Karlsruher Institut für Technologie

Algorithmen II
Vorlesung am 10.12.2013

Algorithmische Geometrie: Konvexe Hülle

INSTITUT FÜR THEORETISCHE INFORMATIK · PROF. DR. DOROTHEA WAGNER

Organisatorisches

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Klausur

- Anmeldung für die Hauptklausur am 24.02.2014 (14:00 Uhr) ist ab jetzt möglich.
- An- und Abmeldung ist bis zum 17.02.2014 online möglich.
- **Danach ist keine Anmeldung mehr möglich.**
- Nachklausur findet erst im Sommer statt.

Karlsruher Institut für Technologie

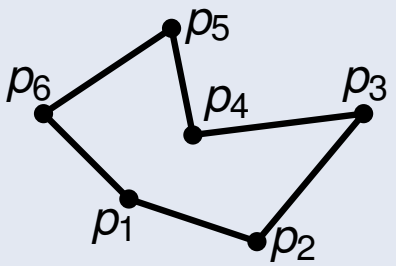
Die konvexe Hülle einer Punktmenge

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Polygone

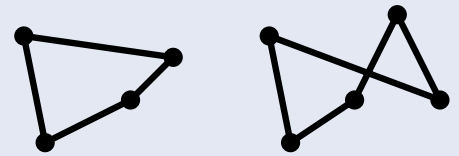
Definition: Polygon **(Definition 6.7)**

Ein *Polygon* ist durch eine Folge von Punkten $\langle p_1, \ldots, p_n \rangle$ gegeben; die p_i sind in der Reihenfolge ihres Auftretens auf dem Rand induziert. $\overline{p_i p_{i+1}}$ heißt (i modulo n)-te Polygonkante.



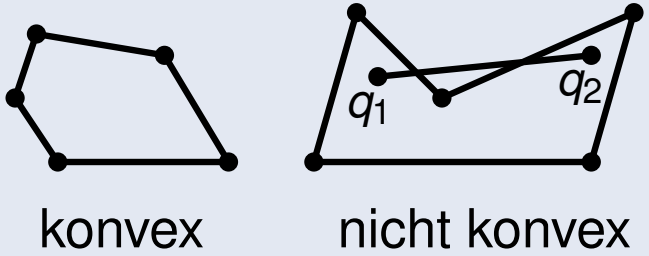
Definition: Einfache Polygone **(Definition 6.8)**

Ein Polygon heißt *einfach*, wenn sich keine zwei Polygonkanten schneiden (außer an gemeinsamen Endpunkten).



Definition: Konvexe Polygone **(Definition 6.9)**

Ein Polygon P heißt *konvex* genau dann wenn für je zwei Punkte q_1, q_2 im Inneren von P gilt, dass die Strecke $\overline{q_1 q_2}$ vollständig im Inneren von P liegt.



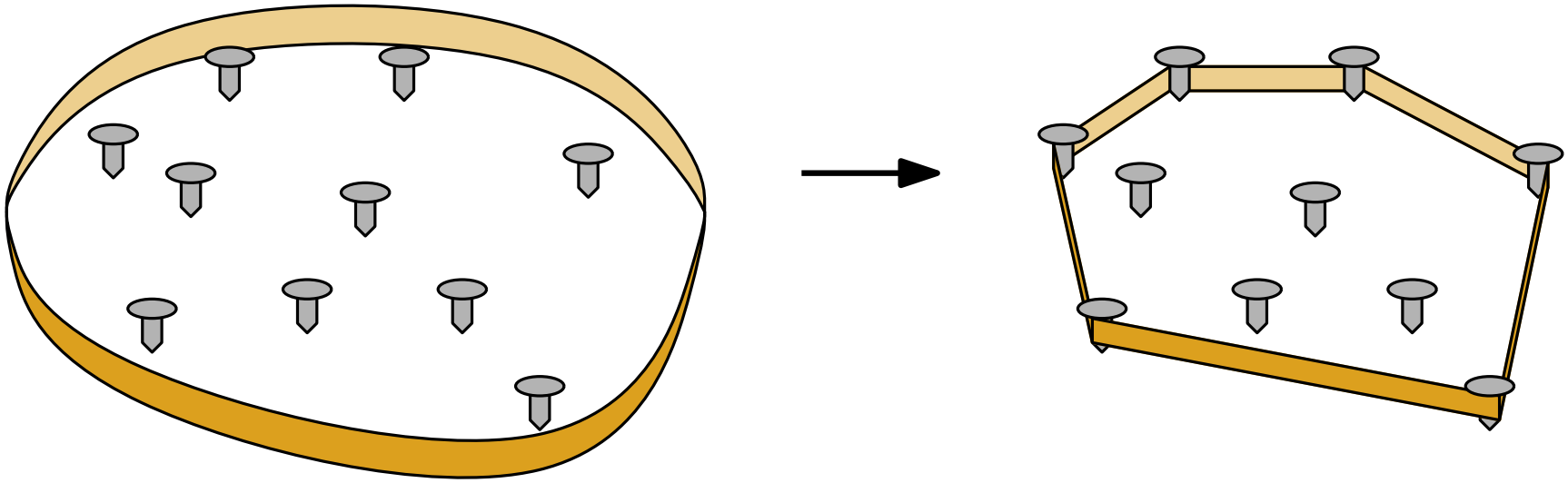
Bemerkung:

Wir zählen die Polygonkanten (also den Rand) zum Inneren eines Polygons.

Konvexe Hülle

Problem: Konvexe Hülle
Gegeben sind $n \geq 3$ Punkte $Q = \{p_0, \ldots, p_{n-1}\}$. Berechne die *konvexe Hülle* $H(Q)$, d.h. das minimale konvexe Polygon, das alle Punkte in Q im Inneren enthält.

Intuition: Fasse jeden Punkte als Nagel auf, ziehe ein Gummiband über alle Nägel und lasse es los.



Wir werden zwei Algorithmen behandeln, die die konvexe Hülle von n Punkten in $O(n \log n)$ bzw. $O(hn)$ bestimmen, wobei h die Anzahl der Eckpunkte der konvexen Hülle ist.

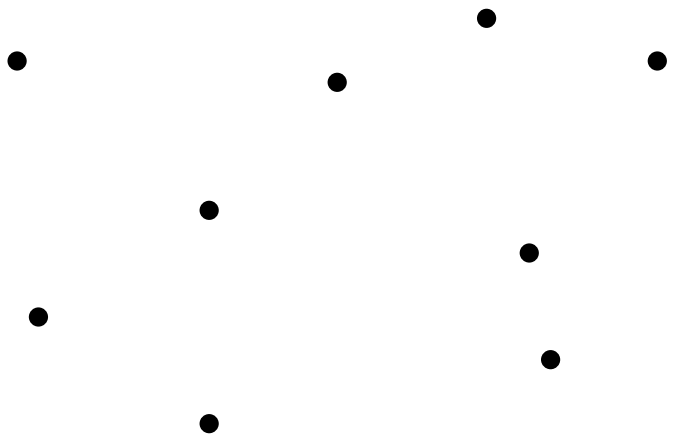
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linkesten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte $p_0, p_1, \ldots, p_{n-1}$ in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

   Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



Zwischenergebnis für die konvexe Hülle:

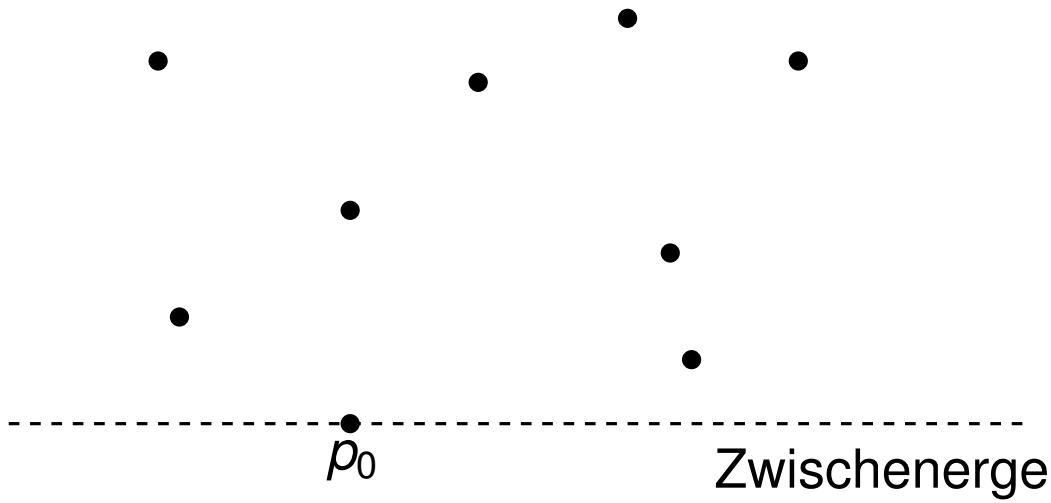
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linkesten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte $p_0, p_1, \ldots, p_{n-1}$ in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

   Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



Zwischenergebnis für die konvexe Hülle: p_0

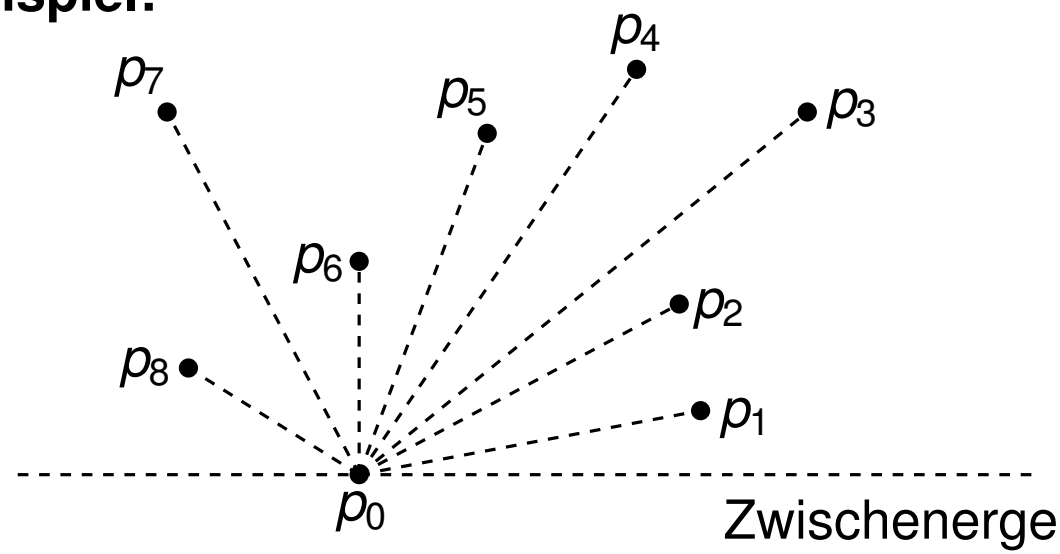
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linkesten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte $p_0, p_1, \ldots, p_{n-1}$ in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

   Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



Zwischenergebnis für die konvexe Hülle: p_0

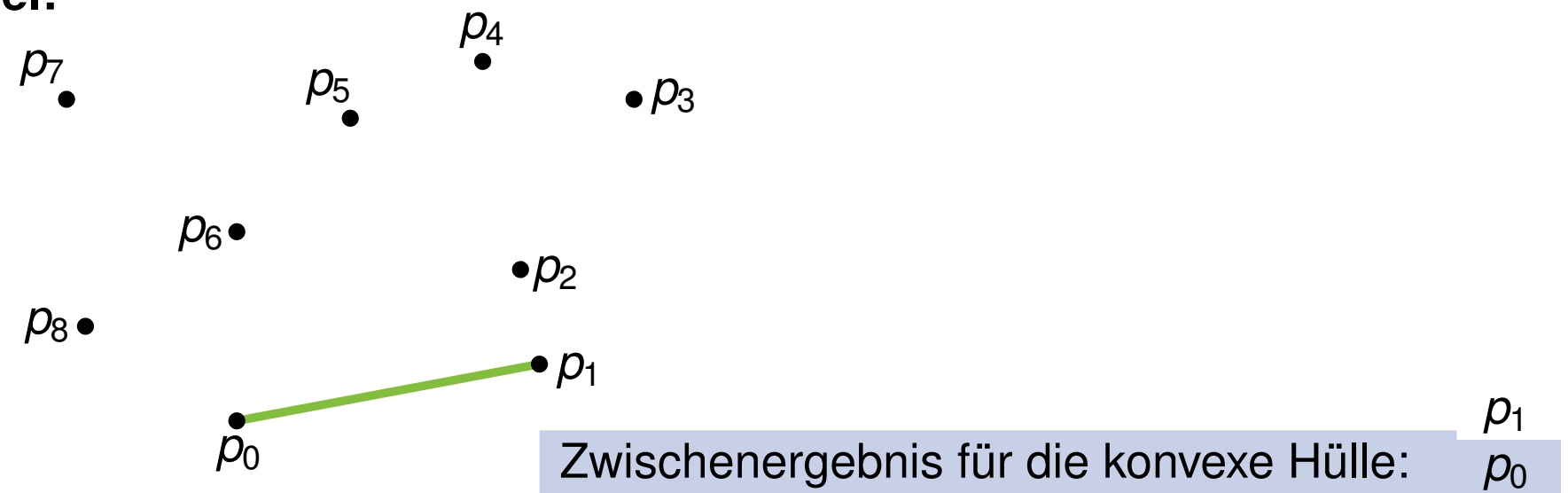
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linkesten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte $p_0, p_1, \ldots, p_{n-1}$ in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

   Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



Zwischenergebnis für die konvexe Hülle: p_1 p_0

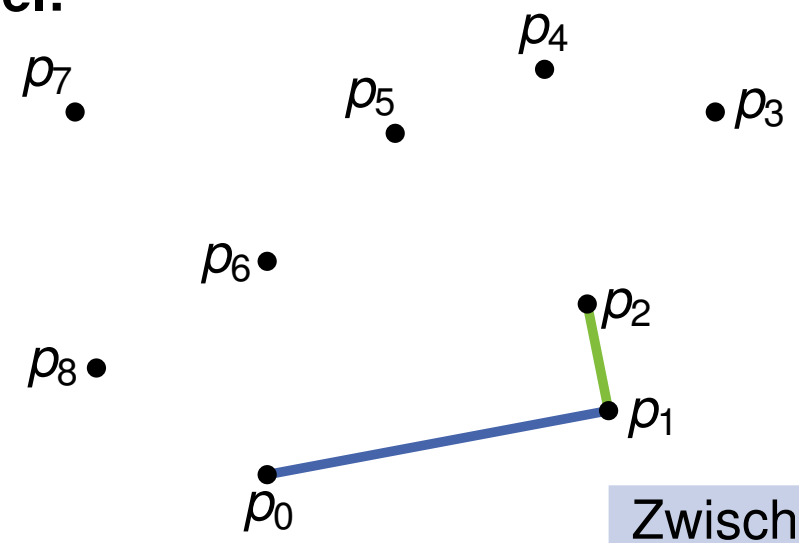
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linkesten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte $p_0, p_1, \ldots, p_{n-1}$ in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

   Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



Zwischenergebnis für die konvexe Hülle: p_2 p_1 p_0

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

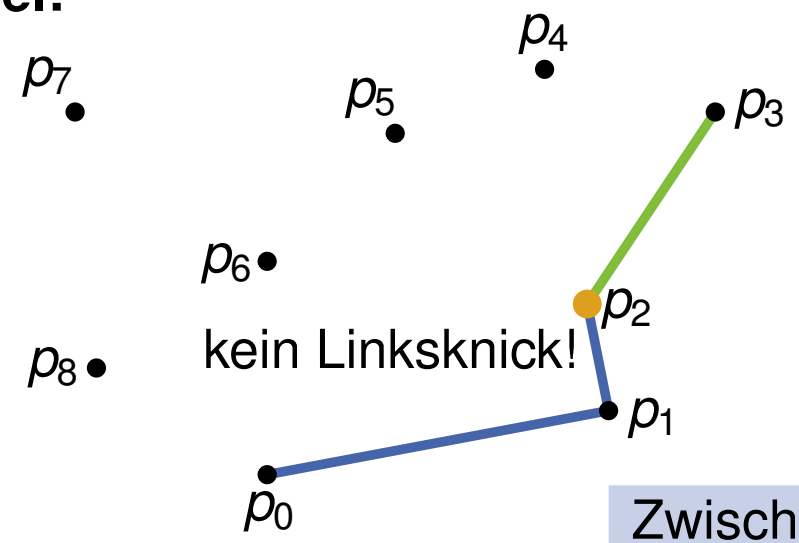
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linkesten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte $p_0, p_1, \ldots, p_{n-1}$ in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

   Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



Zwischenergebnis für die konvexe Hülle:

p_3
p_2
p_1
p_0

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

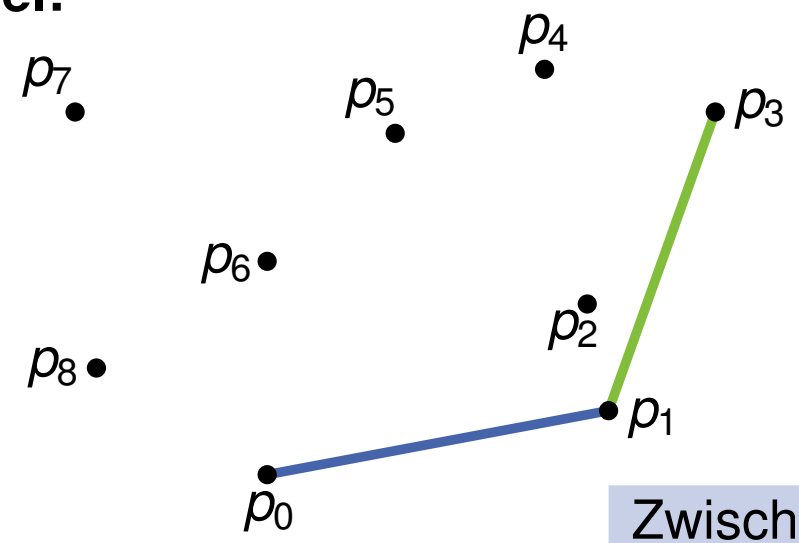
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linkesten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte $p_0, p_1, \ldots, p_{n-1}$ in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

   Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



Zwischenergebnis für die konvexe Hülle:

p_3
p_1
p_0

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

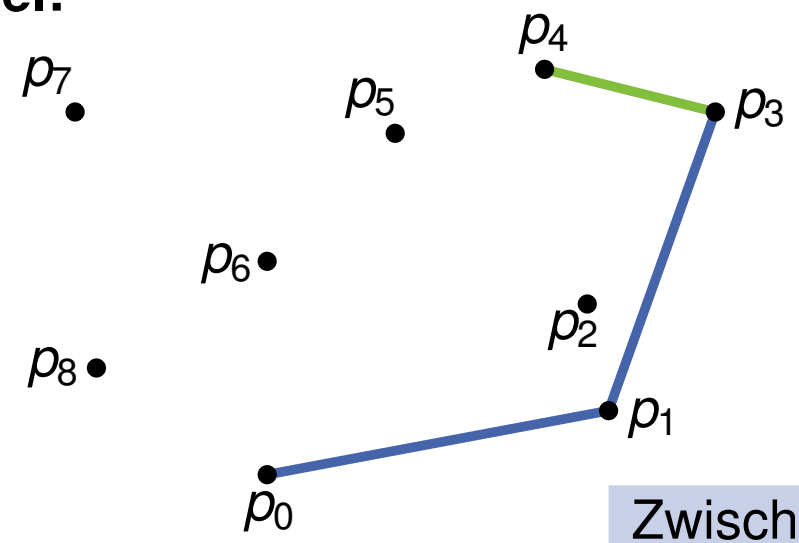
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linkesten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte $p_0, p_1, \ldots, p_{n-1}$ in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

   Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



Zwischenergebnis für die konvexe Hülle:

p_4
p_3
p_1
p_0

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

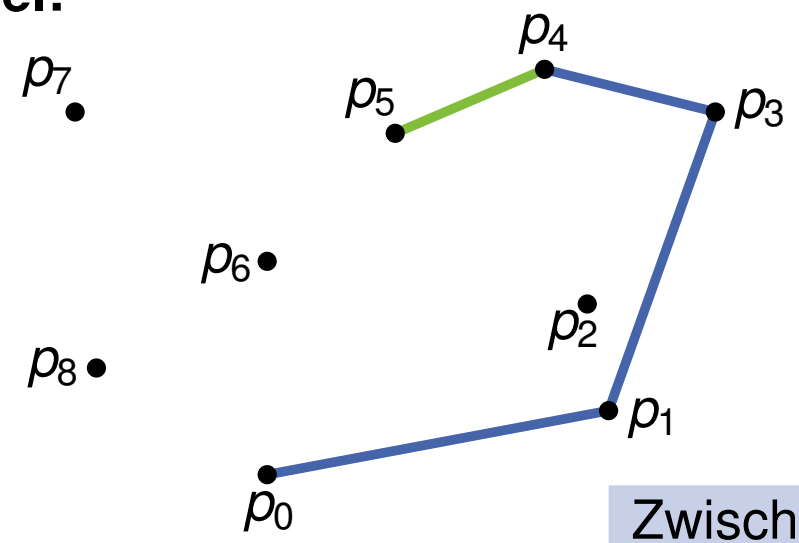
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linkesten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte $p_0, p_1, \ldots, p_{n-1}$ in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

   Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



Zwischenergebnis für die konvexe Hülle:

p_5
p_4
p_3
p_1
p_0

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

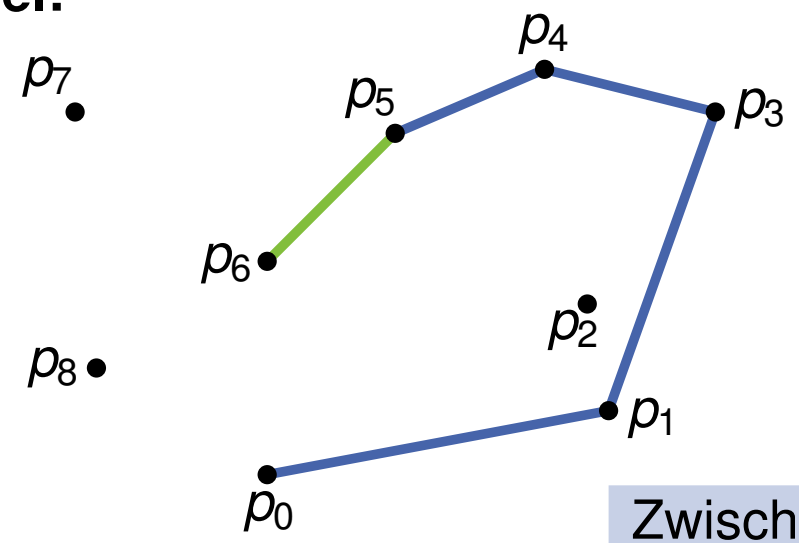
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linkesten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte $p_0, p_1, \ldots, p_{n-1}$ in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

   Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



Zwischenergebnis für die konvexe Hülle:

p_6
p_5
p_4
p_3
p_1
p_0

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

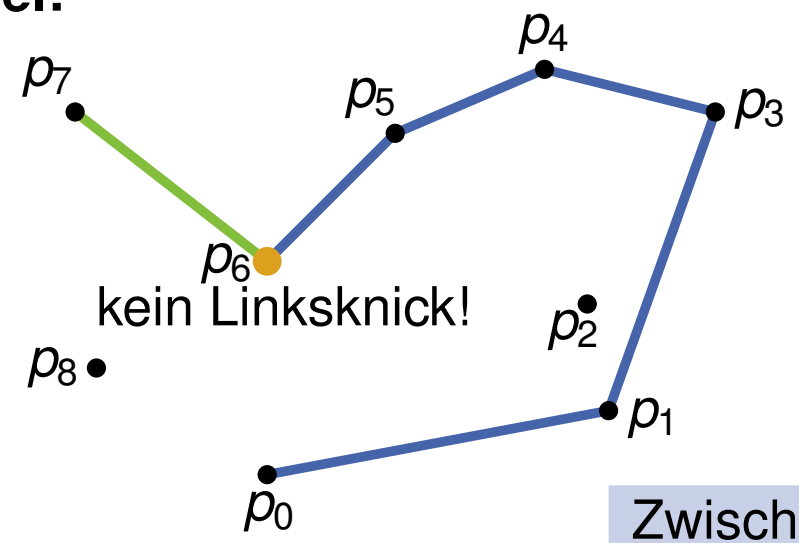
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linkesten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte $p_0, p_1, \ldots, p_{n-1}$ in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

   Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



Zwischenergebnis für die konvexe Hülle:

p_7
p_6
p_5
p_4
p_3
p_1
p_0

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

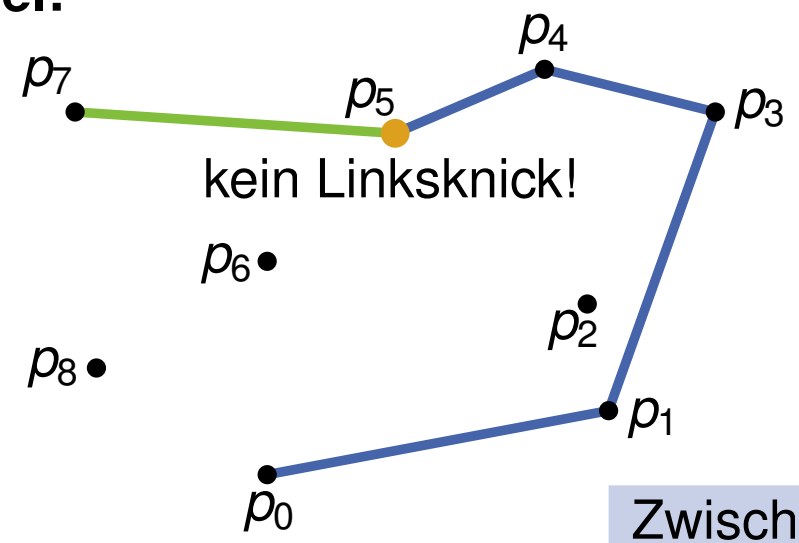
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linkesten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte $p_0, p_1, \ldots, p_{n-1}$ in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

   Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



Zwischenergebnis für die konvexe Hülle:

p_7
p_5
p_4
p_3
p_1
p_0

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

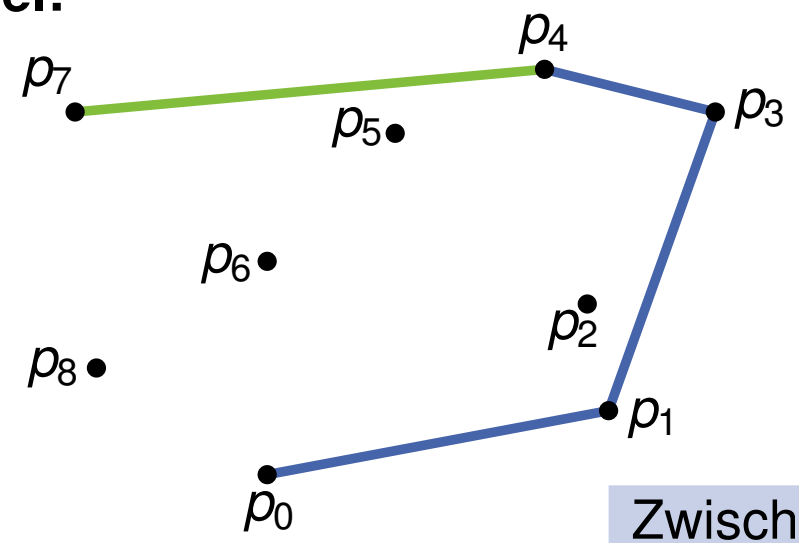
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linkesten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte $p_0, p_1, \ldots, p_{n-1}$ in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

   Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



Zwischenergebnis für die konvexe Hülle:

p_7
p_4
p_3
p_1
p_0

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

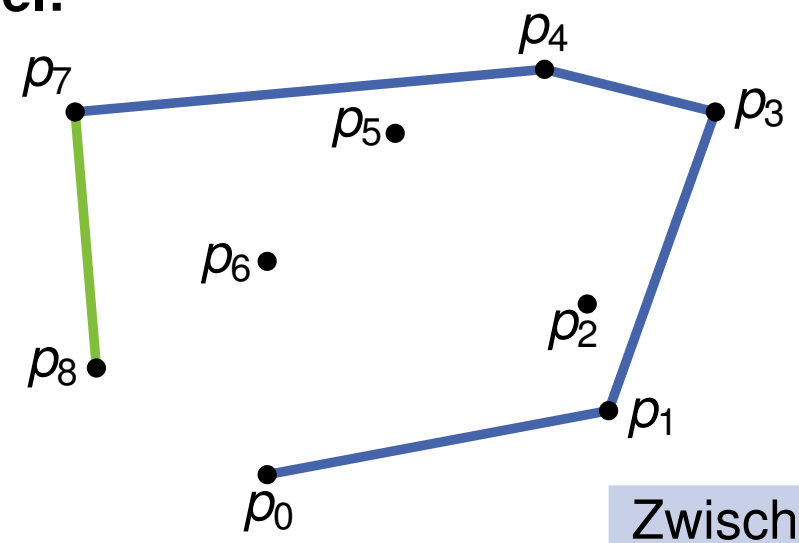
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linkesten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte $p_0, p_1, \ldots, p_{n-1}$ in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

   Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



Zwischenergebnis für die konvexe Hülle:

p_8
p_7
p_4
p_3
p_1
p_0

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

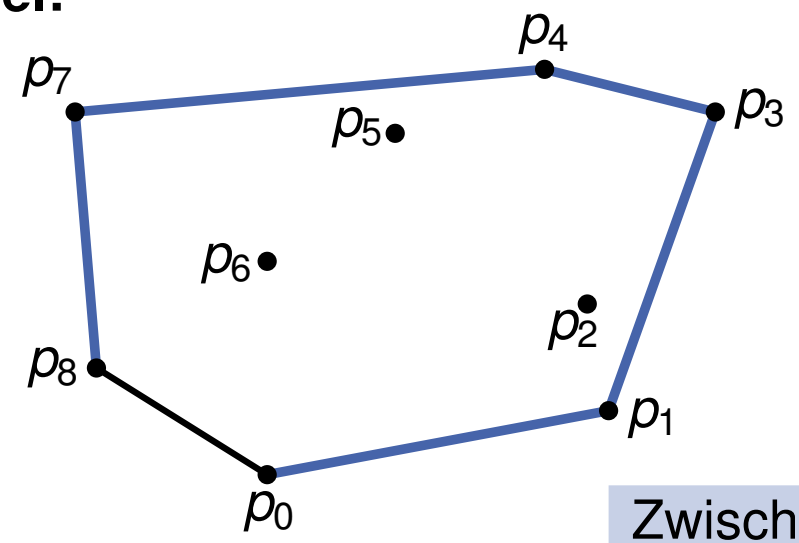
Der Graham Scan (1972)

Idee:

1. Bestimme untersten Punkte p_0 (falls nicht eindeutig, dann den Linkesten).
2. Ordne die restlichen Punkte p_i nach dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$.
3. Durchlaufe die Punkte $p_0, p_1, \ldots, p_{n-1}$ in dieser Reihenfolge und konstruiere sukzessive die konvexe Hülle $H(Q)$ nach folgender Regel:

   Wird beim Durchlaufen der Strecken $\overline{p_i p_j}$ und $\overline{p_j p_k}$ nicht links abgebogen, so ist p_j nicht aus dem Rand von $H(Q)$.

Beispiel:



Zwischenergebnis für die konvexe Hülle:

p_8
p_7
p_4
p_3
p_1
p_0

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Der Graham Scan (1972)

GRAHAM SCAN(Q)

$S \leftarrow$ leerer Stack — Initialisierung $O(n \log n)$

$p_0 \leftarrow$ Punkt in Q mit kleinster y-Koord. (falls uneindeutig, den mit kleinster x-Koord.)

$\langle p_1, \ldots, p_m \rangle \leftarrow$ Die restlichen Punkte sortiert gemäß dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$
(bei mehreren mit gleichem Winkel entferne alle bis auf den Entferntesten).

PUSH(p_0, S), PUSH(p_1, S), PUSH(p_2, S)

eigentlicher Algorithmus

for $i = 3$ **to** m **do**
  while *Strecken* $\overline{\text{NEXTTOTOP}(S)\text{TOP}(S)}$ *und* $\overline{\text{TOP}(S)p_i}$ *bilden keinen Linksknick* **do**
    POP(S)
  PUSH(p_i,S)
gib S aus

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Der Graham Scan (1972)

GRAHAM SCAN(Q)

$S \leftarrow$ leerer Stack — Initialisierung $O(n \log n)$

$p_0 \leftarrow$ Punkt in Q mit kleinster y-Koord. (falls uneindeutig, den mit kleinster x-Koord.)

$\langle p_1, \ldots, p_m \rangle \leftarrow$ Die restlichen Punkte sortiert gemäß dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$
(bei mehreren mit gleichem Winkel entferne alle bis auf den Entferntesten).

PUSH(p_0, S), PUSH(p_1, S), PUSH(p_2, S)

for $i = 3$ **to** m **do** — eigentlicher Algorithmus

  while *Strecken* $\overline{\text{NEXTTOTOP}(S)\text{TOP}(S)}$ *und* $\overline{\text{TOP}(S)p_i}$ *bilden keinen Linksknick* **do**
    POP(S) — amortisiert $O(1)$

  PUSH(p_i,S)

gib S aus

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Der Graham Scan (1972)

GRAHAM SCAN(Q) $O(n \log n)$

$S \leftarrow$ leerer Stack — Initialisierung $O(n \log n)$

$p_0 \leftarrow$ Punkt in Q mit kleinster y-Koord. (falls uneindeutig, den mit kleinster x-Koord.)

$\langle p_1, \ldots, p_m \rangle \leftarrow$ Die restlichen Punkte sortiert gemäß dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$
(bei mehreren mit gleichem Winkel entferne alle bis auf den Entferntesten).

PUSH(p_0, S), PUSH(p_1, S), PUSH(p_2, S)

for $i = 3$ **to** m **do** — eigentlicher Algorithmus $O(n)$

  while *Strecken* $\overline{\text{NEXTTOTOP}(S)\text{TOP}(S)}$ *und* $\overline{\text{TOP}(S)p_i}$ *bilden keinen Linksknick* **do** — amortisiert $O(1)$

    POP(S)

  PUSH(p_i,S)

gib S aus

Satz: Graham Scan
Der Graham Scan berechnet die konvexe Hülle $H(Q)$ von Q in $O(n \log n)$ Zeit.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Der Graham Scan (1972)

GRAHAM SCAN(Q) $O(n \log n)$

$S \leftarrow$ leerer Stack — Initialisierung $O(n \log n)$

$p_0 \leftarrow$ Punkt in Q mit kleinster y-Koord. (falls uneindeutig, den mit kleinster x-Koord.)

$\langle p_1, \ldots, p_m \rangle \leftarrow$ Die restlichen Punkte sortiert gemäß dem Winkel zwischen der Horizontalen durch p_0 und $\overline{p_0 p_i}$
(bei mehreren mit gleichem Winkel entferne alle bis auf den Entferntesten).

PUSH(p_0, S), PUSH(p_1, S), PUSH(p_2, S)

for $i = 3$ **to** m **do** — eigentlicher Algorithmus $O(n)$

  while *Strecken* $\overline{\text{NEXTTOTOP}(S)\text{TOP}(S)}$ *und* $\overline{\text{TOP}(S)p_i}$ *bilden keinen Linksknick* **do** — amortisiert $O(1)$

    POP(S)

  PUSH(p_i,S)

gib S aus

Satz: Graham Scan
Der Graham Scan berechnet die konvexe Hülle $H(Q)$ von Q in $O(n \log n)$ Zeit.

Beweis: Zeige die folgenden beiden Invarianten:

1. Wird ein Punkt von S entfernt (POP(S)), so ist er nicht Eckpunkt von $H(Q)$.
2. Der aktuelle Inhalt von S bestimmt stets ein konvexes Polygon.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Graham Scan – Korrektheit

```
...
for k = 3 to m do
    while Strecken NextToTop(S)Top(S) und Top(S)p_k bilden keinen Linksknick do
        Pop(S)
    Push(p_k, S)
gib S aus
```

Satz: Graham Scan
Der Graham Scan berechnet die konvexe Hülle $H(Q)$ von Q in $O(n \log n)$ Zeit.

Beweis: Zeige die folgenden beiden Invarianten:

1. Wird ein Punkt von S entfernt (Pop(S)), so ist er nicht Eckpunkt von $H(Q)$.
2. Der aktuelle Inhalt von S bestimmt stets ein konvexes Polygon.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Graham Scan – Korrektheit

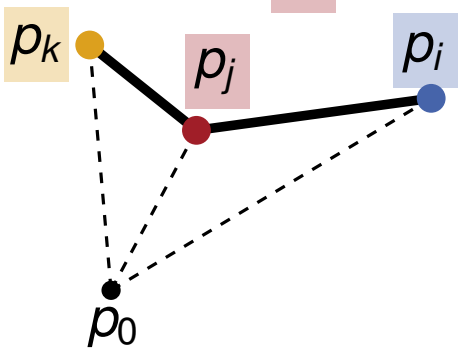
```
...
for k = 3 to m do
    while Strecken NextToTop(S)Top(S) und Top(S)p_k bilden keinen Linksknick do
        Pop(S)                    (p_i)        (p_j)
    Push(p_k, S)
gib S aus
```

Satz: Graham Scan
Der Graham Scan berechnet die konvexe Hülle $H(Q)$ von Q in $O(n \log n)$ Zeit.

Beweis: Zeige die folgenden beiden Invarianten:

1. Wird ein Punkt von S entfernt ($\textsc{Pop}(S)$), so ist er nicht Eckpunkt von $H(Q)$.
2. Der aktuelle Inhalt von S bestimmt stets ein konvexes Polygon.

1. Invariante: p_j wird entfernt; zeige: p_j ist kein Eckpunkt von $H(Q)$



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Graham Scan – Korrektheit

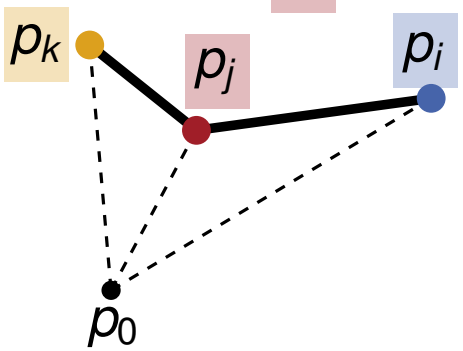
```
...
for k = 3 to m do
    while Strecken NextToTop(S)Top(S) und Top(S)p_k bilden keinen Linksknick do
        Pop(S)                    (p_i)        (p_j)
    Push(p_k, S)
gib S aus
```

Satz: Graham Scan
Der Graham Scan berechnet die konvexe Hülle $H(Q)$ von Q in $O(n \log n)$ Zeit.

Beweis: Zeige die folgenden beiden Invarianten:

1. Wird ein Punkt von S entfernt ($\textsc{Pop}(S)$), so ist er nicht Eckpunkt von $H(Q)$.
2. Der aktuelle Inhalt von S bestimmt stets ein konvexes Polygon.

1. Invariante: p_j wird entfernt; zeige: p_j ist kein Eckpunkt von $H(Q)$



- In der Sortierung um p_0 liegt p_j zwischen p_i und p_k.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Graham Scan – Korrektheit

```
...
for k = 3 to m do
    while Strecken NextToTop(S)Top(S) und Top(S)p_k bilden keinen Linksknick do
        Pop(S)
    Push(p_k, S)
gib S aus
```

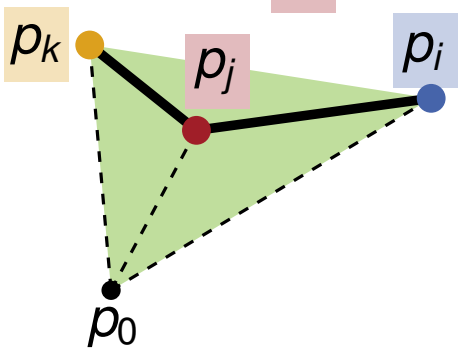
(NextToTop(S) = p_i, Top(S) = p_j)

Satz: Graham Scan
Der Graham Scan berechnet die konvexe Hülle $H(Q)$ von Q in $O(n \log n)$ Zeit.

Beweis: Zeige die folgenden beiden Invarianten:

1. Wird ein Punkt von S entfernt ($\textsc{Pop}(S)$), so ist er nicht Eckpunkt von $H(Q)$.
2. Der aktuelle Inhalt von S bestimmt stets ein konvexes Polygon.

1. Invariante: p_j wird entfernt; zeige: p_j ist kein Eckpunkt von $H(Q)$



- In der Sortierung um p_0 liegt p_j zwischen p_i und p_k.
- p_j liegt innerhalb des Dreiecks $p_0 p_i p_k$ (oder auf dem Rand)

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Graham Scan – Korrektheit

```
...
for k = 3 to m do
    while Strecken NextToTop(S)Top(S) und Top(S)p_k bilden keinen Linksknick do
        Pop(S)
    Push(p_k, S)
gib S aus
```

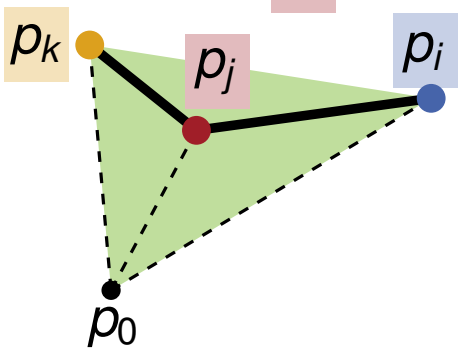
(NextToTop(S) = p_i, Top(S) = p_j)

Satz: Graham Scan
Der Graham Scan berechnet die konvexe Hülle $H(Q)$ von Q in $O(n \log n)$ Zeit.

Beweis: Zeige die folgenden beiden Invarianten:

1. Wird ein Punkt von S entfernt ($\textsc{Pop}(S)$), so ist er nicht Eckpunkt von $H(Q)$. ✓
2. Der aktuelle Inhalt von S bestimmt stets ein konvexes Polygon.

1. Invariante: p_j wird entfernt; zeige: p_j ist kein Eckpunkt von $H(Q)$



- In der Sortierung um p_0 liegt p_j zwischen p_i und p_k.
- p_j liegt innerhalb des Dreiecks $p_0 p_i p_k$ (oder auf dem Rand)
- $\Rightarrow$ p_j ist kein Eckpunkt von $H(Q)$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Graham Scan – Korrektheit

```
...
for k = 3 to m do
    while Strecken NextToTop(S)Top(S) und Top(S)p_k bilden keinen Linksknick do
        Pop(S)
    Push(p_k, S)
gib S aus
```

(NextToTop(S) = p_i, Top(S) = p_j)

Satz: Graham Scan
Der Graham Scan berechnet die konvexe Hülle $H(Q)$ von Q in $O(n \log n)$ Zeit.

Beweis: Zeige die folgenden beiden Invarianten:

1. Wird ein Punkt von S entfernt (Pop(S)), so ist er nicht Eckpunkt von $H(Q)$. ✓
2. Der aktuelle Inhalt von S bestimmt stets ein konvexes Polygon.

2. Invariante: Dreieck $p_0 p_1 p_2$ ist konvex; zeige: Pop und Push erhalten Konvexität

Graham Scan – Korrektheit

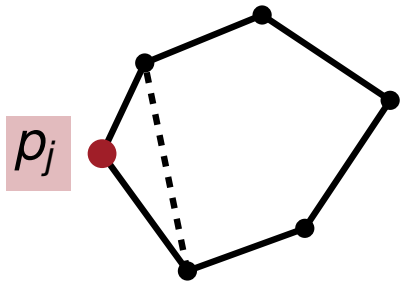
```
...
for k = 3 to m do
    while Strecken NextToTop(S)Top(S) und Top(S)p_k bilden keinen Linksknick do
        Pop(S)                    p_i          p_j
    Push(p_k, S)
gib S aus
```

Satz: Graham Scan
Der Graham Scan berechnet die konvexe Hülle $H(Q)$ von Q in $O(n \log n)$ Zeit.

Beweis: Zeige die folgenden beiden Invarianten:

1. Wird ein Punkt von S entfernt (Pop(S)), so ist er nicht Eckpunkt von $H(Q)$.
2. Der aktuelle Inhalt von S bestimmt stets ein konvexes Polygon.

2. Invariante: Dreieck $p_0 p_1 p_2$ ist konvex; zeige: Pop und Push erhalten Konvexität



- Löschen eines Punkts erhält offensichtlich die Konvexität.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Graham Scan – Korrektheit

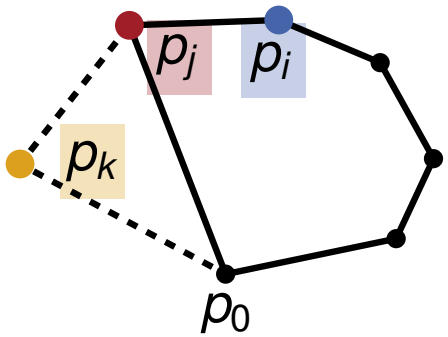
```
...
for k = 3 to m do
    while Strecken NextToTop(S)Top(S) und Top(S)p_k bilden keinen Linksknick do
        Pop(S)
    Push(p_k, S)
gib S aus
```

Satz: Graham Scan
Der Graham Scan berechnet die konvexe Hülle $H(Q)$ von Q in $O(n \log n)$ Zeit.

Beweis: Zeige die folgenden beiden Invarianten:

1. Wird ein Punkt von S entfernt (Pop(S)), so ist er nicht Eckpunkt von $H(Q)$.
2. Der aktuelle Inhalt von S bestimmt stets ein konvexes Polygon.

2. Invariante: Dreieck $p_0 p_1 p_2$ ist konvex; zeige: Pop und Push erhalten Konvexität



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Graham Scan – Korrektheit

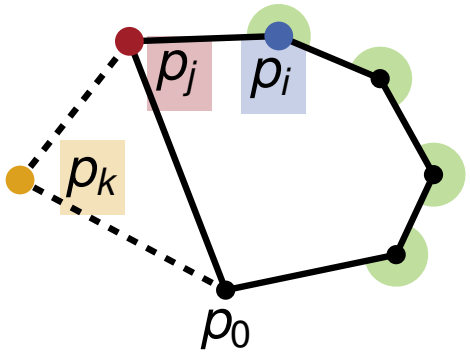
```
...
for k = 3 to m do
    while Strecken NextToTop(S)Top(S) und Top(S)p_k bilden keinen Linksknick do
        Pop(S)
    Push(p_k, S)
gib S aus
```

Satz: Graham Scan
Der Graham Scan berechnet die konvexe Hülle $H(Q)$ von Q in $O(n \log n)$ Zeit.

Beweis: Zeige die folgenden beiden Invarianten:

1. Wird ein Punkt von S entfernt (Pop(S)), so ist er nicht Eckpunkt von $H(Q)$.
2. Der aktuelle Inhalt von S bestimmt stets ein konvexes Polygon.

2. Invariante: Dreieck $p_0 p_1 p_2$ ist konvex; zeige: Pop und Push erhalten Konvexität



- Polygon war vorher konvex $\Rightarrow$ konvexe Winkel bei $p_1 \ldots p_i$

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Graham Scan – Korrektheit

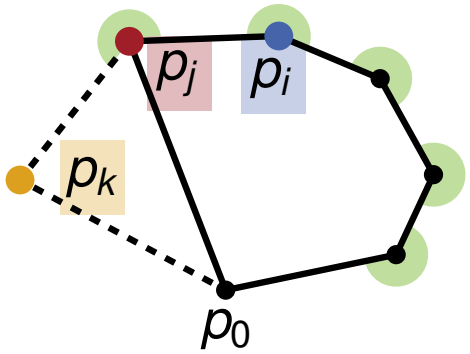
```
...
for k = 3 to m do
    while Strecken NextToTop(S)Top(S) und Top(S)p_k bilden keinen Linksknick do
        Pop(S)
    Push(p_k, S)
gib S aus
```

Satz: Graham Scan
Der Graham Scan berechnet die konvexe Hülle $H(Q)$ von Q in $O(n \log n)$ Zeit.

Beweis: Zeige die folgenden beiden Invarianten:

1. Wird ein Punkt von S entfernt (Pop(S)), so ist er nicht Eckpunkt von $H(Q)$.
2. Der aktuelle Inhalt von S bestimmt stets ein konvexes Polygon.

2. Invariante: Dreieck $p_0 p_1 p_2$ ist konvex; zeige: Pop und Push erhalten Konvexität



- Polygon war vorher konvex $\Rightarrow$ konvexe Winkel bei $p_1 \ldots p_i$
- Konvexer Winkel bei p_j (sonst wäre Pop aufgerufen worden)

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Graham Scan – Korrektheit

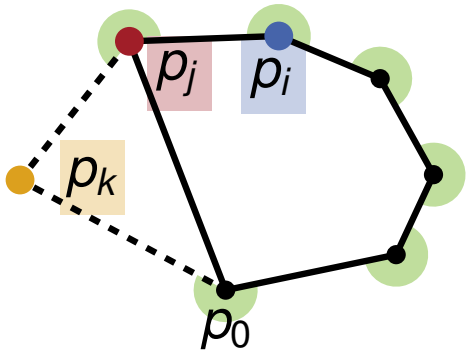
```
...
for k = 3 to m do
    while Strecken NextToTop(S)Top(S) und Top(S)p_k bilden keinen Linksknick do
        Pop(S)
    Push(p_k, S)
gib S aus
```

Satz: Graham Scan
Der Graham Scan berechnet die konvexe Hülle $H(Q)$ von Q in $O(n \log n)$ Zeit.

Beweis: Zeige die folgenden beiden Invarianten:

1. Wird ein Punkt von S entfernt (Pop(S)), so ist er nicht Eckpunkt von $H(Q)$.
2. Der aktuelle Inhalt von S bestimmt stets ein konvexes Polygon.

2. Invariante: Dreieck $p_0 p_1 p_2$ ist konvex; zeige: Pop und Push erhalten Konvexität



- Polygon war vorher konvex $\Rightarrow$ konvexe Winkel bei $p_1 \ldots p_i$
- Konvexer Winkel bei p_j (sonst wäre Pop aufgerufen worden)
- Konvexer Winkel bei p_0 (da p_0 tiefster Punkt)

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Graham Scan – Korrektheit

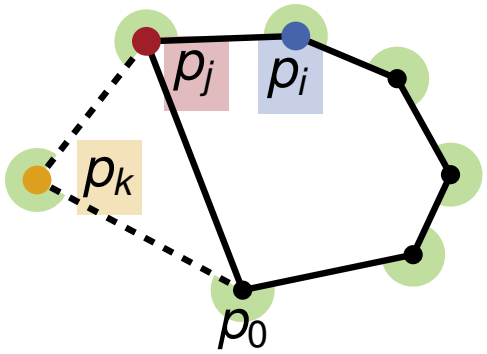
```
...
for k = 3 to m do
    while Strecken NextToTop(S)Top(S) und Top(S)p_k bilden keinen Linksknick do
        Pop(S)                          p_i          p_j
    Push(p_k, S)
gib S aus
```

Satz: Graham Scan
Der Graham Scan berechnet die konvexe Hülle $H(Q)$ von Q in $O(n \log n)$ Zeit.

Beweis: Zeige die folgenden beiden Invarianten:

1. Wird ein Punkt von S entfernt (Pop(S)), so ist er nicht Eckpunkt von $H(Q)$.
2. Der aktuelle Inhalt von S bestimmt stets ein konvexes Polygon.

2. Invariante: Dreieck $p_0 p_1 p_2$ ist konvex; zeige: Pop und Push erhalten Konvexität



- Polygon war vorher konvex $\Rightarrow$ konvexe Winkel bei $p_1 \ldots p_i$
- Konvexer Winkel bei p_j (sonst wäre Pop aufgerufen worden)
- Konvexer Winkel bei p_0 (da p_0 tiefster Punkt)
- Konvexer Winkel bei p_k (p_k kommt nach p_j in Sortierung um p_0)

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Graham Scan – Korrektheit

```
...
for k = 3 to m do
    while Strecken NextToTop(S)Top(S) und Top(S)p_k bilden keinen Linksknick do
        Pop(S)
    Push(p_k, S)
gib S aus
```

Satz: Graham Scan
Der Graham Scan berechnet die konvexe Hülle $H(Q)$ von Q in $O(n \log n)$ Zeit.

Beweis: Zeige die folgenden beiden Invarianten:

1. Wird ein Punkt von S entfernt (Pop(S)), so ist er nicht Eckpunkt von $H(Q)$.
2. Der aktuelle Inhalt von S bestimmt stets ein konvexes Polygon.

Beachte: Jeder Punkt wird einmal auf den Stack gelegt.

1. Invariante $\Rightarrow$ Das resultierend Polygon enthält alle Eckpunkte von $H(Q)$.

2. Invariante $\Rightarrow$ Das resultierend Polygon ist konvex.

$\Rightarrow$ Das resultierende Polygon ist $H(Q)$.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Eine untere Schranke

Satz: Untere Schranke

Die Laufzeit zur Berechnung der konvexen Hülle einer Menge von n Punkten ist in $\Theta(n \log n)$. (d.h. es gibt keinen Algorithmus mit Laufzeit $o(n \log n)$)

Eine untere Schranke

Satz: Untere Schranke

Die Laufzeit zur Berechnung der konvexen Hülle einer Menge von n Punkten ist in $\Theta(n \log n)$. (d.h. es gibt keinen Algorithmus mit Laufzeit $o(n \log n)$)

Beweis:

Annahme: Die konvexe Hülle von n Punkten kann in $T(n)$ Zeit berechnet werden.

Zeige: n Zahlen $a_1, \ldots, a_n$ können in $T(n) + O(n)$ Zeit sortiert werden.

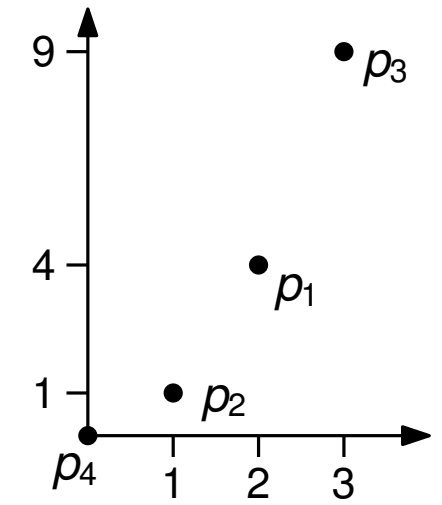
Eine untere Schranke

Satz: Untere Schranke
Die Laufzeit zur Berechnung der konvexen Hülle einer Menge von n Punkten ist in $\Theta(n \log n)$. (d.h. es gibt keinen Algorithmus mit Laufzeit $o(n \log n)$)

Beweis:
Annahme: Die konvexe Hülle von n Punkten kann in $T(n)$ Zeit berechnet werden.
Zeige: n Zahlen $a_1, \ldots, a_n$ können in $T(n) + O(n)$ Zeit sortiert werden.

- Betrachte Punktmenge $P = \{p_1, \ldots, p_n\}$ mit $p_i = (a_i, a_i^2)$

Beispiel: $a_1 = 2$, $a_2 = 1$, $a_3 = 3$, $a_4 = 0$



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

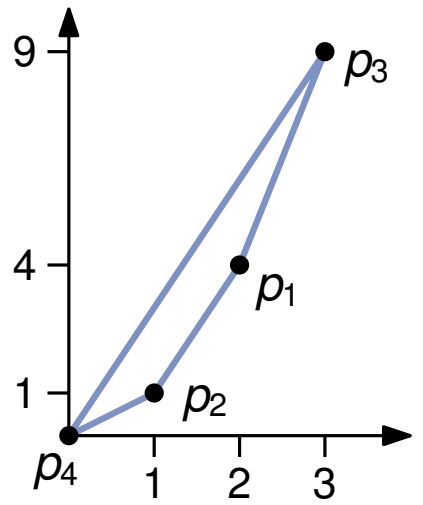
Eine untere Schranke

Satz: Untere Schranke
Die Laufzeit zur Berechnung der konvexen Hülle einer Menge von n Punkten ist in $\Theta(n \log n)$. (d.h. es gibt keinen Algorithmus mit Laufzeit $o(n \log n)$)

Beweis:
Annahme: Die konvexe Hülle von n Punkten kann in $T(n)$ Zeit berechnet werden.
Zeige: n Zahlen $a_1, \ldots, a_n$ können in $T(n) + O(n)$ Zeit sortiert werden.

- Betrachte Punktmenge $P = \{p_1, \ldots, p_n\}$ mit $p_i = (a_i, a_i^2)$
- Berechne konvexe Hülle $H(P)$ in $T(n)$ Zeit.

Beispiel: $a_1 = 2$, $a_2 = 1$, $a_3 = 3$, $a_4 = 0$



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

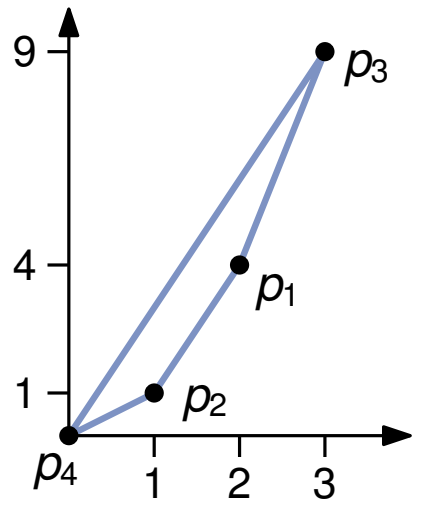
Eine untere Schranke

Satz: Untere Schranke
Die Laufzeit zur Berechnung der konvexen Hülle einer Menge von n Punkten ist in $\Theta(n \log n)$. (d.h. es gibt keinen Algorithmus mit Laufzeit $o(n \log n)$)

Beweis:
Annahme: Die konvexe Hülle von n Punkten kann in $T(n)$ Zeit berechnet werden.
Zeige: n Zahlen $a_1, \ldots, a_n$ können in $T(n) + O(n)$ Zeit sortiert werden.

- Betrachte Punktmenge $P = \{p_1, \ldots, p_n\}$ mit $p_i = (a_i, a_i^2)$
- Berechne konvexe Hülle $H(P)$ in $T(n)$ Zeit.
- $H(P)$ enthält die Punkte p_i sortiert nach a_i.
  $\Rightarrow$ Sortierung der a_i kann in $O(n)$ Zeit abgelesen werden.

Beispiel: $a_1 = 2$, $a_2 = 1$, $a_3 = 3$, $a_4 = 0$



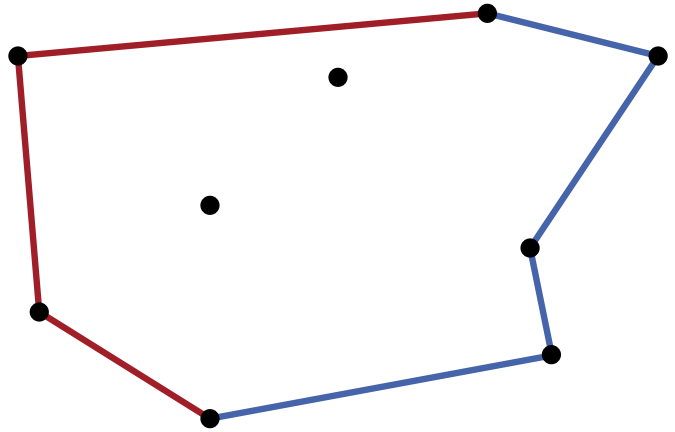
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Gift Wrapping Algorithmus (Jarvis' March)

Idee:

1. Berechne zwei Teile der konvexen Hülle getrennt: Pfad vom untersten Punkt p_0 der Konvexen Hülle zum Obersten („Rechtskette“) und dann Pfad vom Obersten zum Untersten („Linkskette“).
2. Berechnung der „Rechtskette“: Wähle p_{i+1} sodass der Winkel zwischen der Horizontalen (nach rechts) durch p_i und der Strecke $\overline{p_i p_{i+1}}$ minimal ist.
3. Berechnung der „Linkskette“: Analog von oben nach unten.

Beispiel:

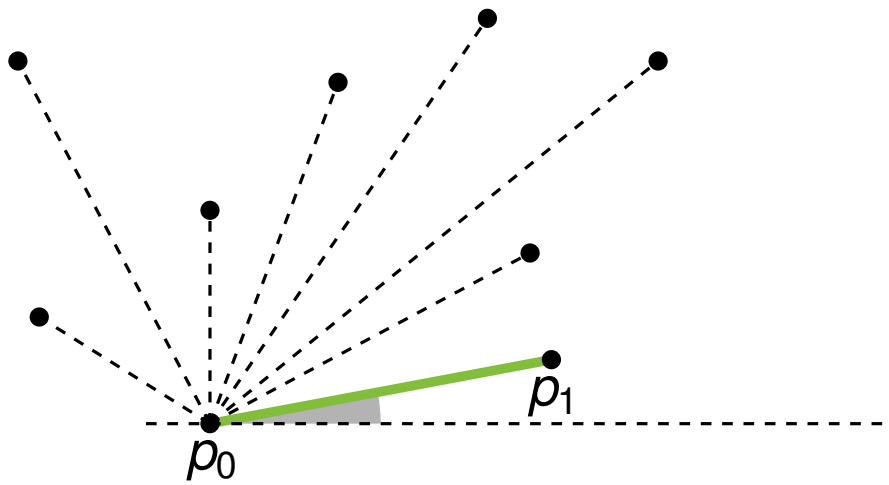


Gift Wrapping Algorithmus (Jarvis' March)

Idee:

1. Berechne zwei Teile der konvexen Hülle getrennt: Pfad vom untersten Punkt p_0 der Konvexen Hülle zum Obersten („Rechtskette“) und dann Pfad vom Obersten zum Untersten („Linkskette“).
2. Berechnung der „Rechtskette“: Wähle p_{i+1} sodass der Winkel zwischen der Horizontalen (nach rechts) durch p_i und der Strecke $\overline{p_i p_{i+1}}$ minimal ist.
3. Berechnung der „Linkskette“: Analog von oben nach unten.

Beispiel:

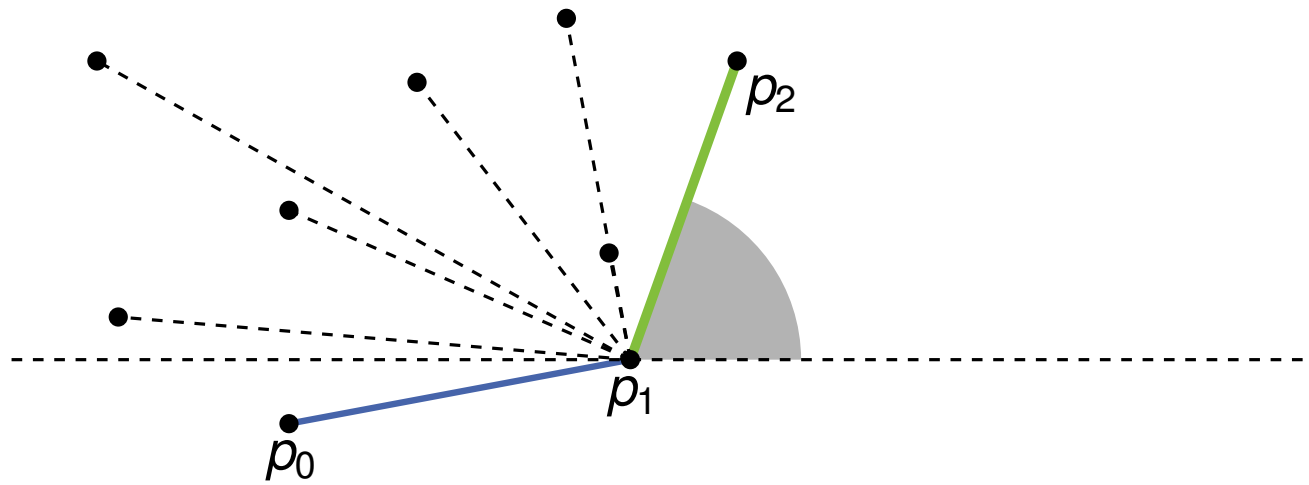


Gift Wrapping Algorithmus (Jarvis' March)

Idee:

1. Berechne zwei Teile der konvexen Hülle getrennt: Pfad vom untersten Punkt p_0 der Konvexen Hülle zum Obersten („Rechtskette“) und dann Pfad vom Obersten zum Untersten („Linkskette“).
2. Berechnung der „Rechtskette“: Wähle p_{i+1} sodass der Winkel zwischen der Horizontalen (nach rechts) durch p_i und der Strecke $\overline{p_i p_{i+1}}$ minimal ist.
3. Berechnung der „Linkskette“: Analog von oben nach unten.

Beispiel:

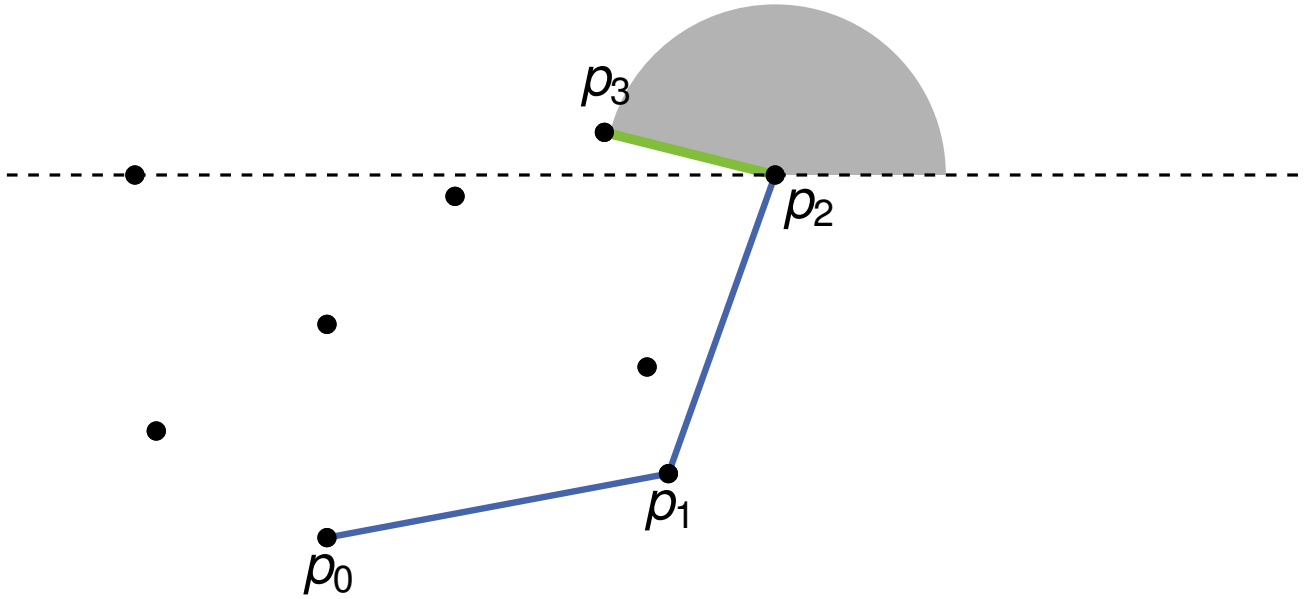


Gift Wrapping Algorithmus (Jarvis' March)

Idee:

1. Berechne zwei Teile der konvexen Hülle getrennt: Pfad vom untersten Punkt p_0 der Konvexen Hülle zum Obersten („Rechtskette“) und dann Pfad vom Obersten zum Untersten („Linkskette“).
2. Berechnung der „Rechtskette“: Wähle p_{i+1} sodass der Winkel zwischen der Horizontalen (nach rechts) durch p_i und der Strecke $\overline{p_i p_{i+1}}$ minimal ist.
3. Berechnung der „Linkskette“: Analog von oben nach unten.

Beispiel:

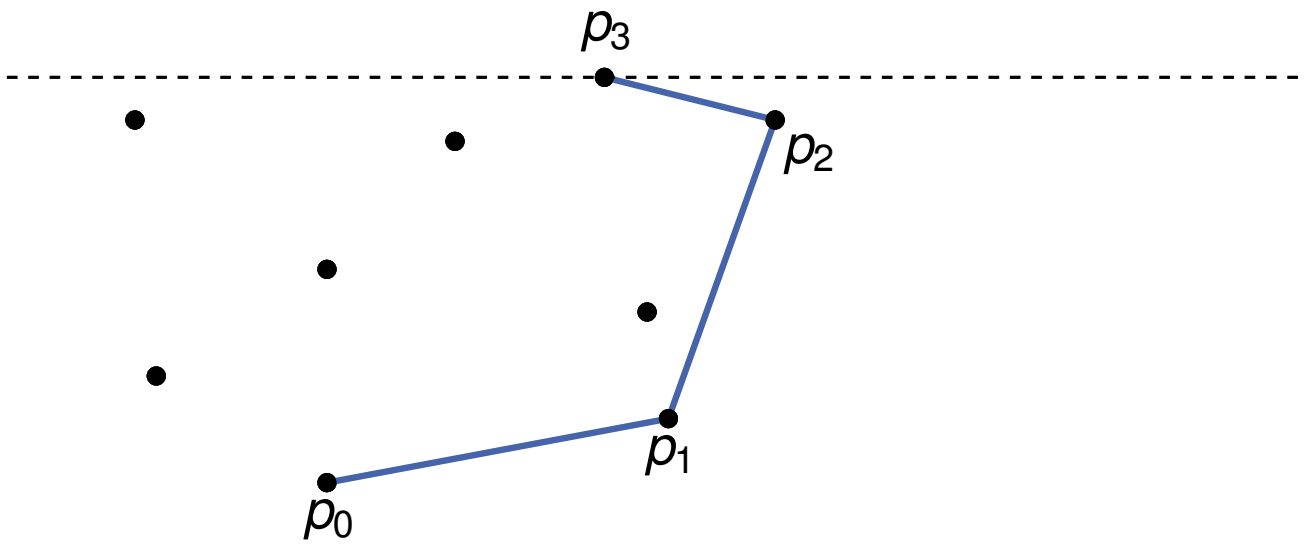


Gift Wrapping Algorithmus (Jarvis' March)

Idee:

1. Berechne zwei Teile der konvexen Hülle getrennt: Pfad vom untersten Punkt p_0 der Konvexen Hülle zum Obersten („Rechtskette“) und dann Pfad vom Obersten zum Untersten („Linkskette“).
2. Berechnung der „Rechtskette“: Wähle p_{i+1} sodass der Winkel zwischen der Horizontalen (nach rechts) durch p_i und der Strecke $\overline{p_i p_{i+1}}$ minimal ist.
3. Berechnung der „Linkskette“: Analog von oben nach unten.

Beispiel:

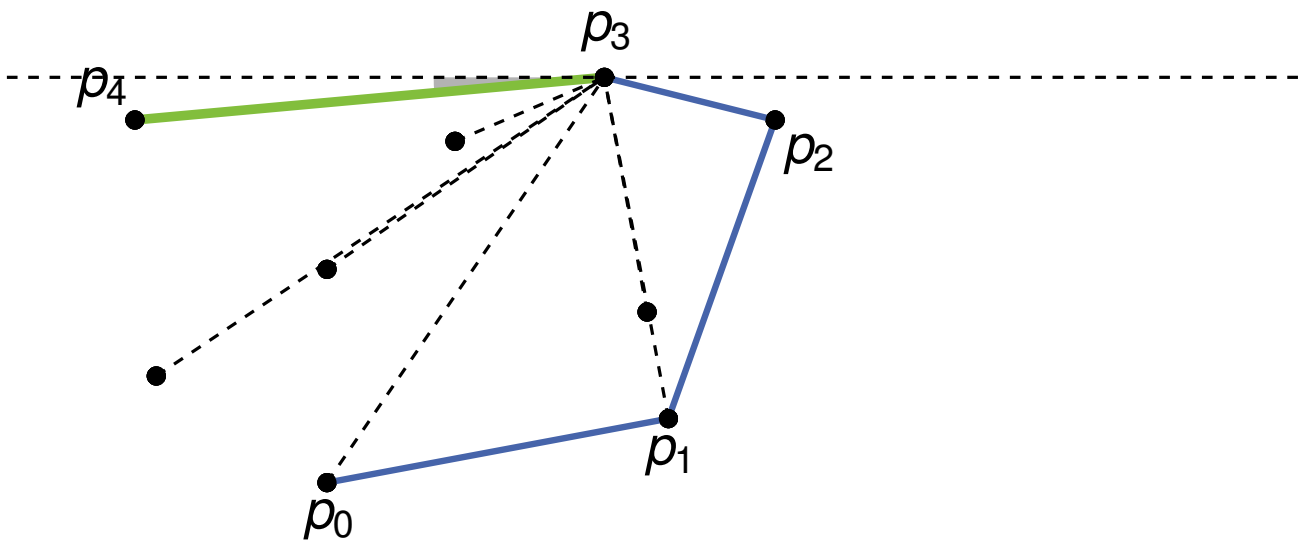


Gift Wrapping Algorithmus (Jarvis' March)

Idee:

1. Berechne zwei Teile der konvexen Hülle getrennt: Pfad vom untersten Punkt p_0 der Konvexen Hülle zum Obersten („Rechtskette") und dann Pfad vom Obersten zum Untersten („Linkskette").
2. Berechnung der „Rechtskette": Wähle p_{i+1} sodass der Winkel zwischen der Horizontalen (nach rechts) durch p_i und der Strecke $\overline{p_i p_{i+1}}$ minimal ist.
3. Berechnung der „Linkskette": Analog von oben nach unten.

Beispiel:

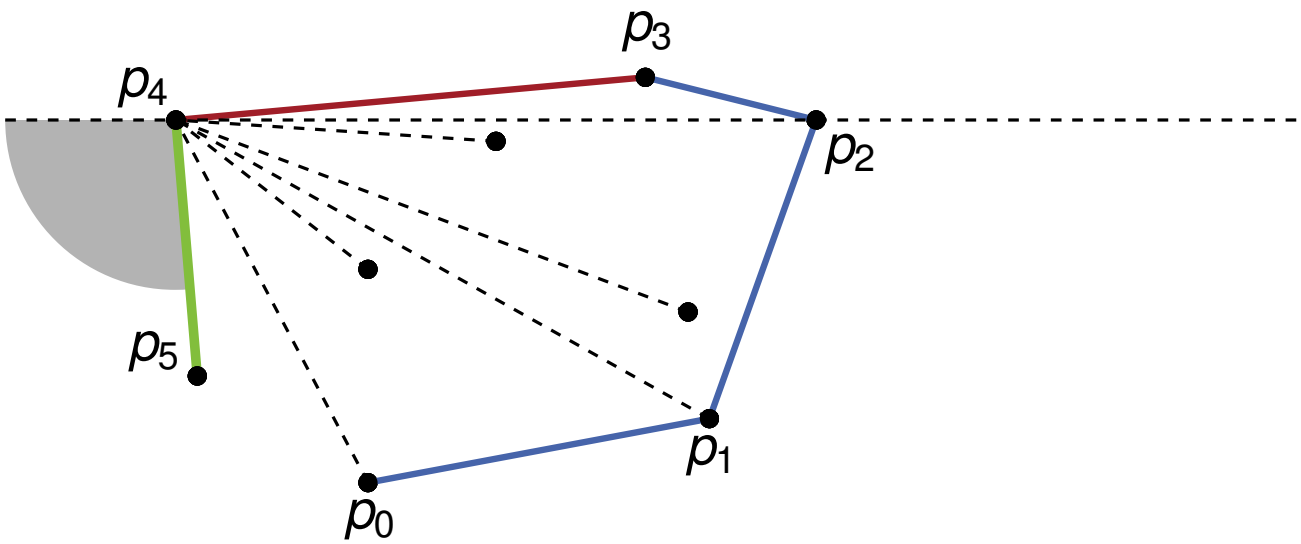


Gift Wrapping Algorithmus (Jarvis' March)

Idee:

1. Berechne zwei Teile der konvexen Hülle getrennt: Pfad vom untersten Punkt p_0 der Konvexen Hülle zum Obersten („Rechtskette“) und dann Pfad vom Obersten zum Untersten („Linkskette“).
2. Berechnung der „Rechtskette“: Wähle p_{i+1} sodass der Winkel zwischen der Horizontalen (nach rechts) durch p_i und der Strecke $\overline{p_i p_{i+1}}$ minimal ist.
3. Berechnung der „Linkskette“: Analog von oben nach unten.

Beispiel:

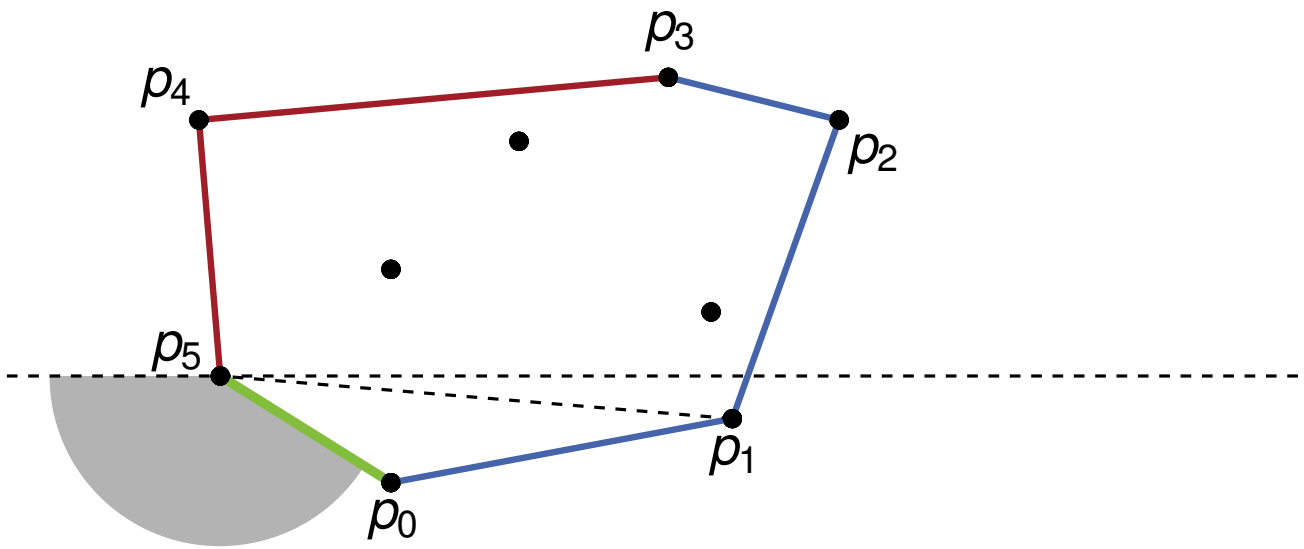


Gift Wrapping Algorithmus (Jarvis' March)

Idee:

1. Berechne zwei Teile der konvexen Hülle getrennt: Pfad vom untersten Punkt p_0 der Konvexen Hülle zum Obersten („Rechtskette“) und dann Pfad vom Obersten zum Untersten („Linkskette“).
2. Berechnung der „Rechtskette“: Wähle p_{i+1} sodass der Winkel zwischen der Horizontalen (nach rechts) durch p_i und der Strecke $\overline{p_i p_{i+1}}$ minimal ist.
3. Berechnung der „Linkskette“: Analog von oben nach unten.

Beispiel:

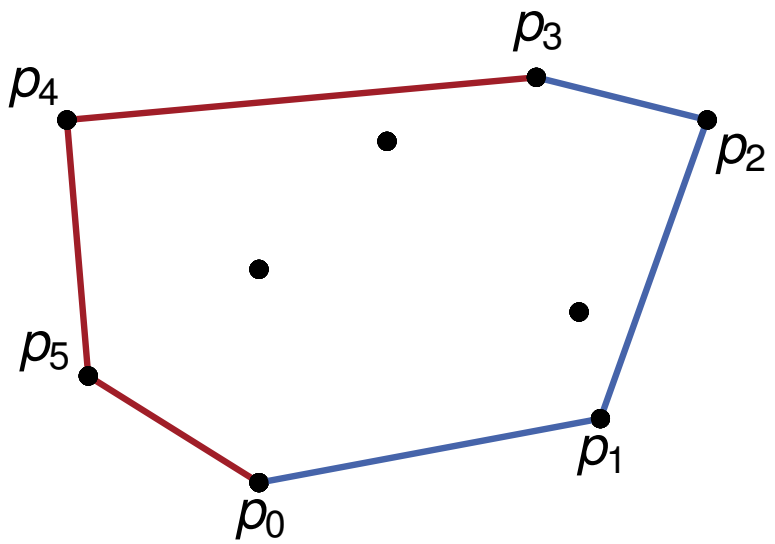


Gift Wrapping Algorithmus (Jarvis' March)

Idee:

1. Berechne zwei Teile der konvexen Hülle getrennt: Pfad vom untersten Punkt p_0 der Konvexen Hülle zum Obersten („Rechtskette“) und dann Pfad vom Obersten zum Untersten („Linkskette“).
2. Berechnung der „Rechtskette“: Wähle p_{i+1} sodass der Winkel zwischen der Horizontalen (nach rechts) durch p_i und der Strecke $\overline{p_i p_{i+1}}$ minimal ist.
3. Berechnung der „Linkskette“: Analog von oben nach unten.

Beispiel:



Gift Wrapping Algorithmus (Jarvis' March)

JARVIS' MARCH(Q)

$p_0 \leftarrow$ Punkt in Q mit kleinster y-Koordinate

$i \leftarrow 0$

while *Es gibt einen Punkt oberhalb von* p_i **do**

$p_{i+1} \leftarrow$ Punkt oberhalb von p_i, sodass der Winkel zwischen der Strecke $\overline{p_i p_{i+1}}$ und der Horizontalen durch p_i nach rechts minimal ist.

$i \leftarrow i + 1$

while $p_i \neq p_0$ **do**

$p_{i+1} \leftarrow$ Punkt unterhalb von p_i, sodass der Winkel zwischen der Strecke $\overline{p_i p_{i+1}}$ und der Horizontalen durch p_i nach links minimal ist.

$i \leftarrow i + 1$

Gift Wrapping Algorithmus (Jarvis' March)

JARVIS' MARCH(Q)

$p_0 \leftarrow$ Punkt in Q mit kleinster y-Koordinate
$i \leftarrow 0$ — $O(1)$

while *Es gibt einen Punkt oberhalb von* p_i **do**

$p_{i+1} \leftarrow$ Punkt oberhalb von p_i, sodass der Winkel zwischen der Strecke $\overline{p_i p_{i+1}}$ und der Horizontalen durch p_i nach rechts minimal ist.
$i \leftarrow i + 1$ — $O(n)$

while $p_i \neq p_0$ **do**

$p_{i+1} \leftarrow$ Punkt unterhalb von p_i, sodass der Winkel zwischen der Strecke $\overline{p_i p_{i+1}}$ und der Horizontalen durch p_i nach links minimal ist.
$i \leftarrow i + 1$ — $O(n)$

Gift Wrapping Algorithmus (Jarvis' March)

JARVIS' MARCH(Q) — $O(hn)$

- $p_0 \leftarrow$ Punkt in Q mit kleinster y-Koordinate — $O(1)$
  $i \leftarrow 0$
- **while** *Es gibt einen Punkt oberhalb von* p_i **do** — $O(hn)$
  - $p_{i+1} \leftarrow$ Punkt oberhalb von p_i, sodass der Winkel zwischen der Strecke $\overline{p_i p_{i+1}}$ und der Horizontalen durch p_i nach rechts minimal ist. — $O(n)$
    $i \leftarrow i + 1$
- **while** $p_i \neq p_0$ **do** — $O(hn)$
  - $p_{i+1} \leftarrow$ Punkt unterhalb von p_i, sodass der Winkel zwischen der Strecke $\overline{p_i p_{i+1}}$ und der Horizontalen durch p_i nach links minimal ist. — $O(n)$
    $i \leftarrow i + 1$

Satz: Jarvis' March
Der Gift Wrapping Algorithmus (Jarvis' March) berechnet die konvexe Hülle $H(Q)$ von Q in $O(hn)$ Zeit, wobei h die Anzahl der Eckpunkte von $H(Q)$ ist.

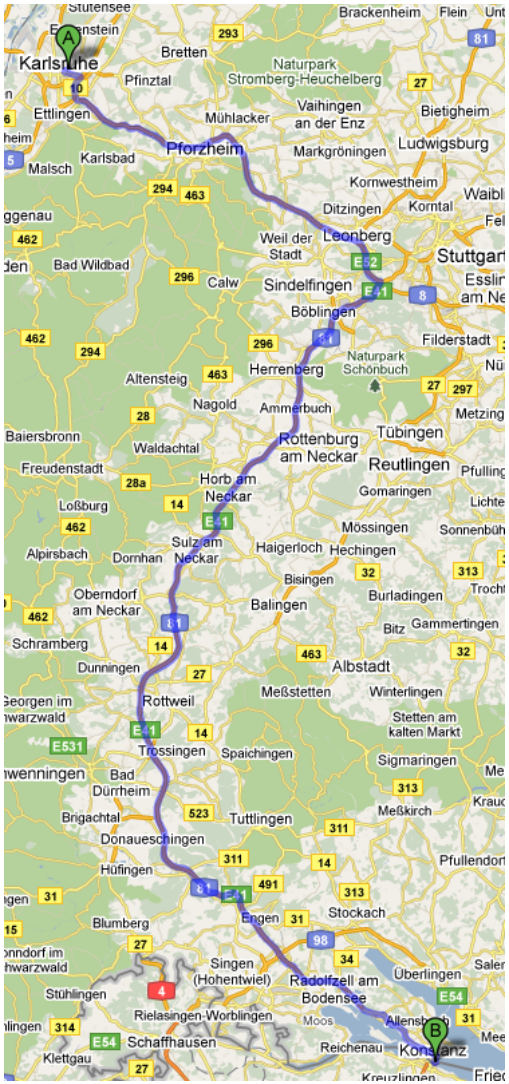
Ein solcher Algorithmus wird *ausgabesensitiv* genannt.

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

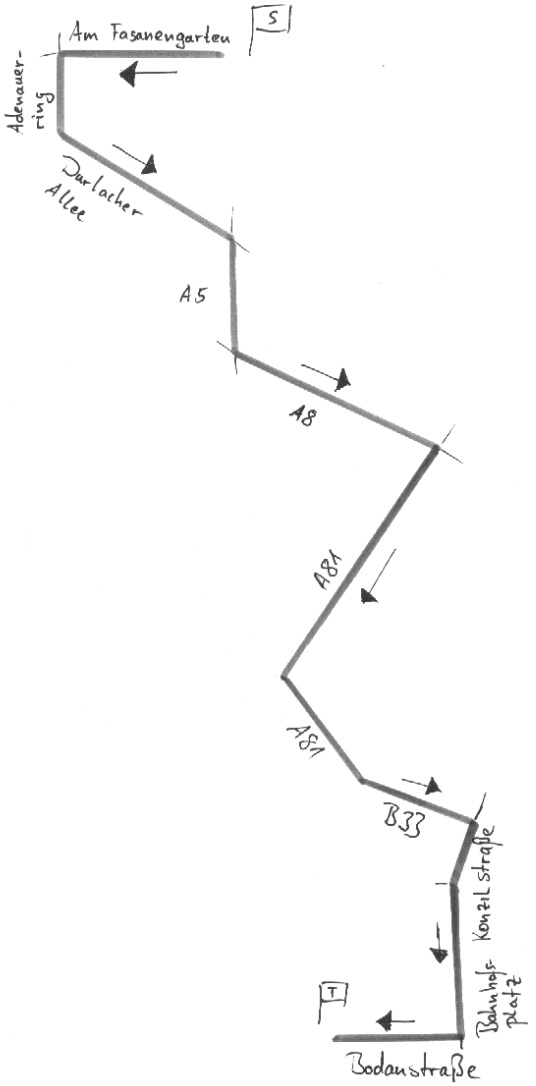
Algorithmische Geometrie – Anwendungen

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

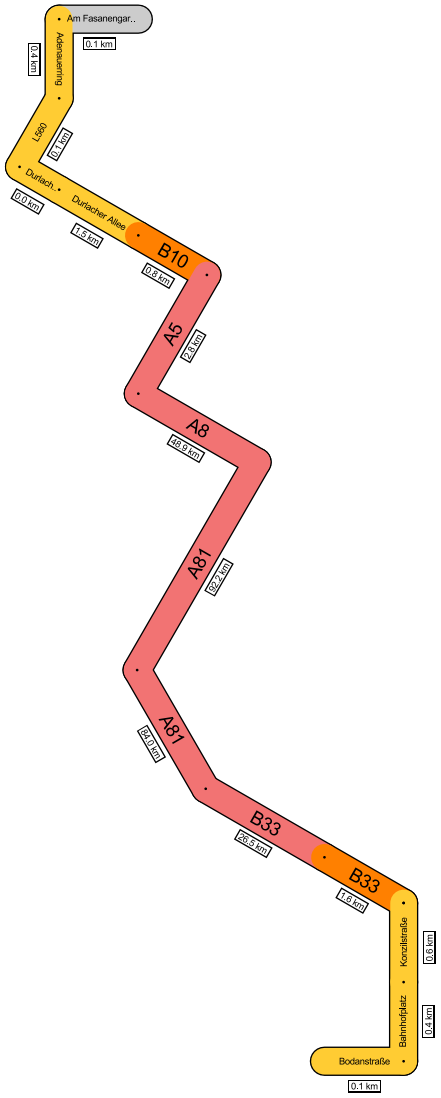
Routenskizzen



Google Maps



Manuell gezeichnete Routenskizze



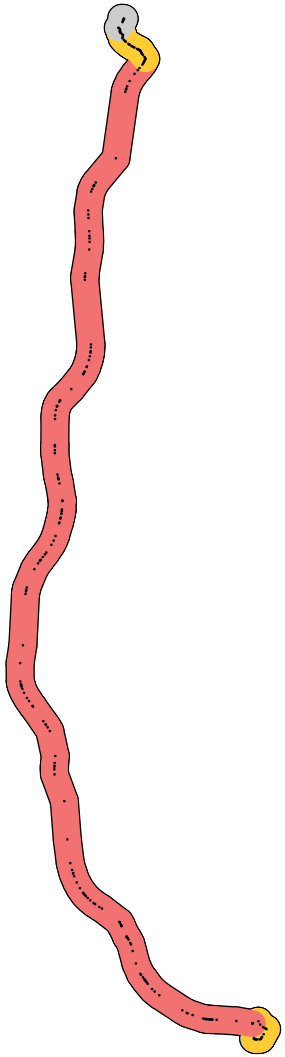
Automatisch erzeugte Routenskizze

von Karlsruhe nach Konstanz

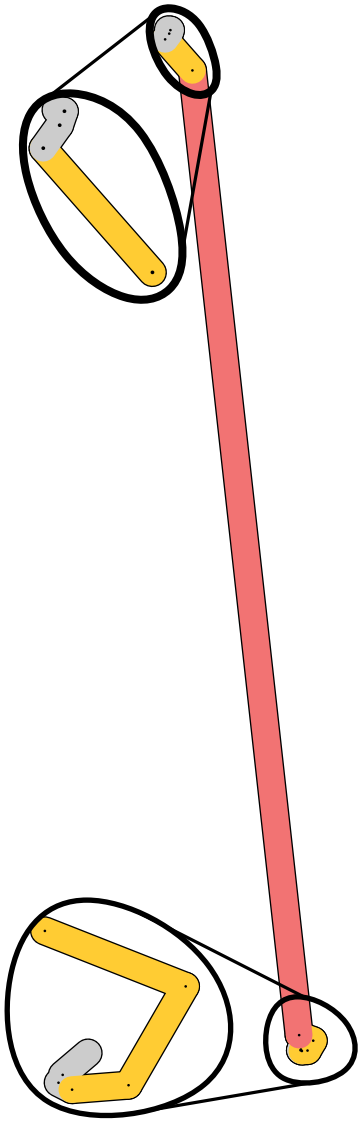
Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Routenskizzen

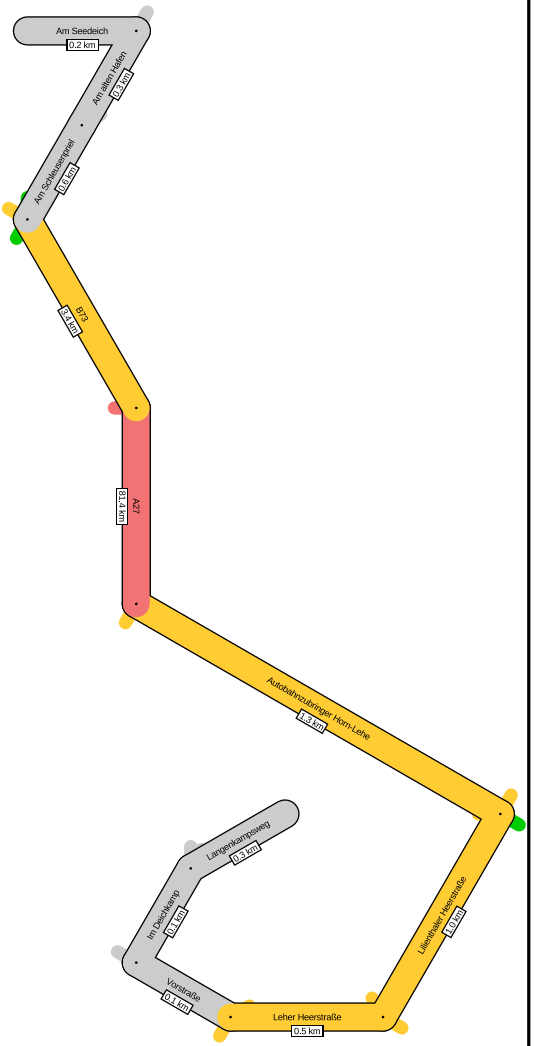
Eingabe



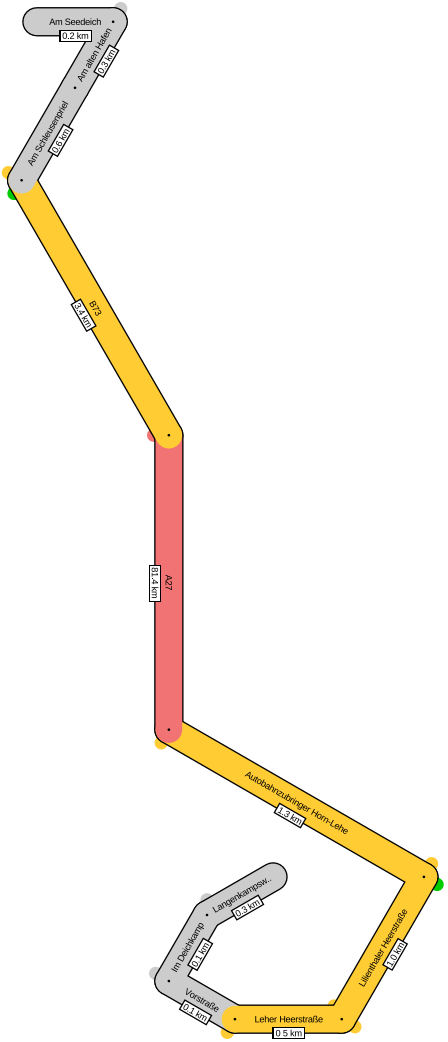
Vereinfachung



Schematisierung (I)

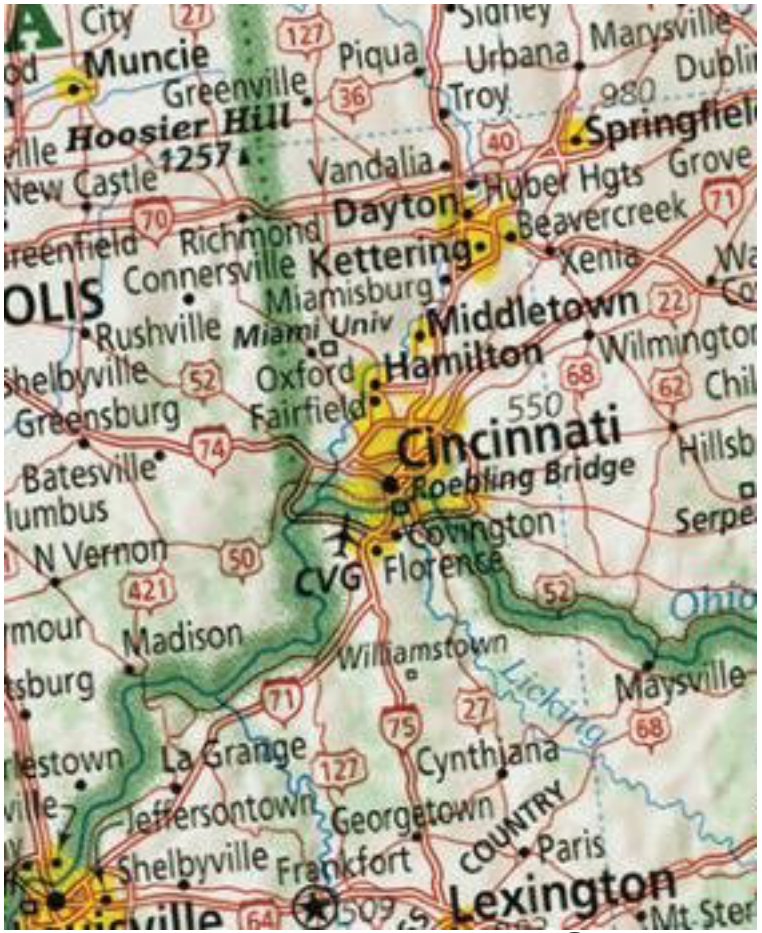


Schematisierung (II)



Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Kartenbeschriftung (Labeling)



© David Imus

„Poor, sloppy, amateurisch type placement is irresponsible; it spoils even the best image and impedes reading.“ (E. Imhof ’75)

Die Kartografie hat umfangreiche Erfahrung im manuellen Beschriften von Karten.

Einige Platzierungsrichtlinien:

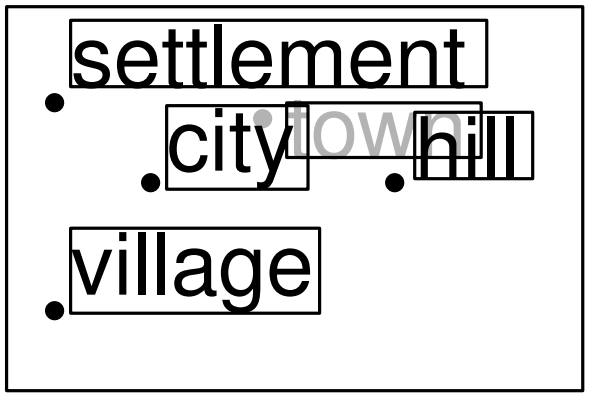
- neben, über oder unter dem Objekt
- wenn möglich oben rechts
- vermeide Überdeckungen
- eindeutige grafische Zuordnung
- ...

$\rightarrow$ automatische Kartenbeschriftung wurde vor $>$ 20 Jahren als Problem der algorithmischen Geometrie formuliert

Kartenbeschriftung (Labeling)

Eingabe: n Punkte in der Ebene und für jeden Punkt ein Label repräsentiert durch dessen Bounding Box

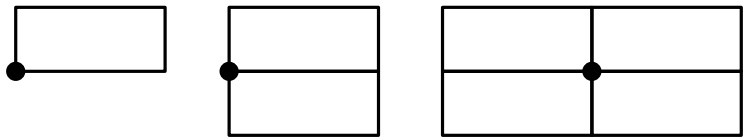
Ziel: finde eine zulässige* Platzierung für eine maximale Teilmenge der Label, ohne dass sich Labels überschneiden (MAXNUMBER)



MAXNUMBER

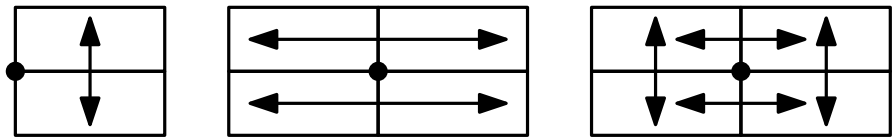
* Was ist eine zulässige Platzierung?

diskrete Modelle



1P 2P 4P

Slider-Modelle



1S 2S 4S

Institut für Theoretische Informatik
Prof. Dr. Dorothea Wagner

Dynamische Kartenbeschriftung

Die meisten Karten sind heute nicht mehr statisch und allgemein, sondern dynamisch und individuell.

Kartenansicht bewegt sich kontinuierlich wenn der Nutzer die Karte

- zoomt
- verschiebt
- rotiert
- neigt

Layout und Beschriftung müssen sich an die Kartenbewegung anpassen

Panorama Labeling