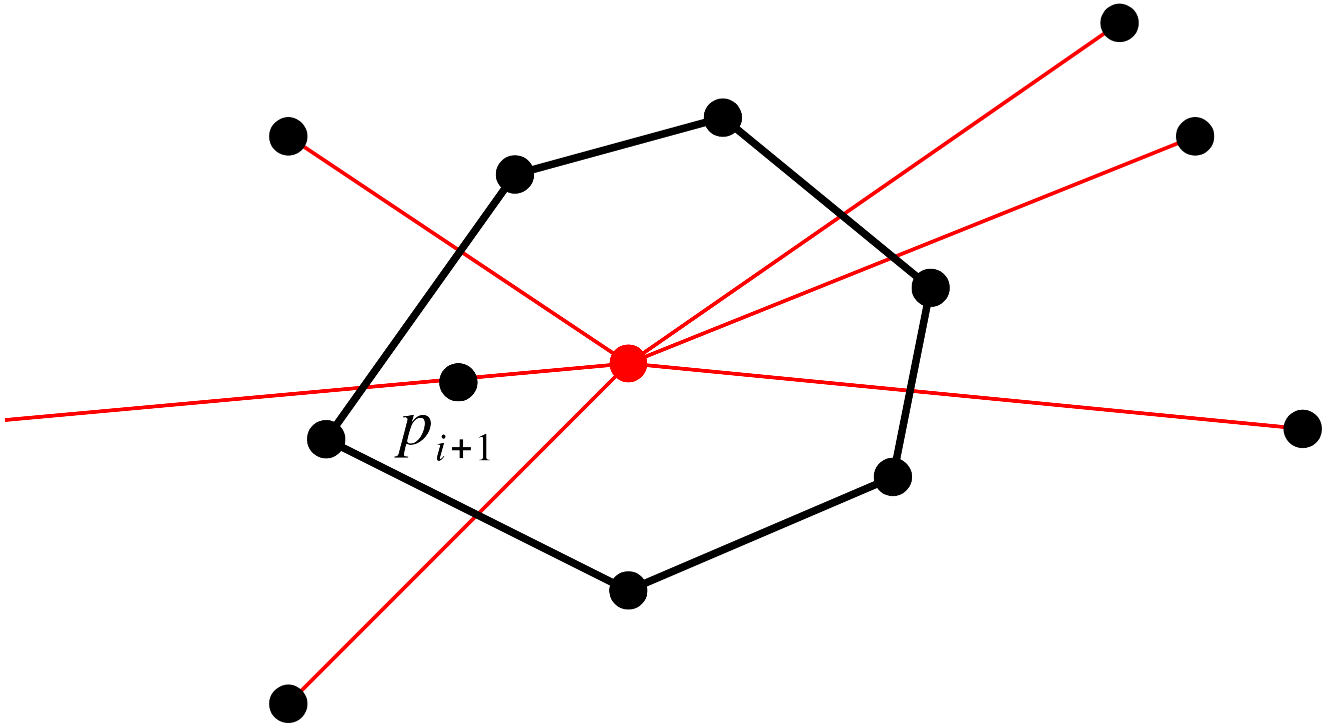
Wir stellen sicher:

Jeder Punkt weiß, welche Seite des Dreiecks sein Strahl schneidet.
Umgekehrt weiß jede Dreiecksseite, die Strahlen welcher Punkte sie schneiden.

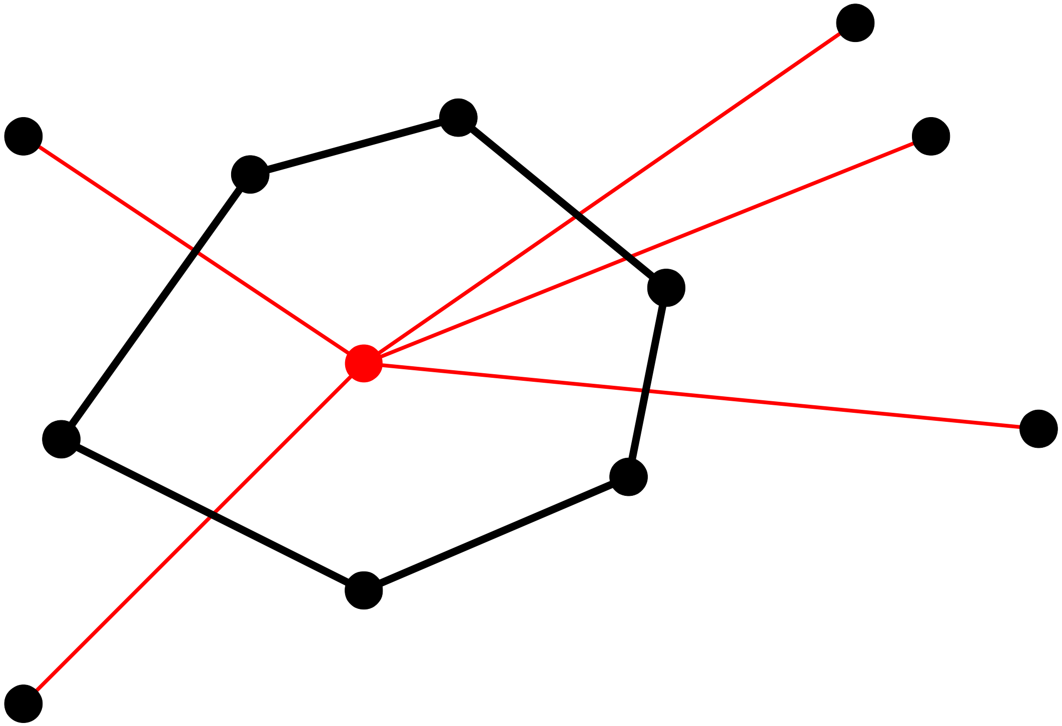


Übergang von $\mathrm{CH}(p_1, p_2, \ldots, p_i)$ zu $\mathrm{CH}(p_1, p_2, \ldots, p_{i+1})$

1.Fall: einfach!

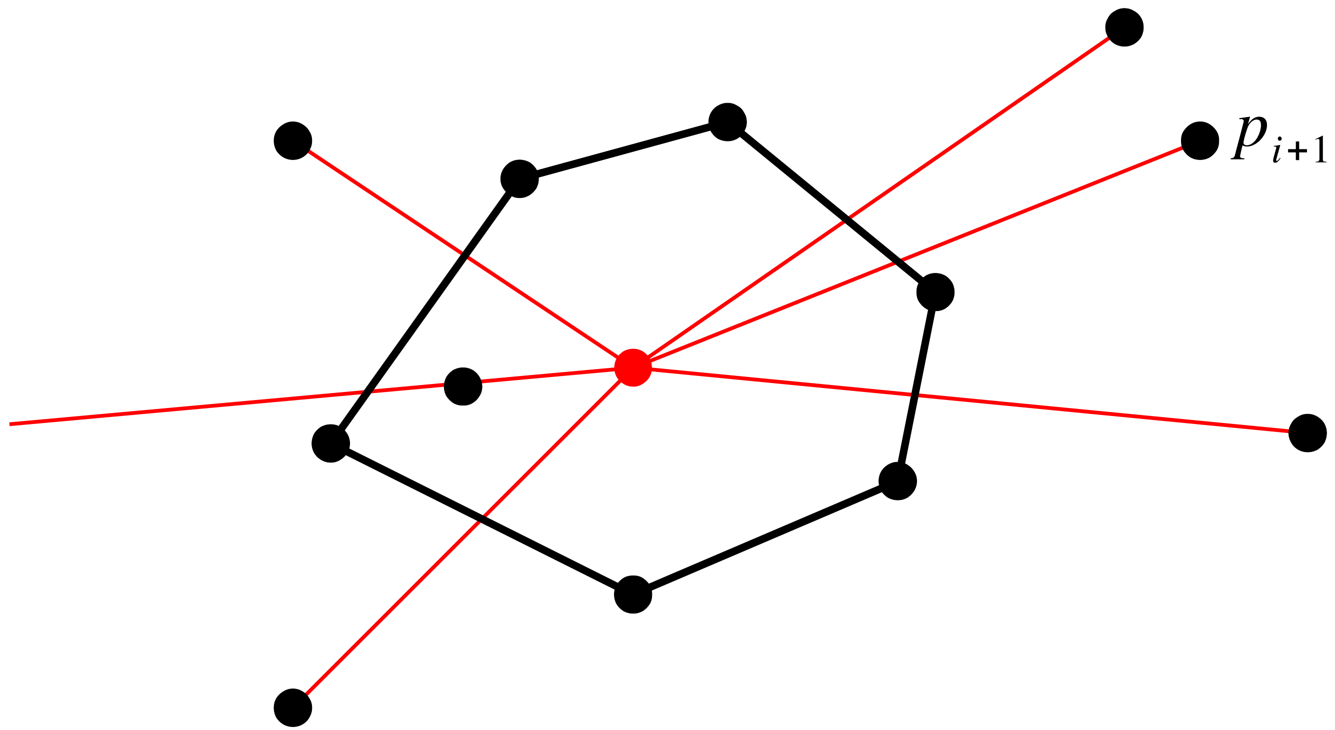


1.Fall: Ergebnis

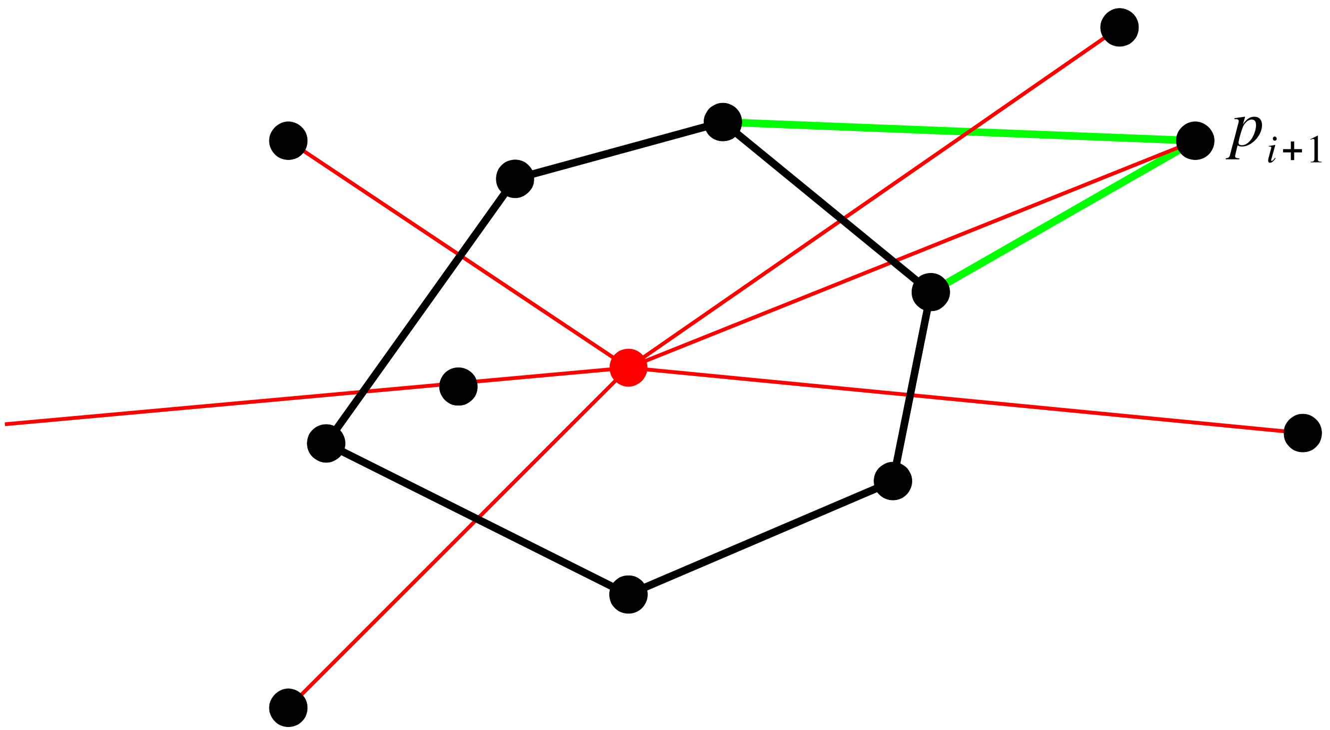


Übergang von $\mathrm{CH}(p_1, p_2, \ldots, p_i)$ zu $\mathrm{CH}(p_1, p_2, \ldots, p_{i+1})$

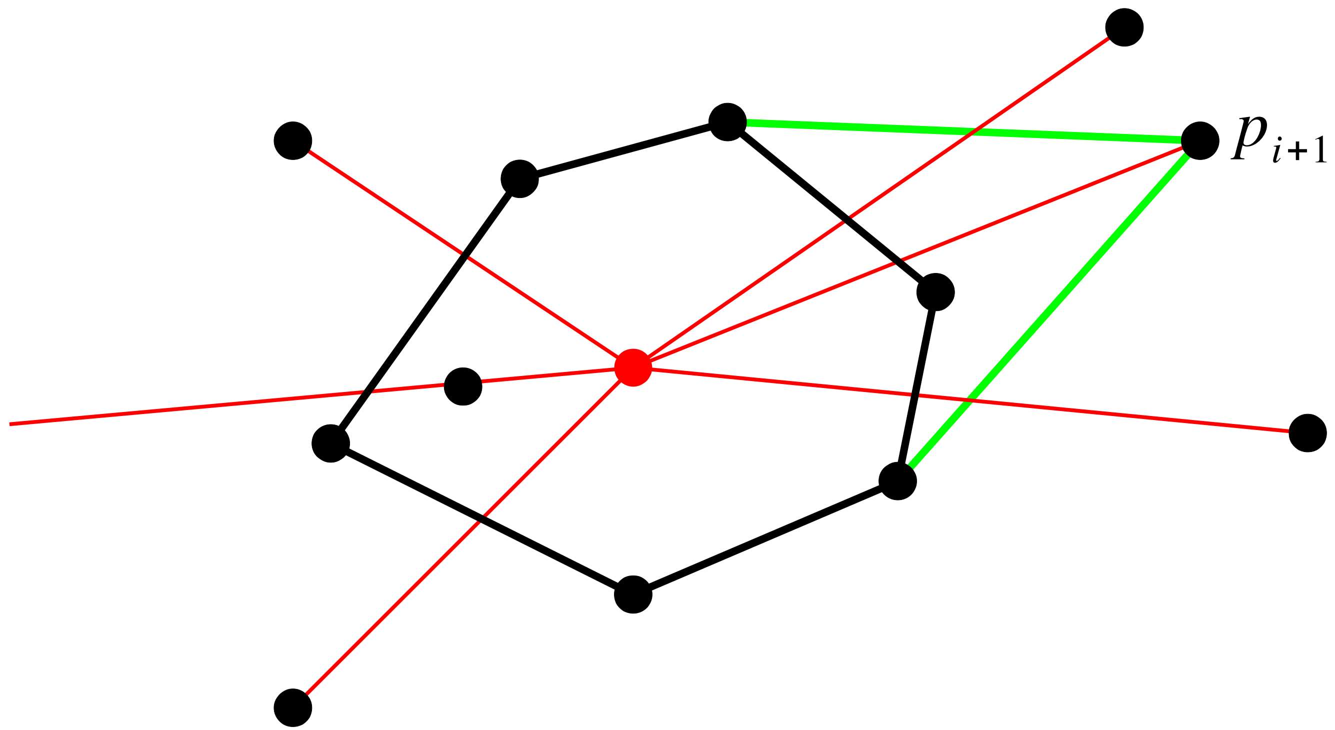
2.Fall:



Über den Strahl des Punktes finden wir die geschnittene Seite von $\mathrm{CH}(p_1, p_2, \ldots, p_i)$.

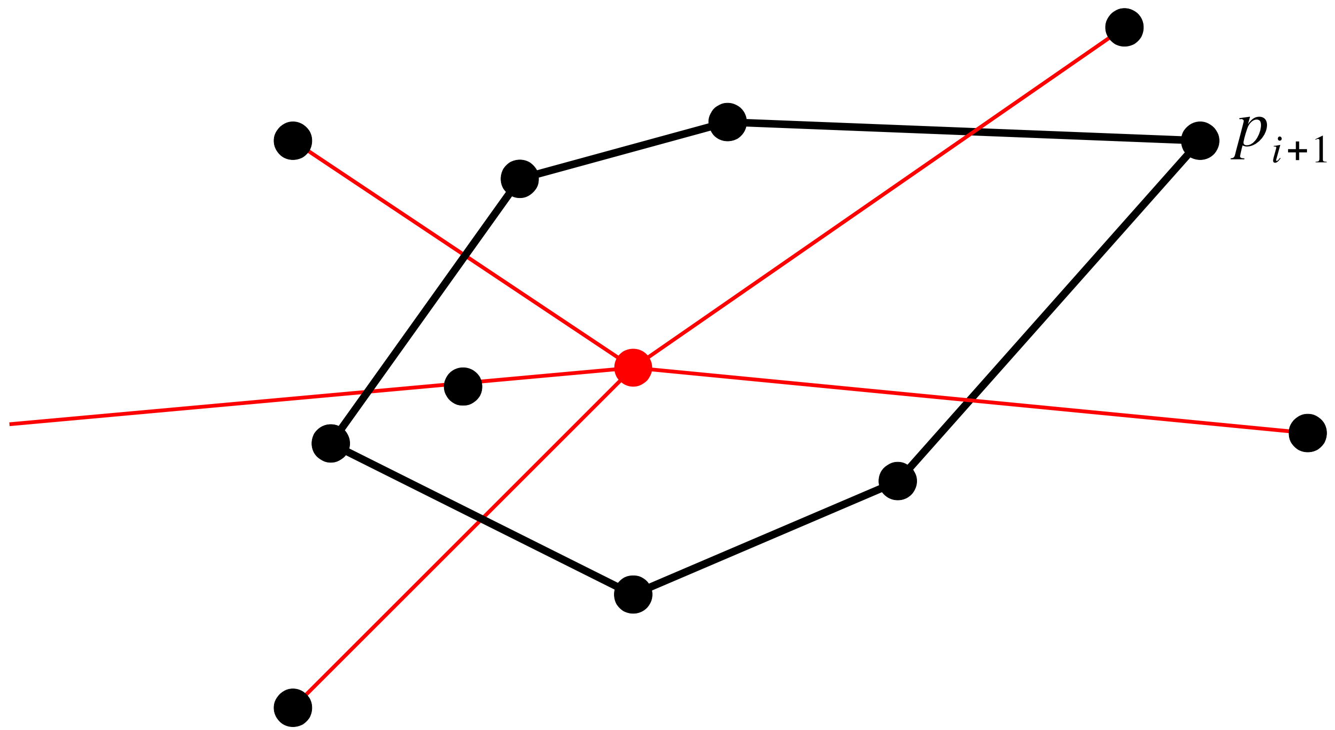


Ausgehend von dieser Seite finden wir die Punkte, wo die beiden Tangenten aufliegen.



Damit haben wir $\mathrm{CH}\left(p_1, p_2, \ldots, p_{i+1}\right)$.

2.Fall: Ergebnis



Analyse des randomisierten inkrementellen Algorithmus

Übergang von $\mathrm{CH}(p_1, p_2, \ldots, p_i)$ zu $\mathrm{CH}(p_1, p_2, \ldots, p_{i+1})$

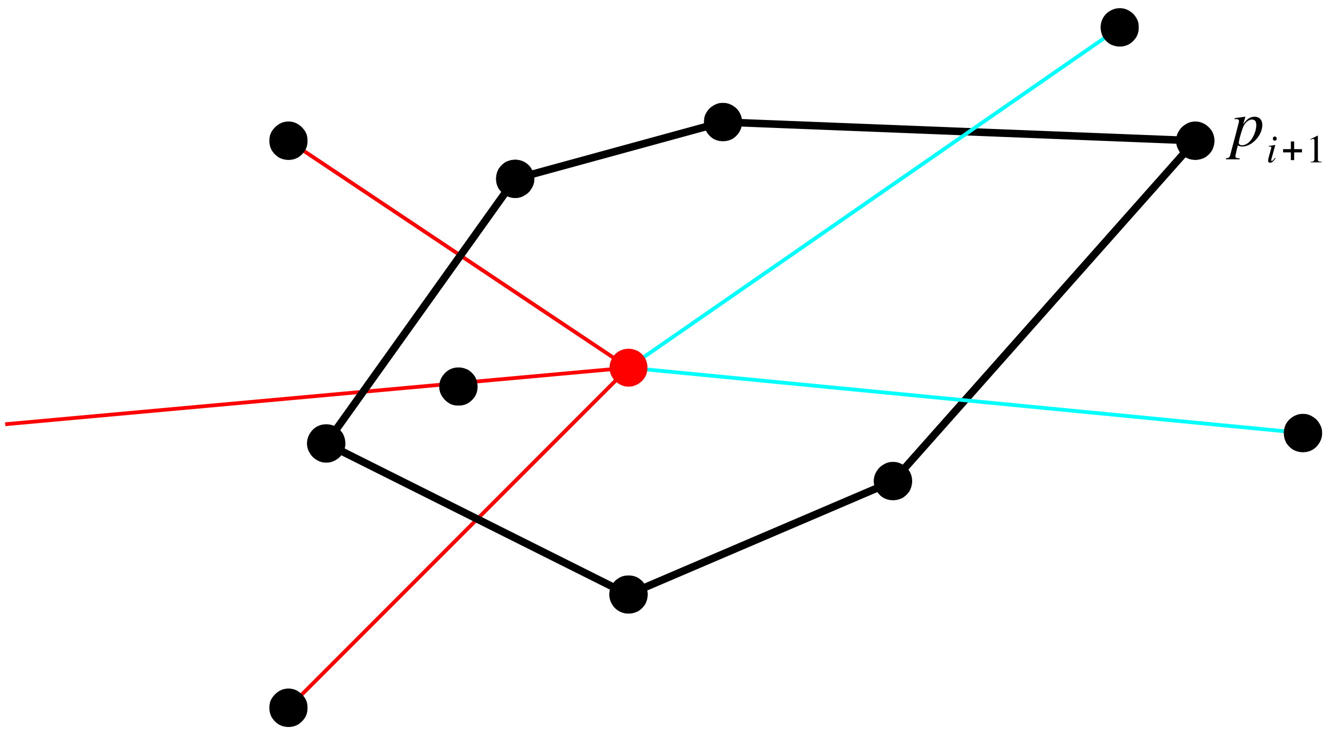
Aufwand ohne Verwaltung der Strahlen: $O(1)$ amortisiert

Aufwand zum Verwalten der Strahlen?

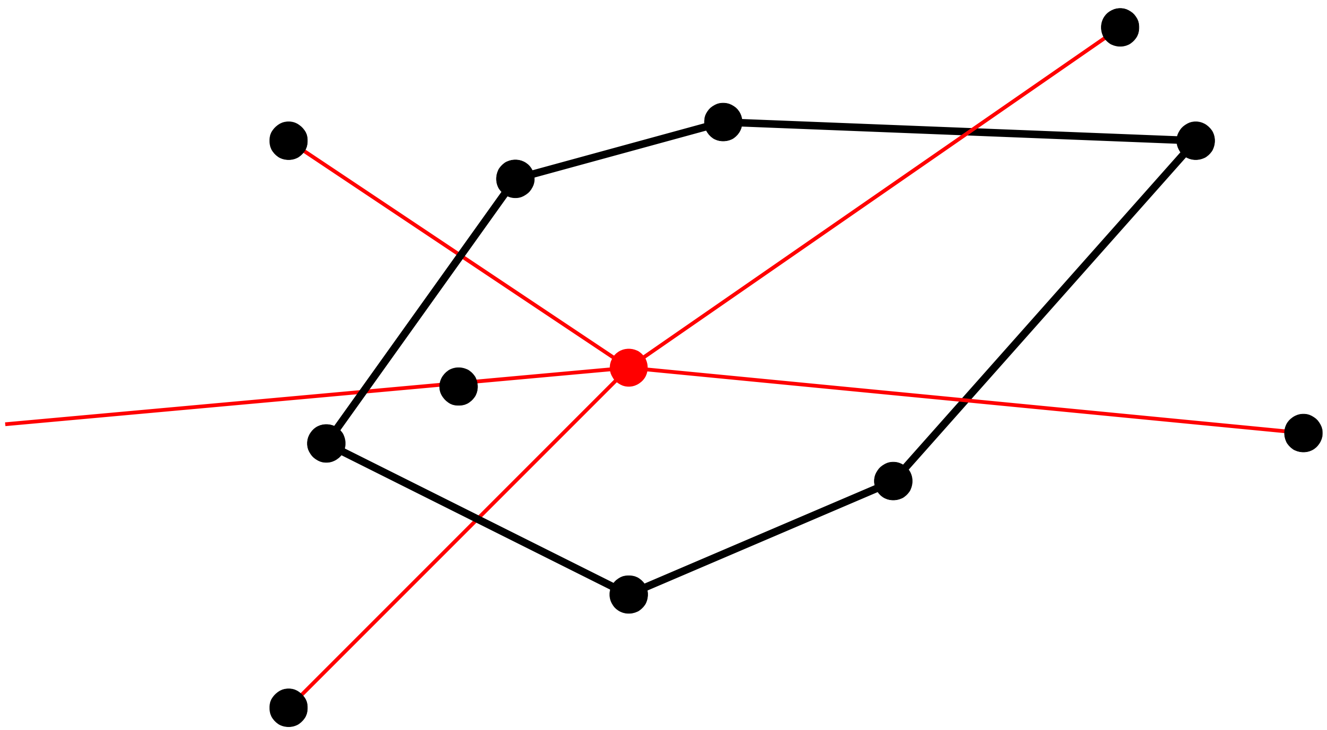
1. Fall: $O(1)$

2. Fall: Anzahl der Strahlen, die eine der beiden zu p_{i+1} inzidenten Seiten von $\mathrm{CH}(p_1, \ldots, p_{i+1})$ schneiden.

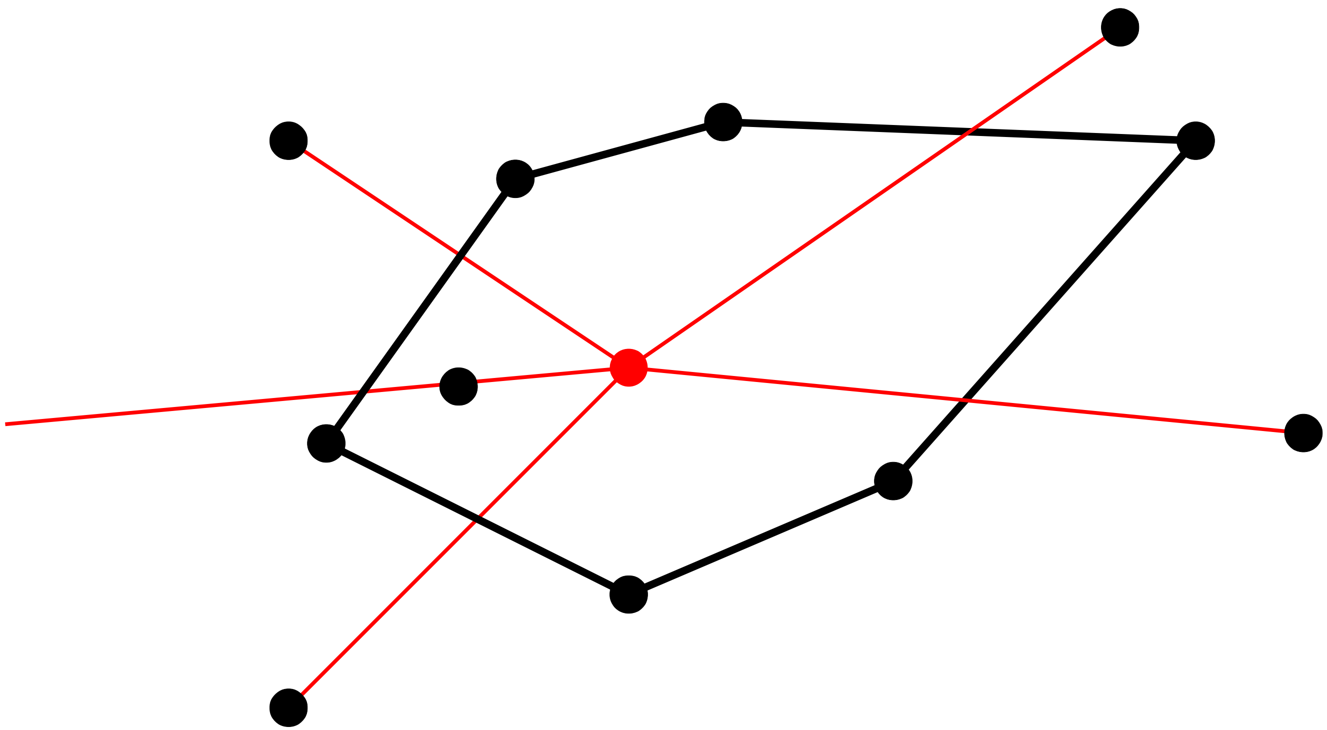
Die interessanten Strahlen im 2. Fall:



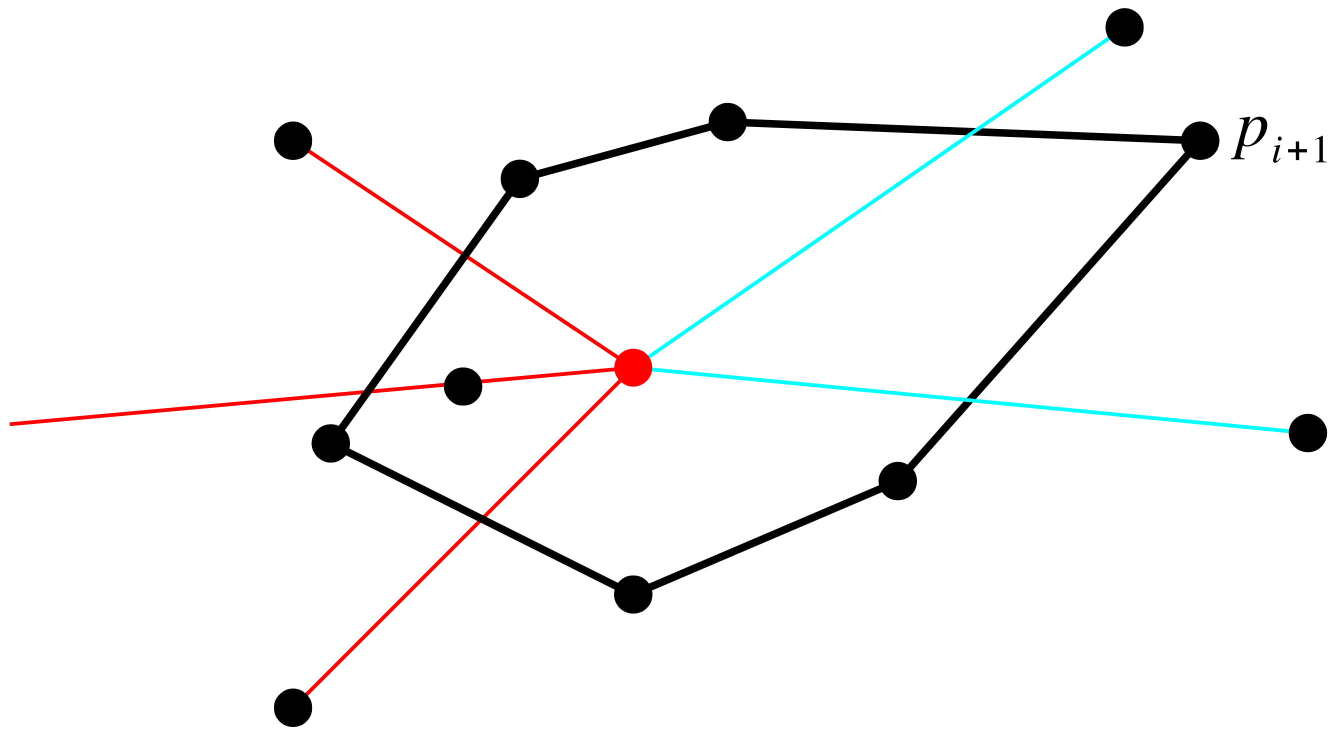
Nach dem Hinzunehmen von p_{i+1} gibt es noch $n-i-1$ Strahlen.

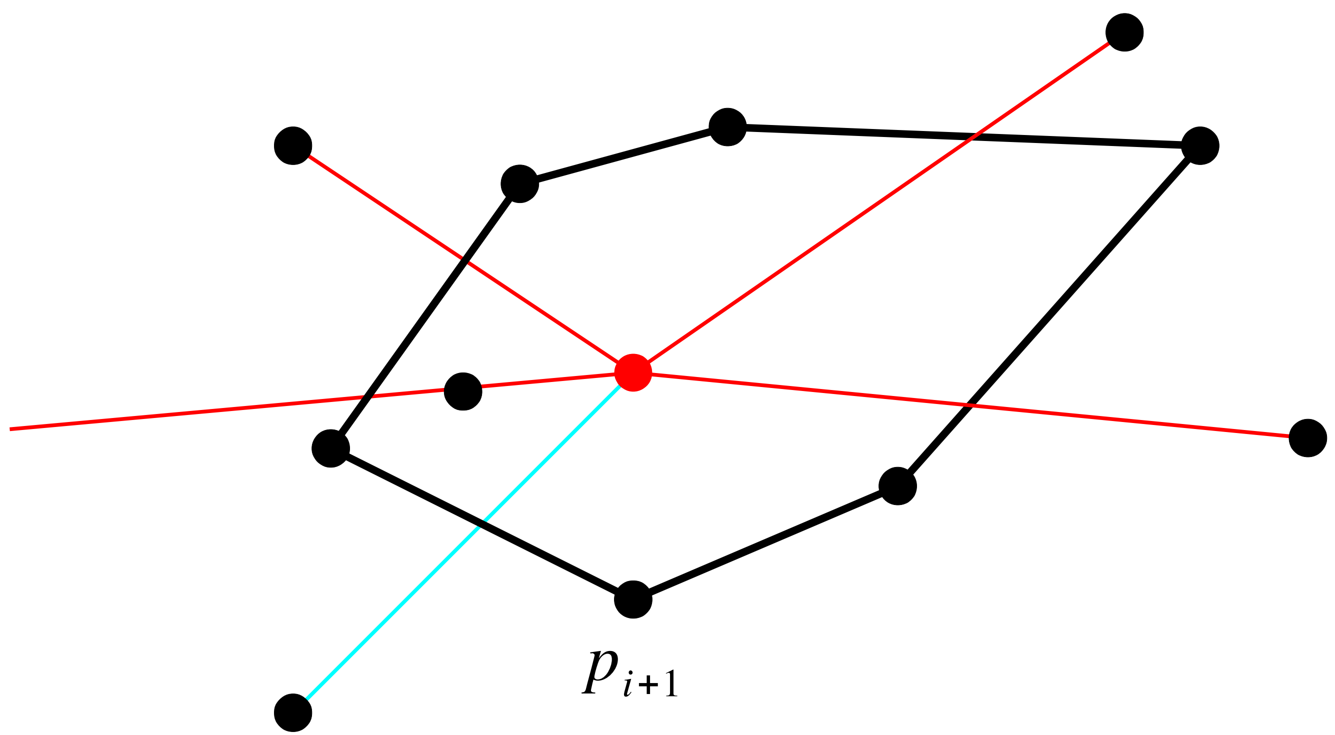


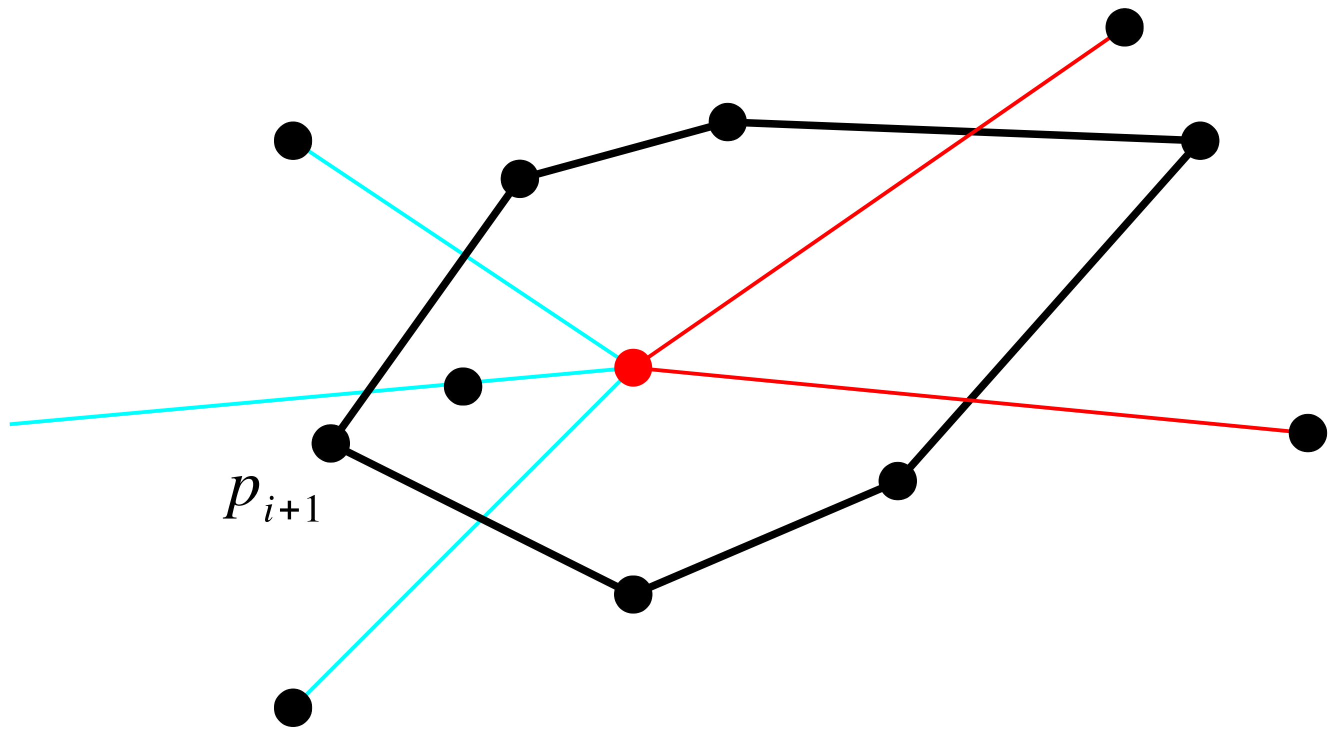
Jeder dieser Strahlen schneidet genau eine Seite von $\mathrm{CH}(p_1, \ldots, p_{i+1})$.



Da wir mit einer zufälligen Reihenfolge der Punkte aus P arbeiten, ist für jeden Punkt aus $\{p_1, \ldots, p_{i+1}\}$ die Wahrscheinlichkeit, an Stelle $i+1$ zu stehen, gleich.




p_{i+1}


p_{i+1}

Zusammenfassung der Analyse der Laufzeit

Erwarteter Aufwand zum Verwalten der Strahlen im 2.Fall:

$$O\left(\frac{2(n-i-1)}{i+1}\right) \subseteq O\left(\frac{2n}{i+1}\right)$$

Das muss jetzt noch aufsummiert werden:

$$\sum_{i=4}^{n} \frac{2n}{i} \leq 2n \sum_{i=1}^{n} \frac{1}{i} \in O(n \log n)$$

Erwartete Laufzeit: $O(n \log n)$

6. Zusammenfassung zur Berechnung der konvexen Hülle

Der deterministische und der randomisierte inkrementelle Algorithmus verarbeiten die einzelnen Punkte auf **ähnliche** Weise (relative Lage von 3 Punkten, Tangenten).

Beide Algorithmen haben eine Laufzeit von **O(*n* log *n*)**.

Beim deterministischen Algorithmus wird die Laufzeit klar durch das **Sortieren** der Punkte bestimmt.

Um einen Aufwand ähnlich wie beim Sortieren kommt man **grundsätzlich nicht** herum.

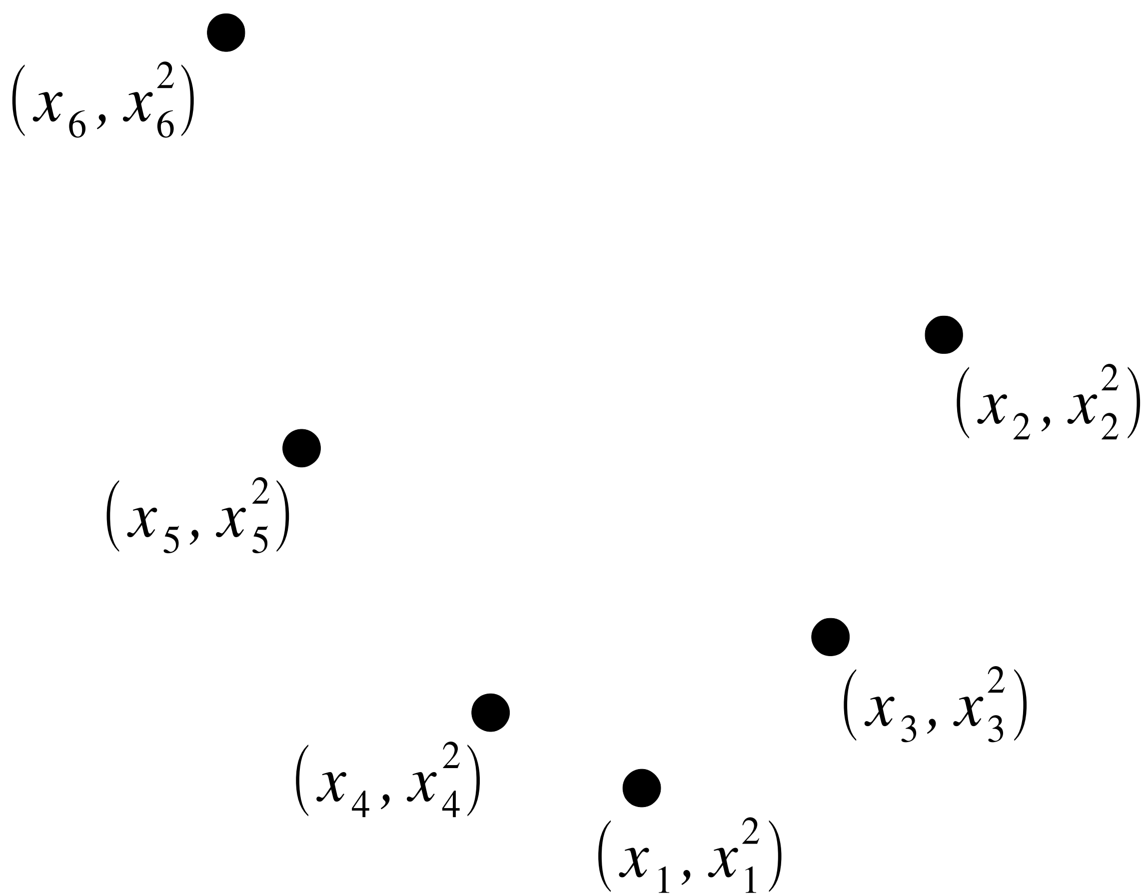
Sortieren steckt in der Berechnung der konvexen Hülle drin:

$$\left(x_1, x_2, \ldots, x_n\right)$$

$$\Downarrow$$

$$\left(\left(x_1, x_1^2\right), \left(x_2, x_2^2\right), \ldots, \left(x_n, x_n^2\right)\right)$$

Grafische Darstellung der sich ergebenden Menge von Punkten:



Aus der (wie auch immer berechneten) konvexen Hülle ergibt sich die sortierte Reihenfolge der x-Werte.