

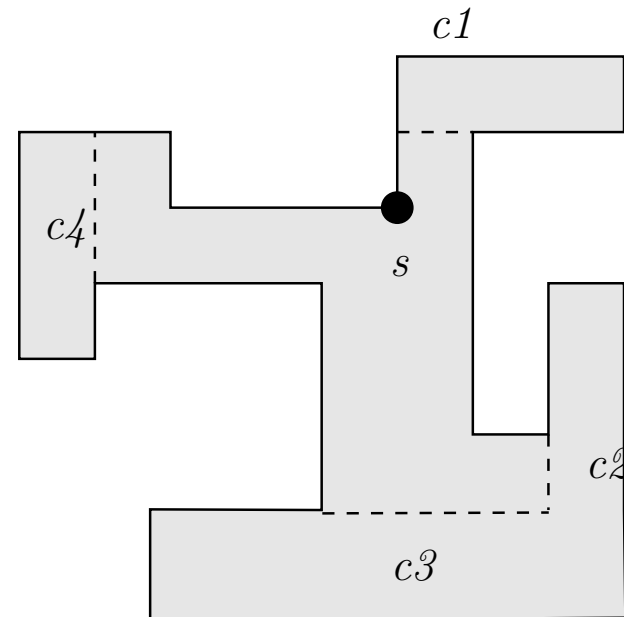
Offline Bewegungsplanung: SWR und Touring

Elmar Langetepe
University of Bonn

Lemma 1.26 Ordnung entlang des Randes

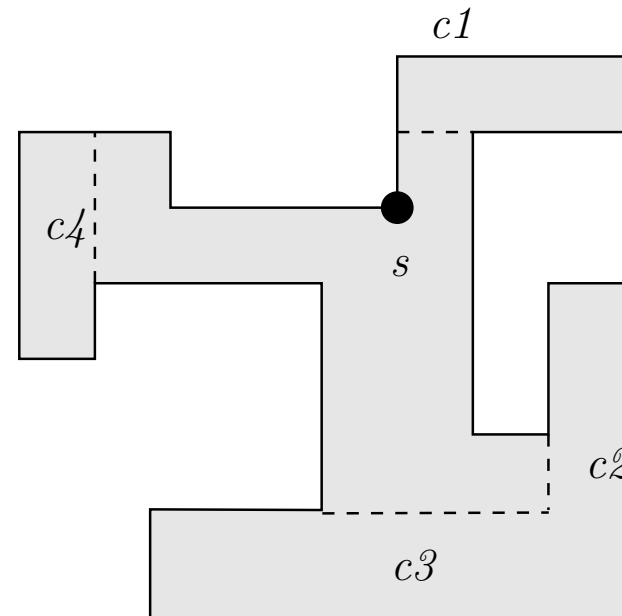
Lemma 1.26 Ordnung entlang des Randes

- Rechtwinkeliges Polygon



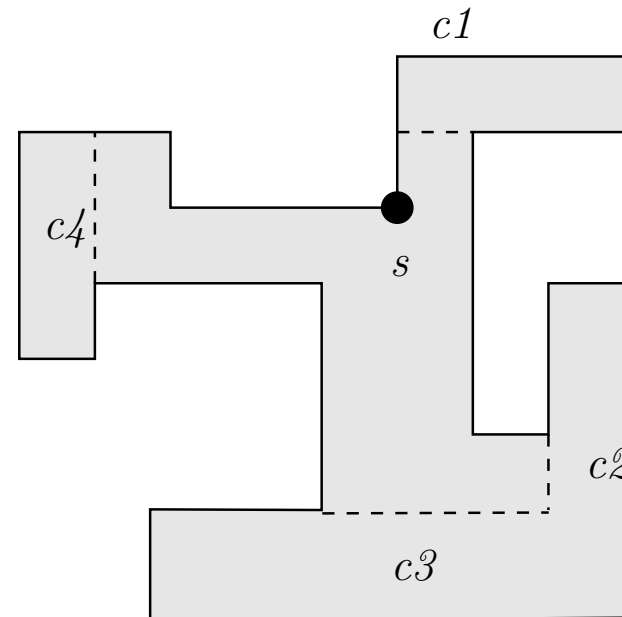
Lemma 1.26 Ordnung entlang des Randes

- Rechtwinkeliges Polygon
- Wesentliche Cuts werden nie überschritten!



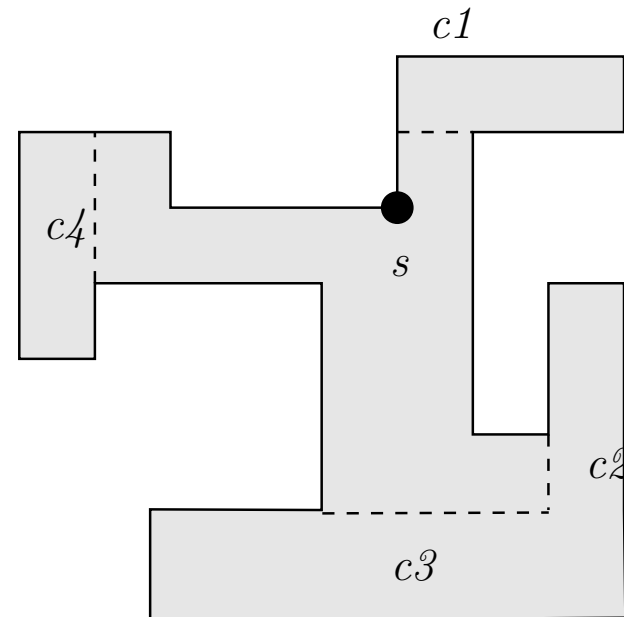
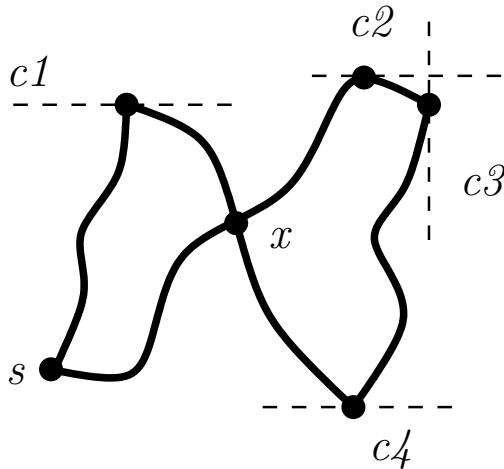
Lemma 1.26 Ordnung entlang des Randes

- Rechtwinkeliges Polygon
- Wesentliche Cuts werden nie überschritten!
- SWR besucht Cuts gemäß Ordnung



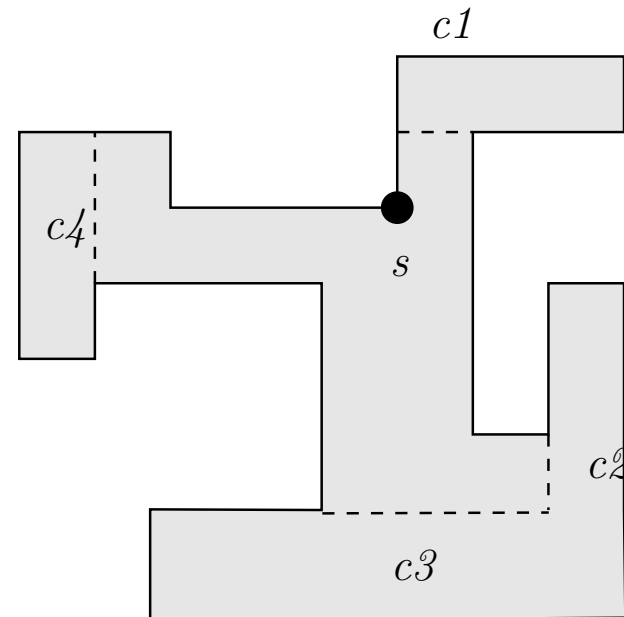
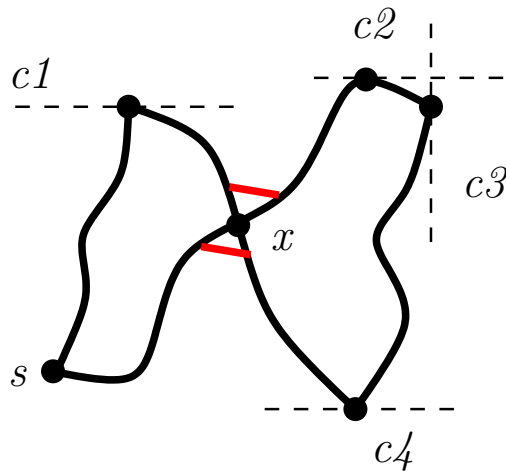
Lemma 1.26 Ordnung entlang des Randes

- Rechtwinkeliges Polygon
- Wesentliche Cuts werden nie überschritten!
- SWR besucht Cuts gemäß Ordnung
- Widerspruch! Abkürzung!



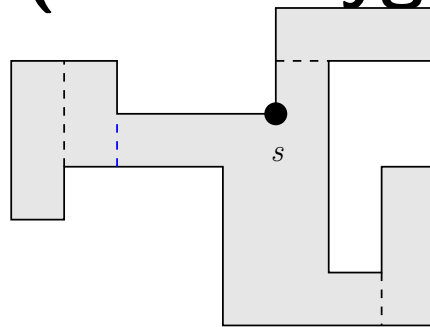
Lemma 1.26 Ordnung entlang des Randes

- Rechtwinkeliges Polygon
- Wesentliche Cuts werden nie überschritten!
- SWR besucht Cuts gemäß Ordnung
- Widerspruch! Abkürzung!

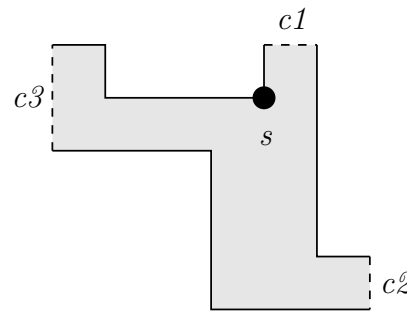


SWR (RW Polygon) Alg. 1.9/Th. 1.27: $O(n)$

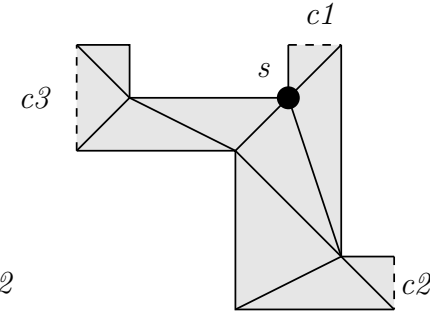
SWR (RW Polygon) Alg. 1.9/Th. 1.27: $O(n)$



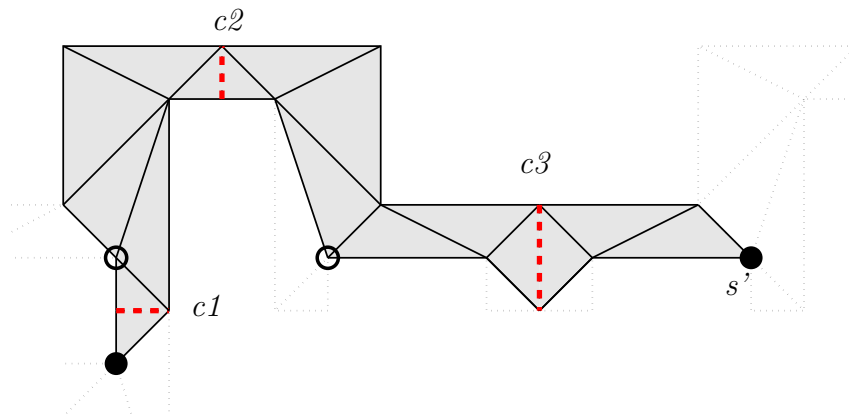
(i) Wesentliche Cuts



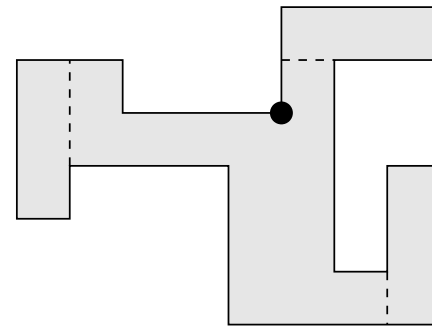
(ii) Abschneiden!



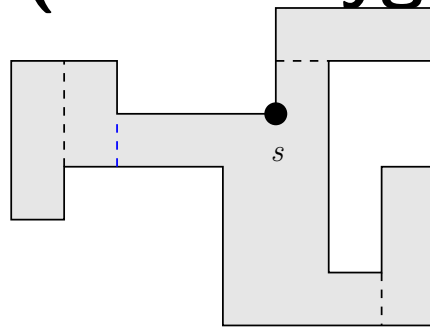
(iii) Triangulation



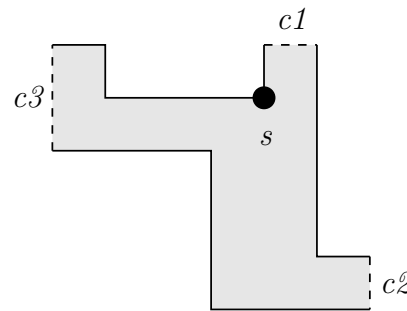
(iv) Spiegeln und Ausrollen!!



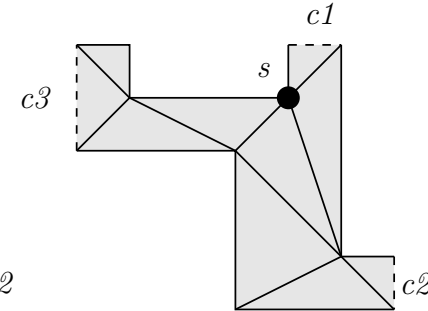
SWR (RW Polygon) Alg. 1.9/Th. 1.27: $O(n)$



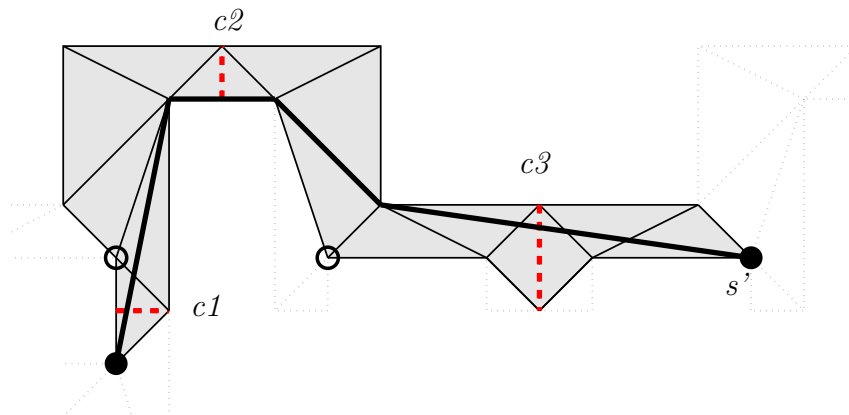
(i) Wesentliche Cuts



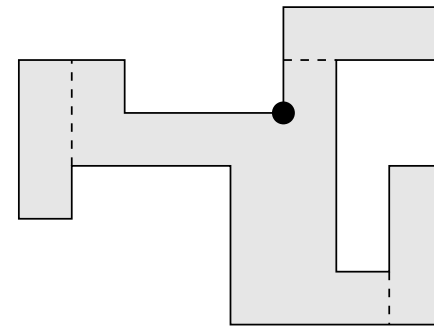
(ii) Abschneiden!



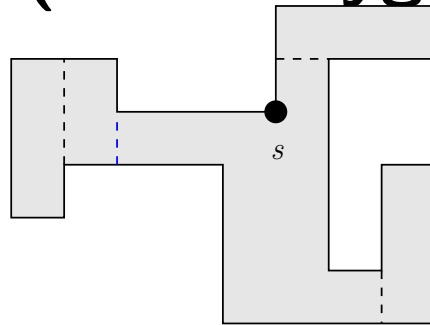
(iii) Triangulation



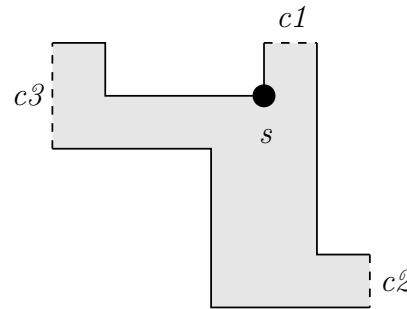
(iv) Spiegeln und Ausrollen!!
 (v) Weg berechnen



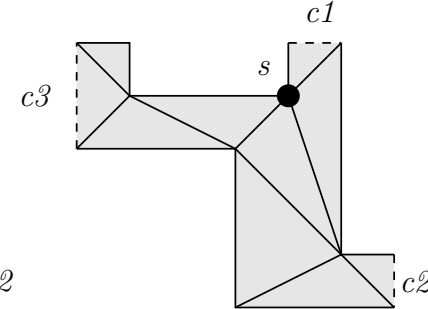
SWR (RW Polygon) Alg. 1.9/Th. 1.27: $O(n)$



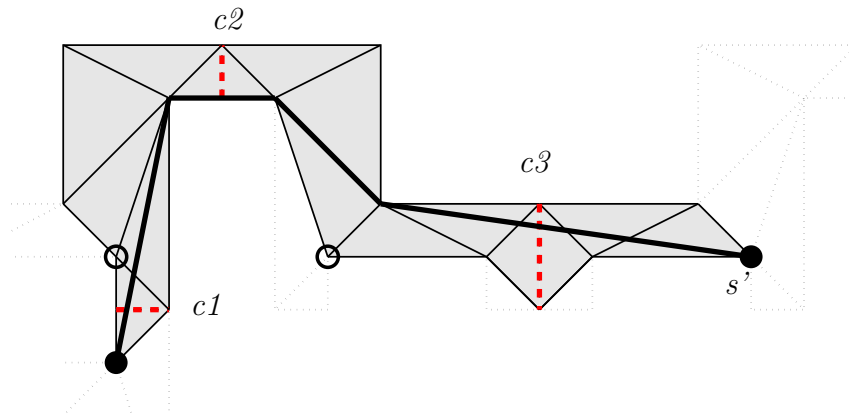
(i) Wesentliche Cuts



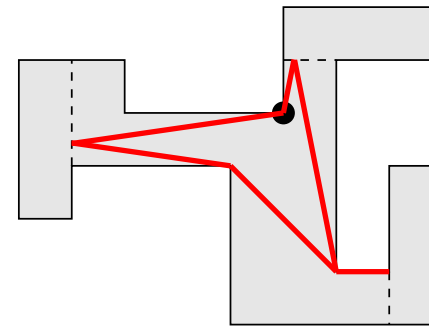
(ii) Abschneiden!



(iii) Triangulation

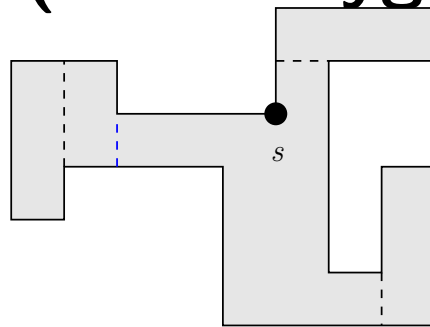


(iv) Spiegeln und Ausrollen!!
(v) Weg berechnen

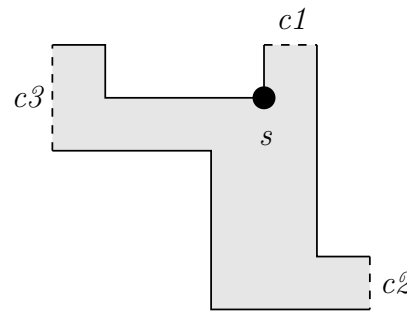


(vi) Zusammenklappen!

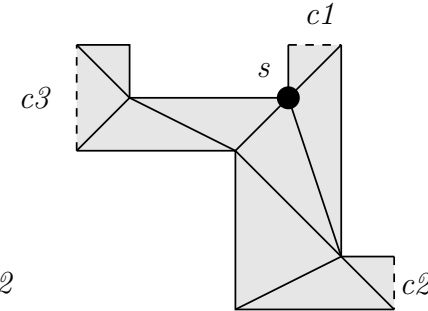
SWR (RW Polygon) Alg. 1.9/Th. 1.27: $O(n)$



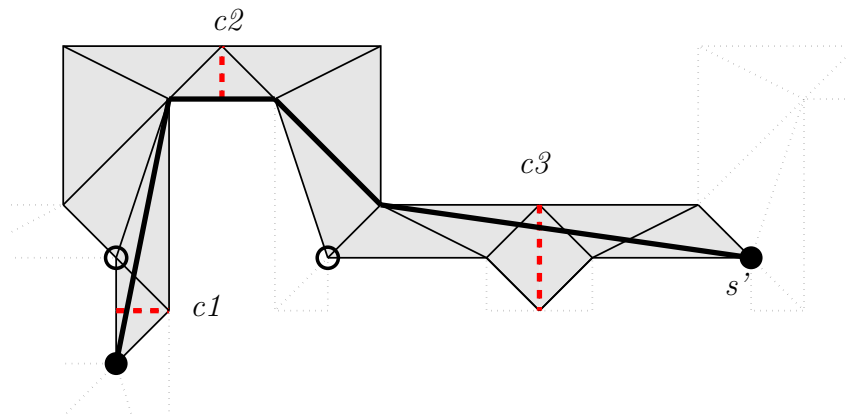
(i) Wesentliche Cuts
 $O(n)$



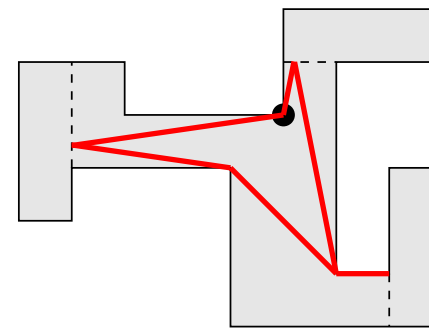
(ii) Abschneiden!



(iii) Triangulation

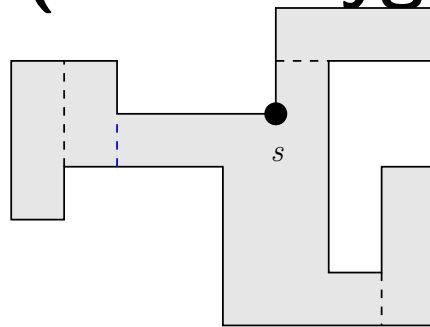


(iv) Spiegeln und Ausrollen!!
(v) Weg berechnen

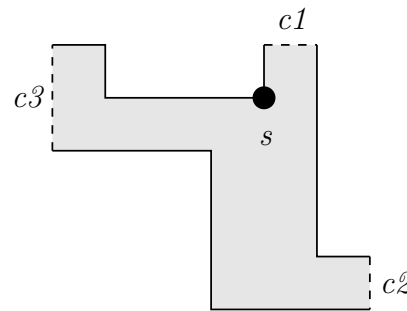


(vi) Zusammenklappen!

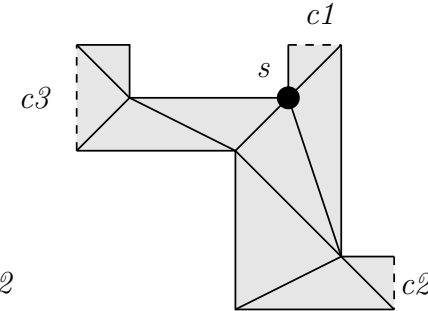
SWR (RW Polygon) Alg. 1.9/Th. 1.27: $O(n)$



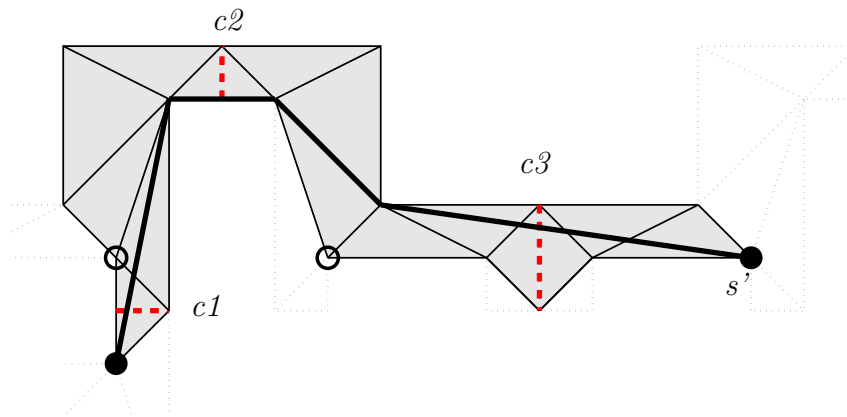
(i) Wesentliche Cuts
 $O(n)$



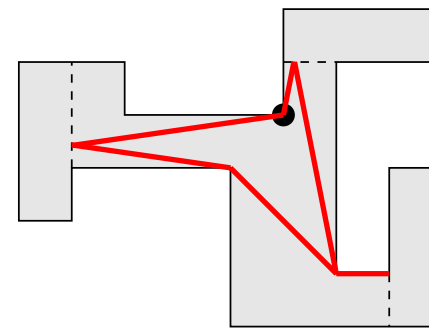
(ii) Abschneiden!
 $O(n)$



(iii) Triangulation

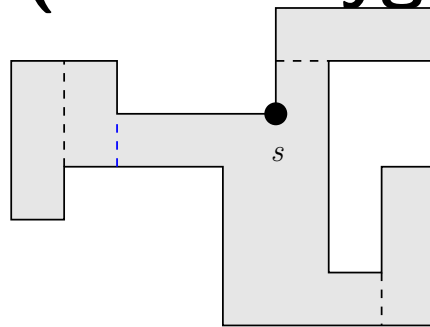


(iv) Spiegeln und Ausrollen!!
(v) Weg berechnen

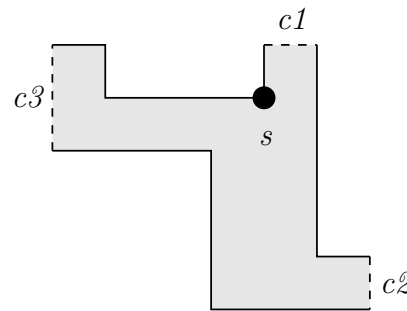


(vi) Zusammenklappen!

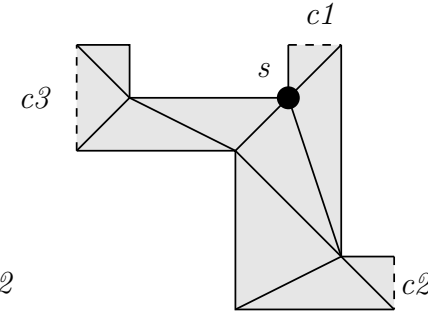
SWR (RW Polygon) Alg. 1.9/Th. 1.27: $O(n)$



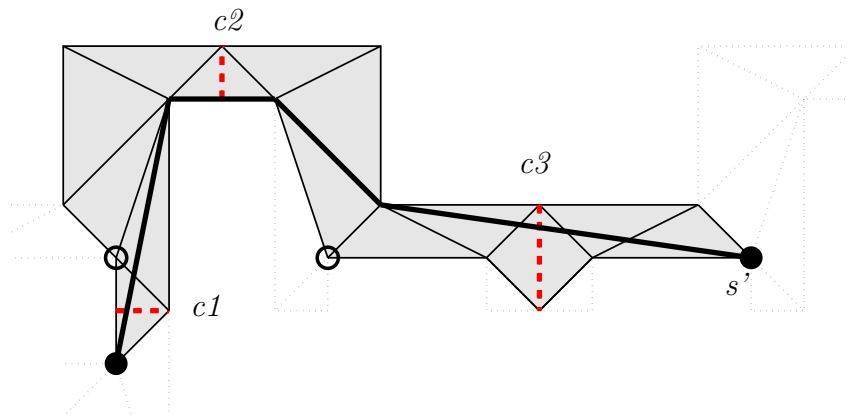
(i) Wesentliche Cuts
 $O(n)$



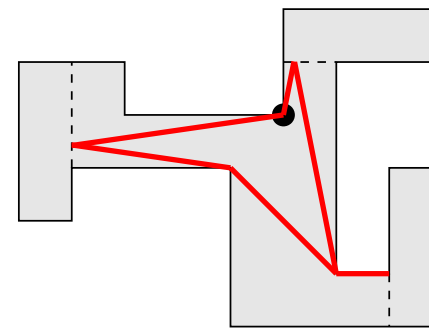
(ii) Abschneiden!
 $O(n)$



(iii) Triangulation
 $O(n)$

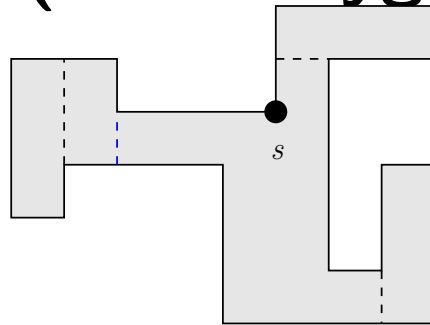


(iv) Spiegeln und Ausrollen!!
(v) Weg berechnen

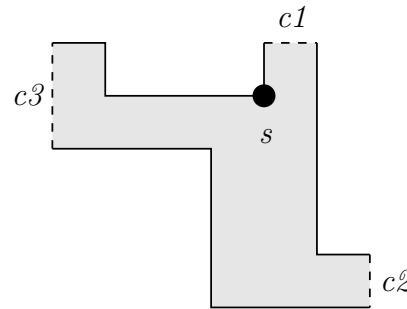


(vi) Zusammenklappen!

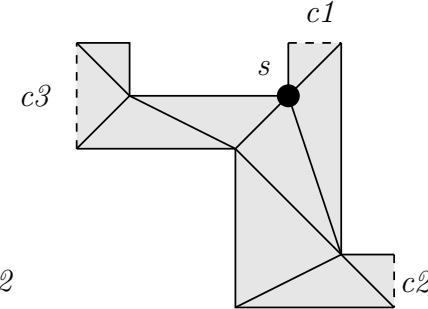
SWR (RW Polygon) Alg. 1.9/Th. 1.27: $O(n)$



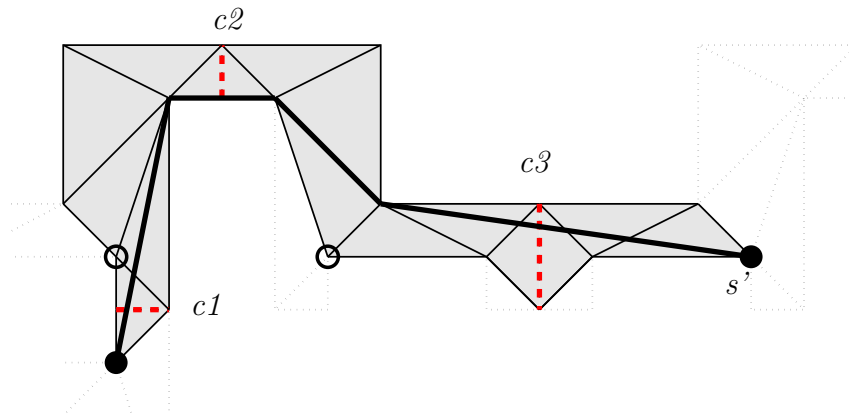
(i) Wesentliche Cuts
 $O(n)$



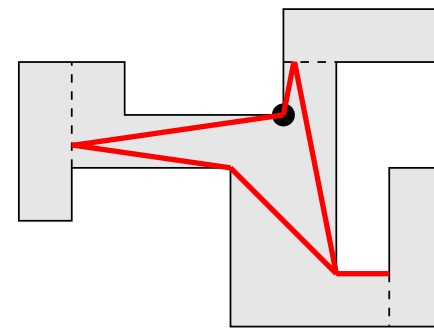
(ii) Abschneiden!
 $O(n)$



(iii) Triangulation
 $O(n)$

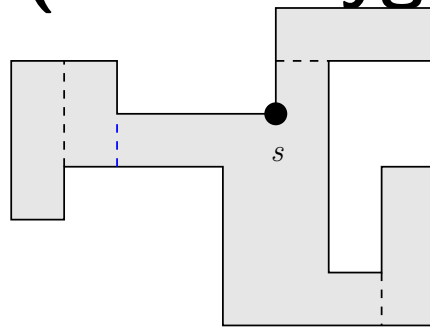


(iv) Spiegeln und Ausrollen!!
(v) Weg berechnen
 $O(n)$

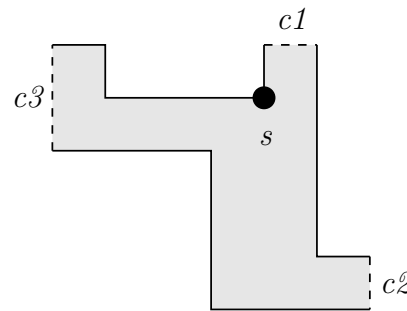


(vi) Zusammenklappen!

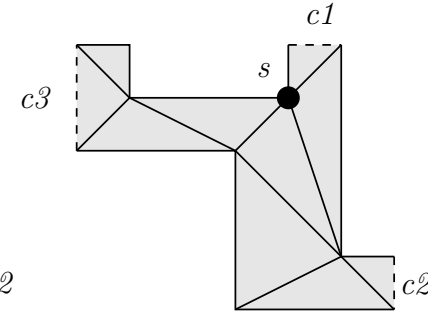
SWR (RW Polygon) Alg. 1.9/Th. 1.27: $O(n)$



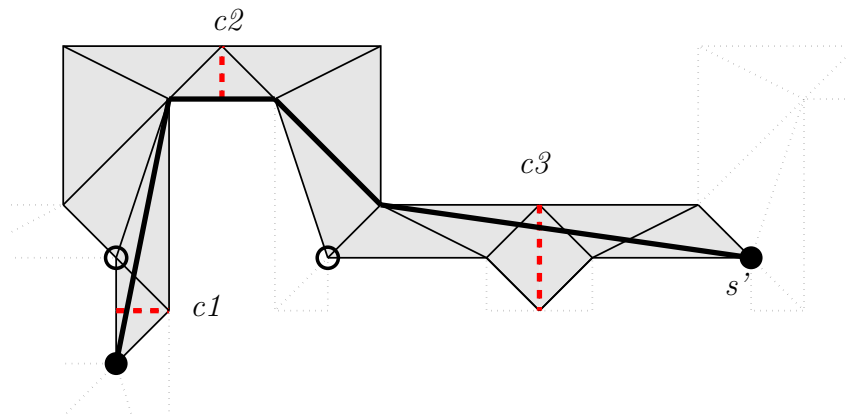
(i) Wesentliche Cuts
 $O(n)$



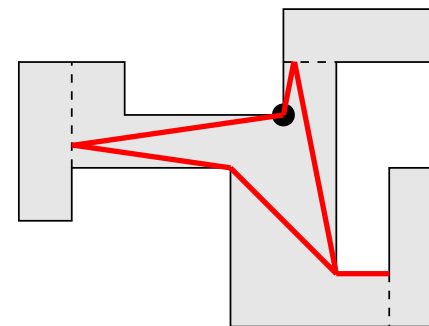
(ii) Abschneiden!
 $O(n)$



(iii) Triangulation
 $O(n)$



(iv) Spiegeln und Ausrollen!!
(v) Weg berechnen
 $O(n)$

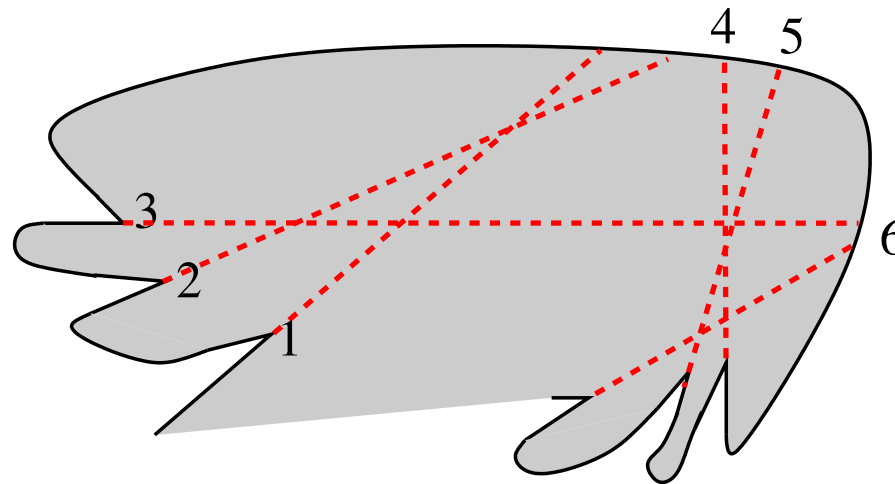


(vi) Zusammenklappen!
 $O(n)$

SWR (Allgemeiner Fall)

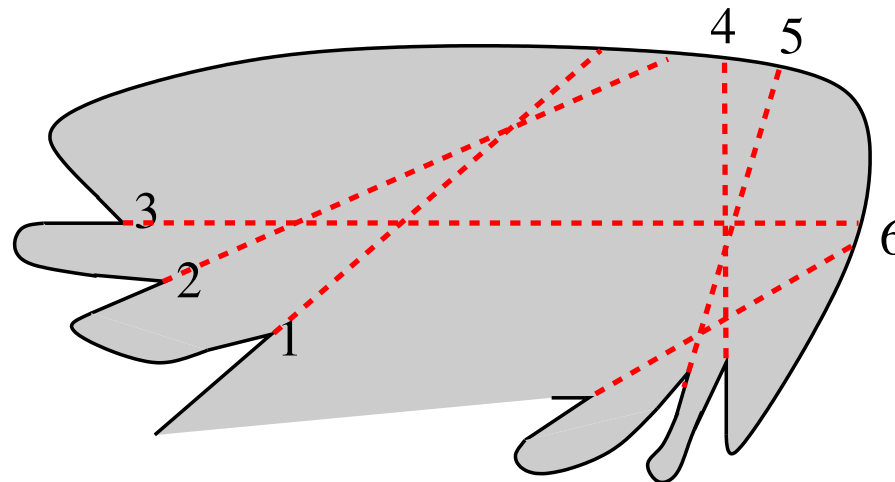
SWR (Allgemeiner Fall)

- Corner Problem!!



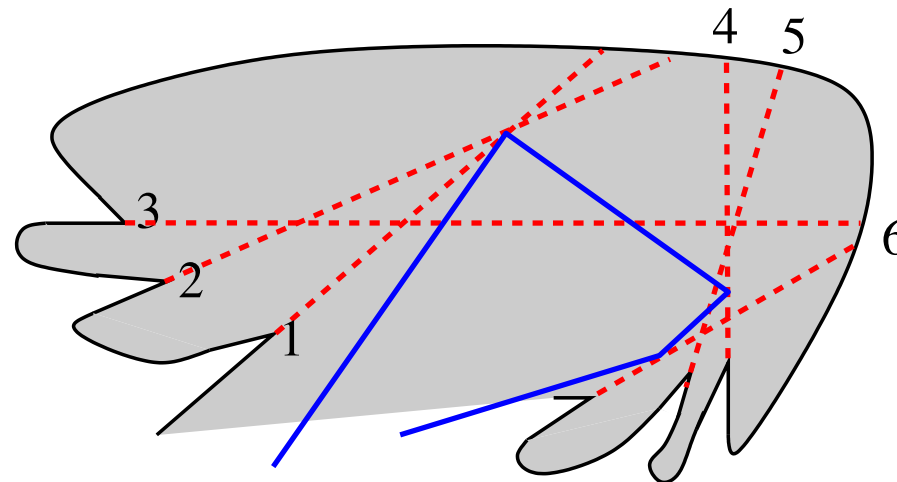
SWR (Allgemeiner Fall)

- Corner Problem!!
- **Def. 1.28** Folge wes. Cuts, die sich sukzessive schneiden



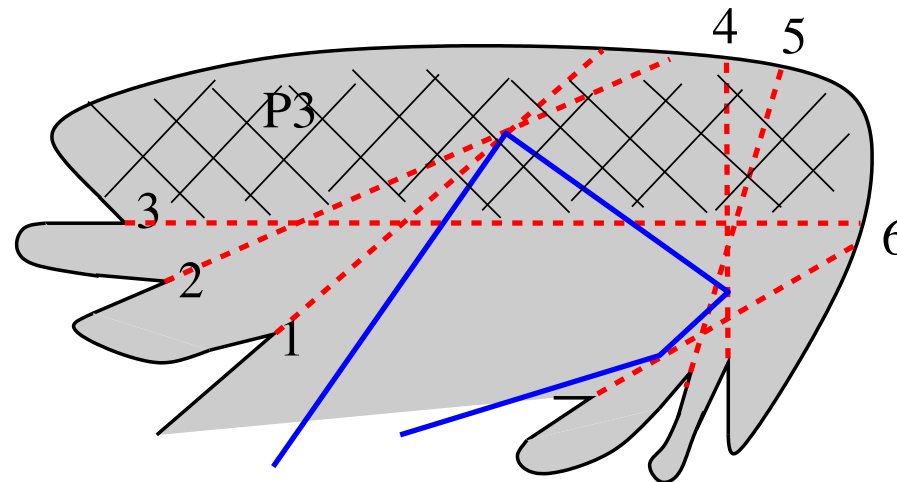
SWR (Allgemeiner Fall)

- Corner Problem!!
- **Def. 1.28** Folge wes. Cuts, die sich sukzessive schneiden
- Müssen nicht gemäß Ordnung besucht werden



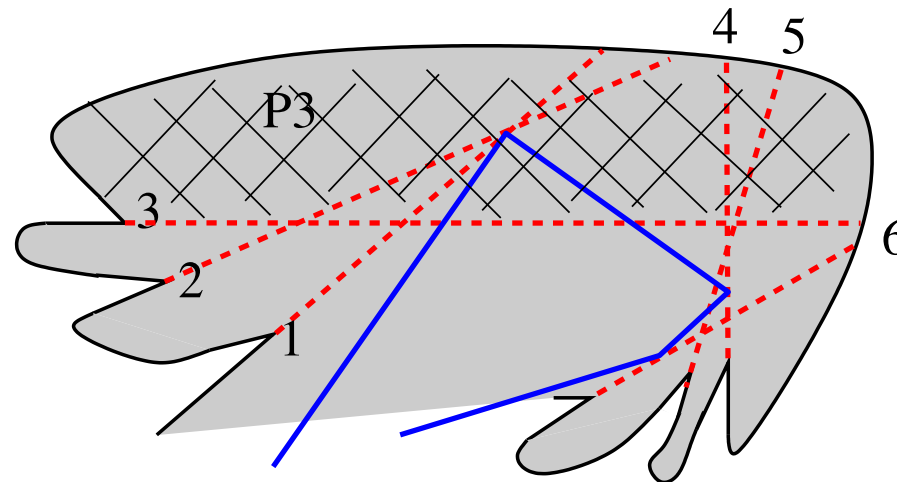
SWR (Allgemeiner Fall)

- Corner Problem!!
- **Def. 1.28** Folge wes. Cuts, die sich sukzessive schneiden
- Müssen nicht gemäß Ordnung besucht werden
- Aber die zugehörigen P_{c_i} !!!



SWR (Allgemeiner Fall)

- Corner Problem!!
- **Def. 1.28** Folge wes. Cuts, die sich sukzessive schneiden
- Müssen nicht gemäß Ordnung besucht werden
- Aber die zugehörigen P_{c_i} !!!



Lemma 1.29

Lemma 1.29

Die SWR besucht die Corner gemäß der Ordnung entlang des Randes!

Lemma 1.29

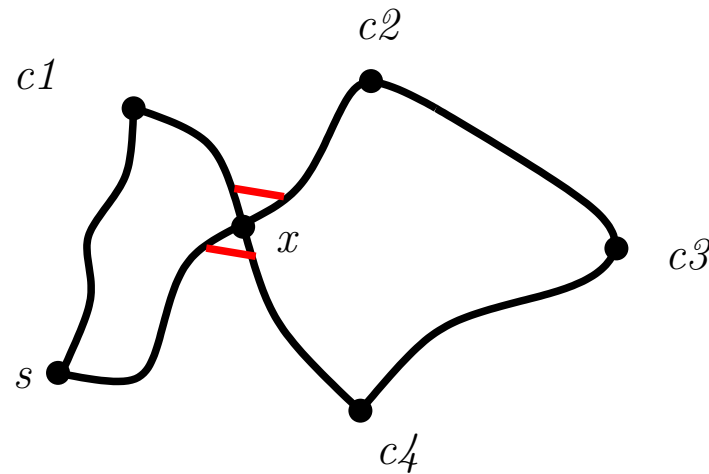
Die SWR besucht die Corner gemäß der Ordnung entlang des Randes!

Beweis: Genauso wie Lemma 1.26!

Lemma 1.29

Die SWR besucht die Corner gemäß der Ordnung entlang des Randes!

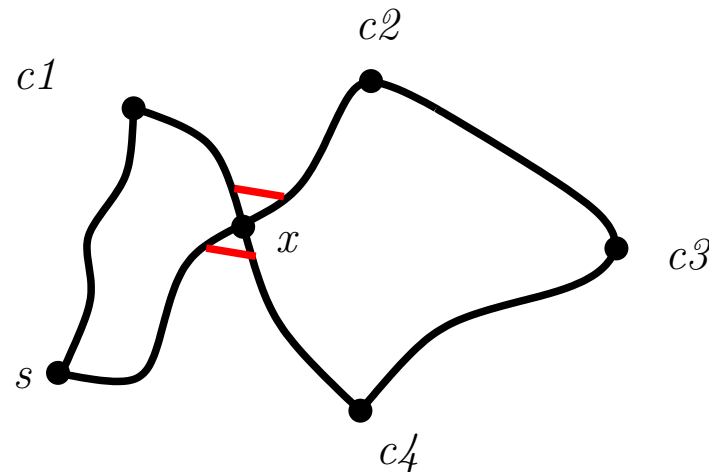
Beweis: Genauso wie Lemma 1.26! Anpassungen im Corner!



Lemma 1.29

Die SWR besucht die Corner gemäß der Ordnung entlang des Randes!

Beweis: Genauso wie Lemma 1.26! Anpassungen im Corner!



Anpassungen innerhalb der Corner: Fehleranfällig!!

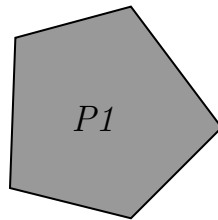
Touring a sequence of polygons (TPP)

Touring a sequence of polygons (TPP)

- Sequenz konvexer Polygone

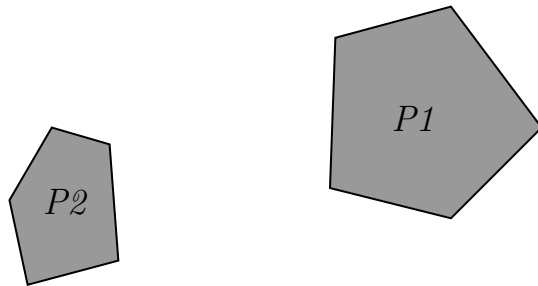
Touring a sequence of polygons (TPP)

- Sequenz konvexer Polygone



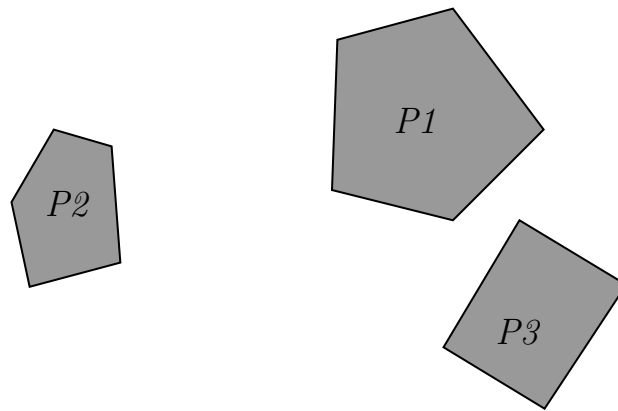
Touring a sequence of polygons (TPP)

- Sequenz konvexer Polygone



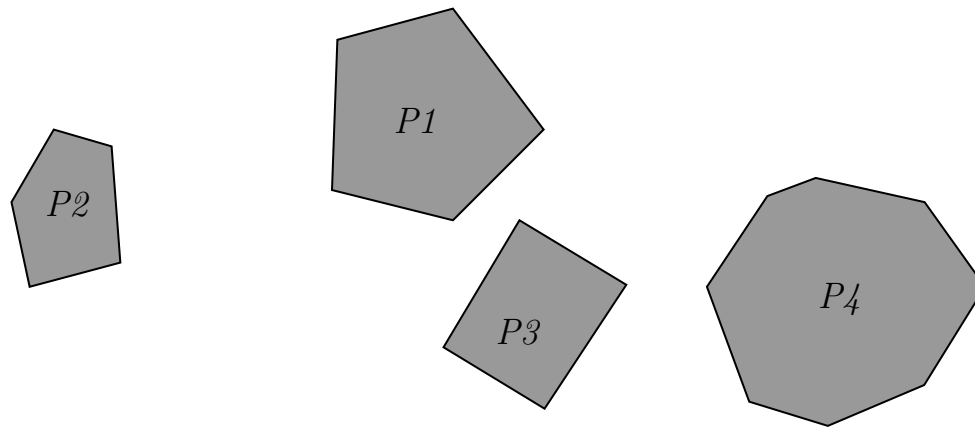
Touring a sequence of polygons (TPP)

- Sequenz konvexer Polygone



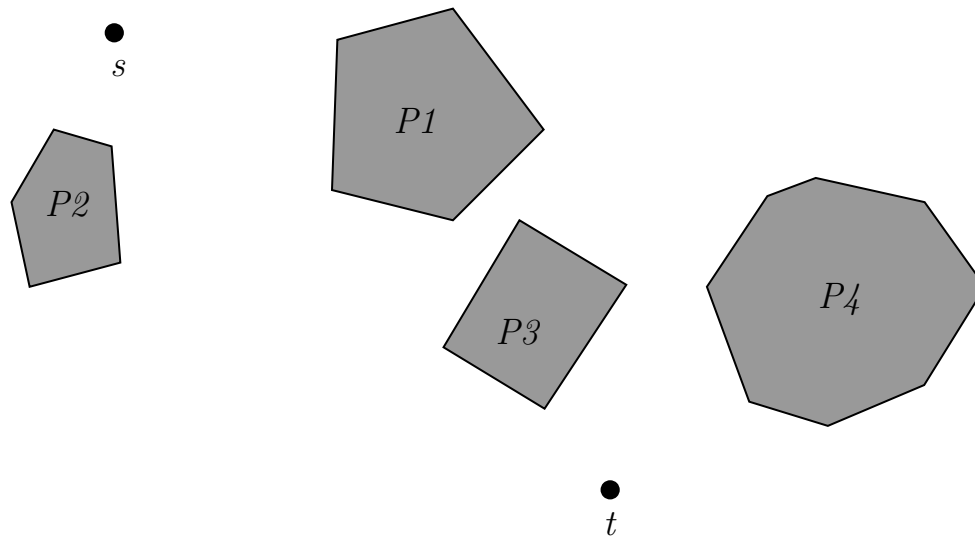
Touring a sequence of polygons (TPP)

- Sequenz konvexer Polygone



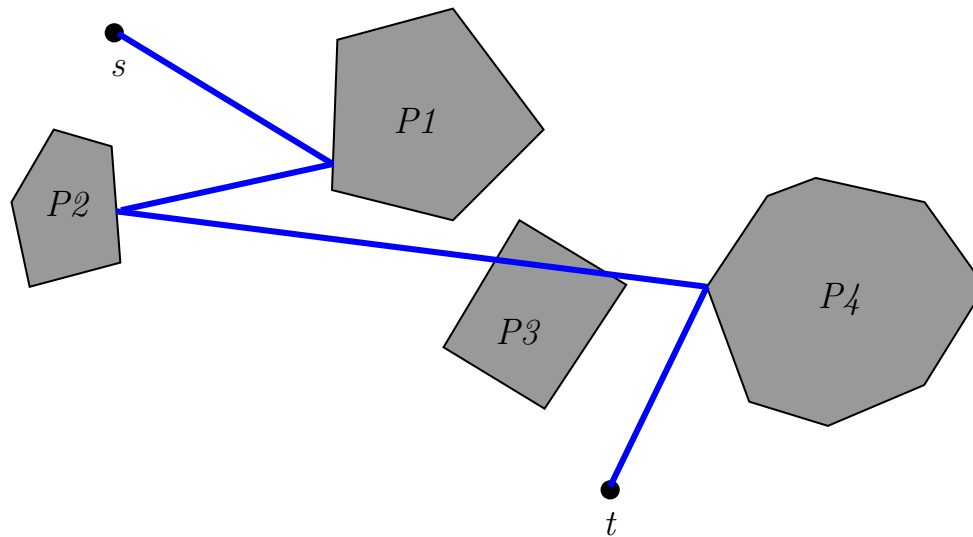
Touring a sequence of polygons (TPP)

- Sequenz konvexer Polygone
- Start s und Ziel t



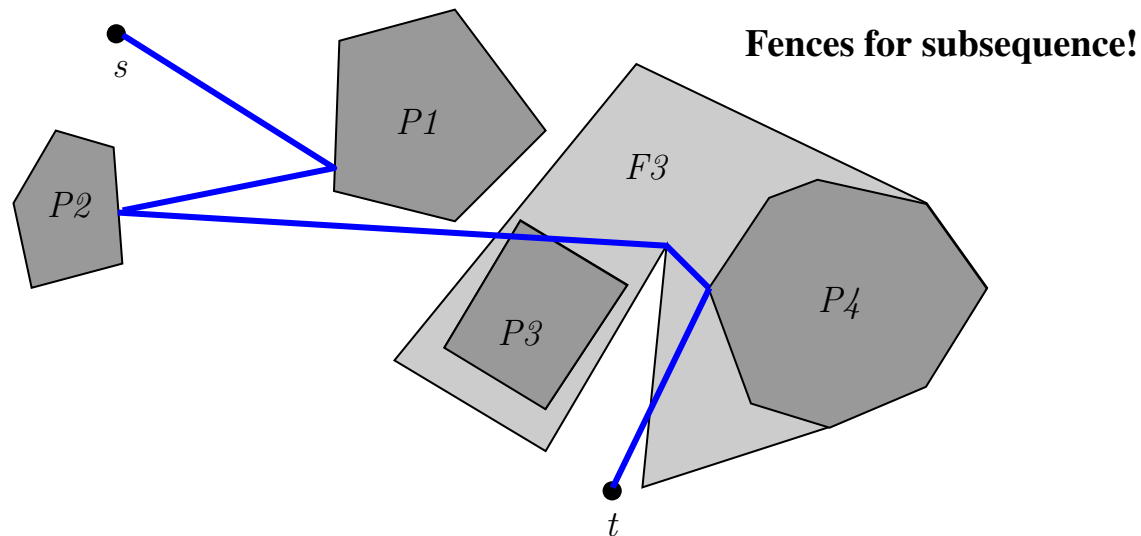
Touring a sequence of polygons (TPP)

- Sequenz konvexer Polygone
- Start s und Ziel t
- Besuche Polygone nach geg. Reihenfolge, Shortest Path



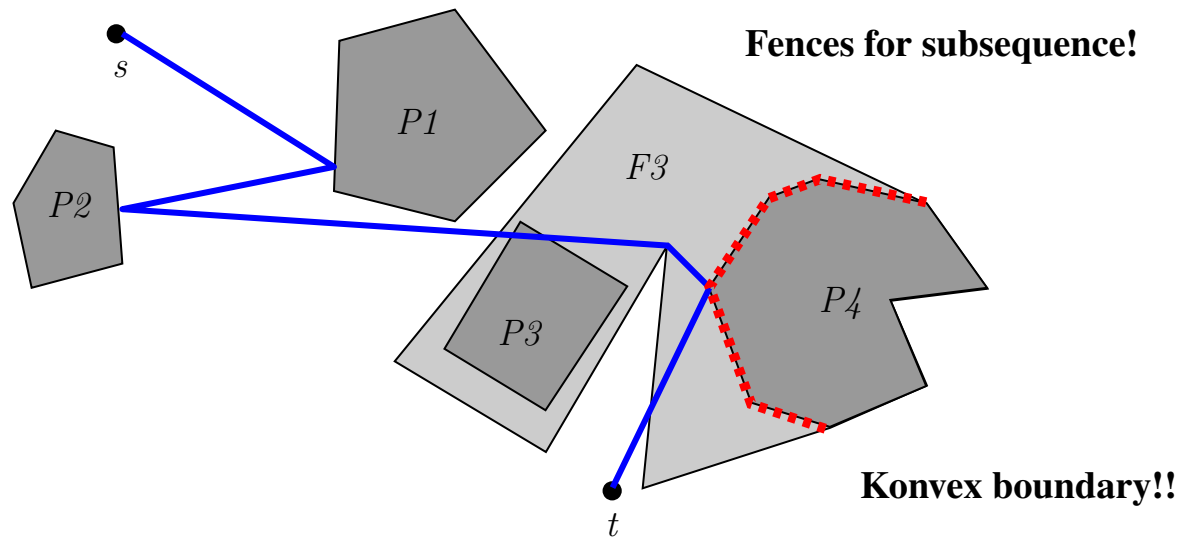
Touring a sequence of polygons (TPP)

- Sequenz konvexer Polygone
- Start s und Ziel t
- Besuche Polygone nach geg. Reihenfolge, Shortest Path



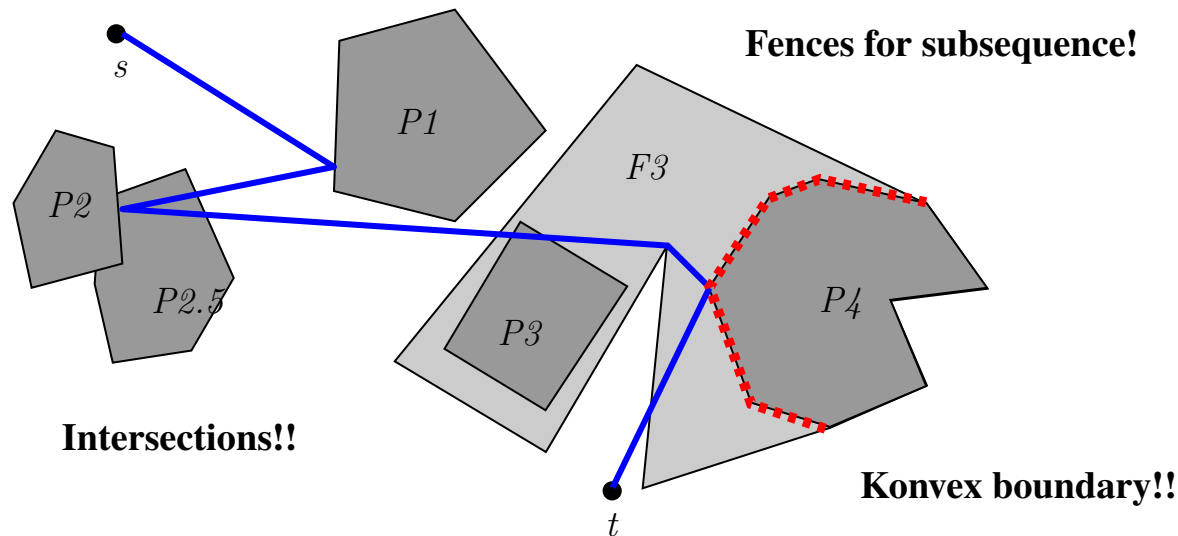
Touring a sequence of polygons (TPP)

- Sequenz konvexer Polygone
- Start s und Ziel t
- Besuche Polygone nach geg. Reihenfolge, Shortest Path



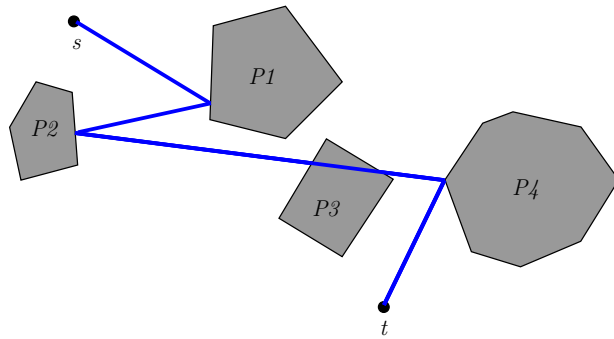
Touring a sequence of polygons (TPP)

- Sequenz konvexer Polygone
- Start s und Ziel t
- Besuche Polygone nach geg. Reihenfolge, Shortest Path

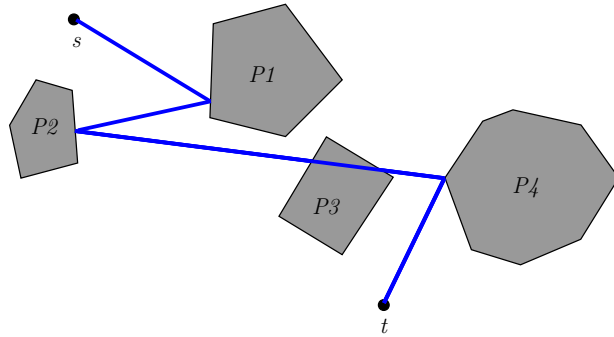


TPP

TPP

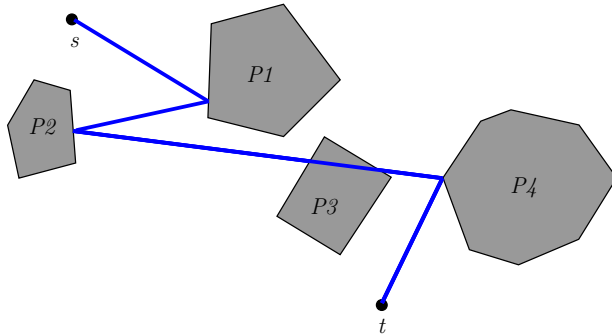


TPP



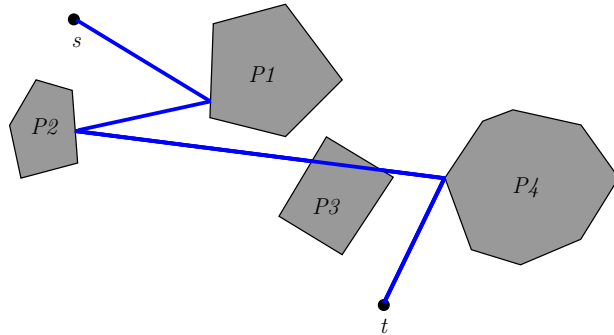
- Einfache Version:
- $O(nk \log \frac{n}{k})$

TPP



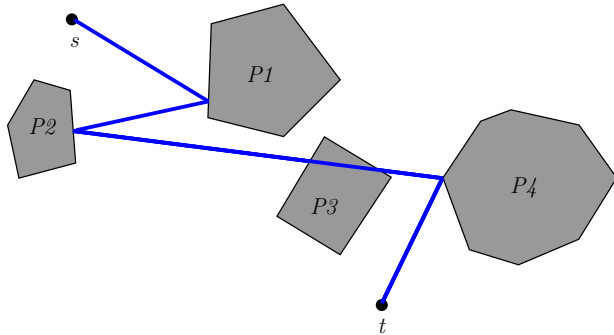
- Einfache Version:
- $O(nk \log \frac{n}{k})$
- Build(Query): $O(nk \log \frac{n}{k})$

TPP



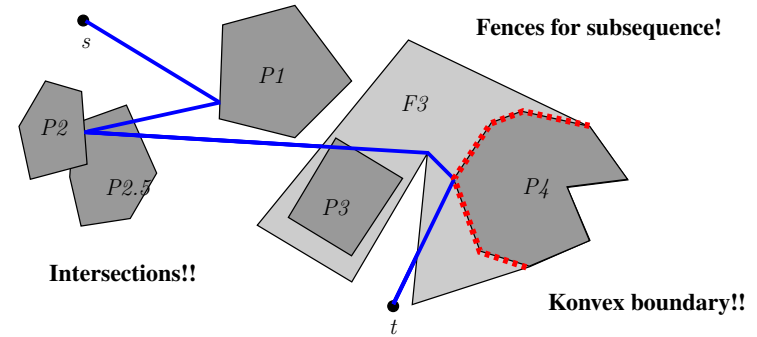
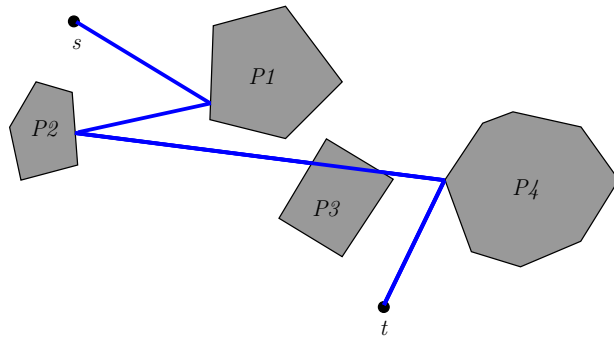
- Einfache Version:
- $O(nk \log \frac{n}{k})$
- Build(Query): $O(nk \log \frac{n}{k})$
- Kompl.: $O(n)$

TPP



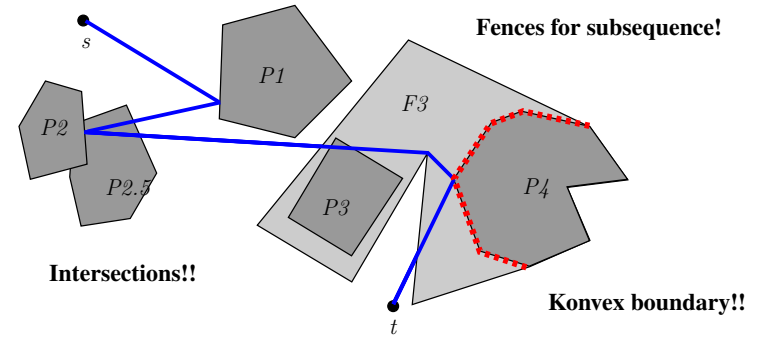
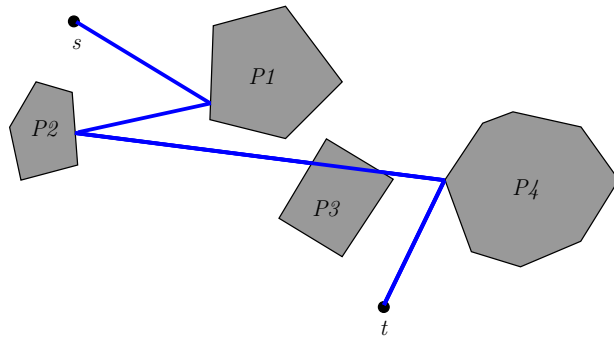
- Einfache Version:
- $O(nk \log \frac{n}{k})$
- Build(Query): $O(nk \log \frac{n}{k})$
- Kompl.: $O(n)$
- Query (festes s): $O(k \log \frac{n}{k})$

TPP



- Einfache Version:
- $O(nk \log \frac{n}{k})$
- Build(Query): $O(nk \log \frac{n}{k})$
- Kompl.: $O(n)$
- Query (festes s): $O(k \log \frac{n}{k})$

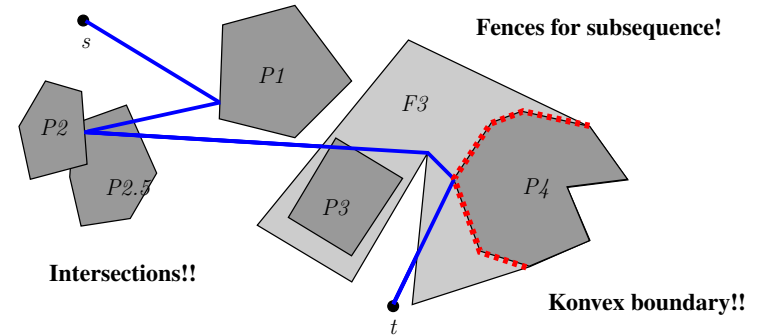
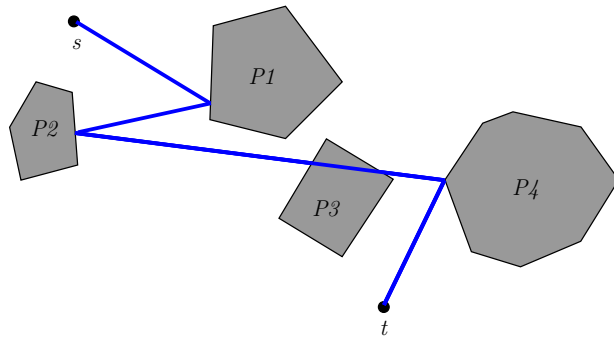
TPP



- Einfache Version:
- $O(nk \log \frac{n}{k})$
- Build(Query): $O(nk \log \frac{n}{k})$
- Kompl.: $O(n)$
- Query (festes s): $O(k \log \frac{n}{k})$

- Allgemeine Version:
- Zäune, Rand konvex, etc.

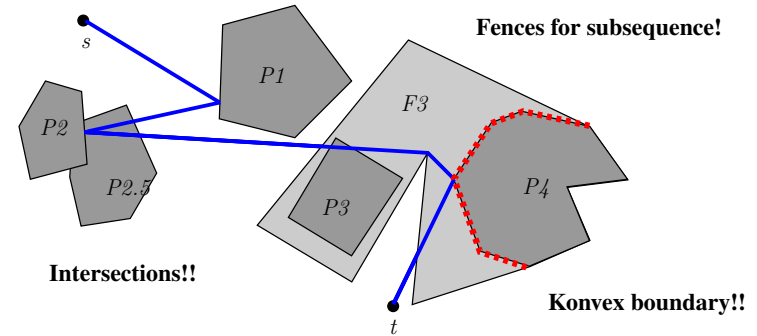
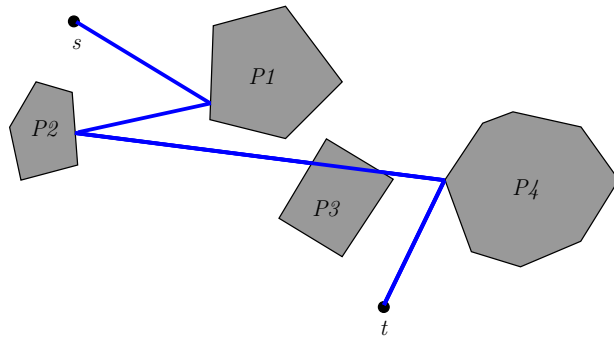
TPP



- Einfache Version:
- $O(nk \log \frac{n}{k})$
- Build(Query): $O(nk \log \frac{n}{k})$
- Kompl.: $O(n)$
- Query (festes s): $O(k \log \frac{n}{k})$

- Allgemeine Version:
- Zäune, Rand konvex, etc.
- $O(nk^2 \log n)$

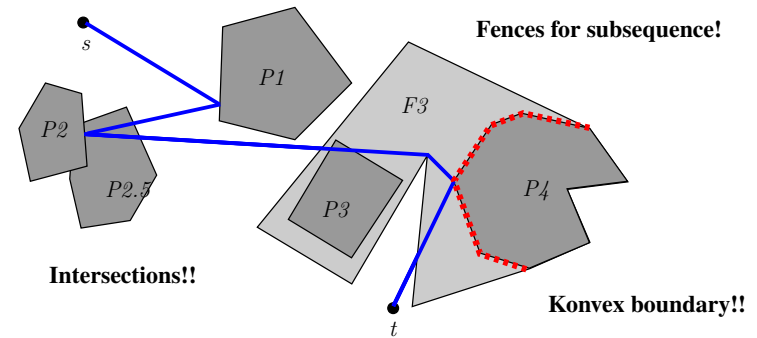
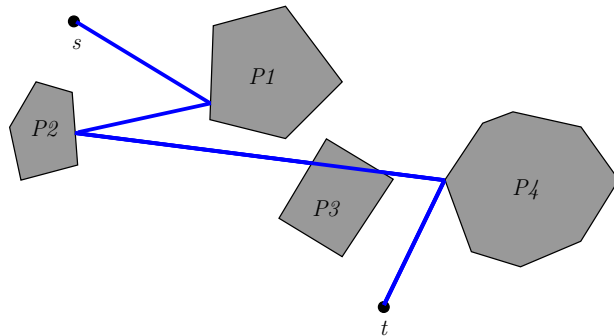
TPP



- Einfache Version:
- $O(nk \log \frac{n}{k})$
- Build(Query): $O(nk \log \frac{n}{k})$
- Kompl.: $O(n)$
- Query (festes s): $O(k \log \frac{n}{k})$

- Allgemeine Version:
- Zäune, Rand konvex, etc.
- $O(nk^2 \log n)$
- Build(Query): $O(nk^2 \log n)$

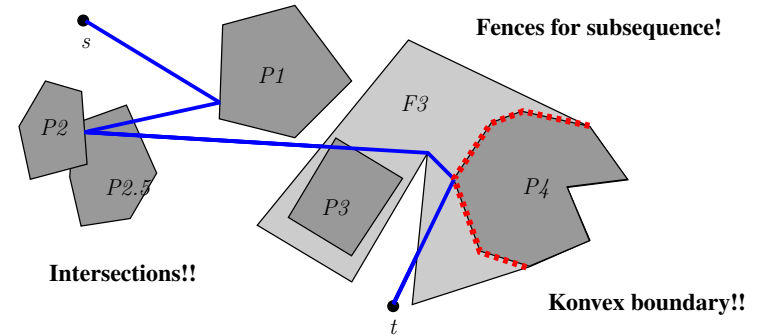
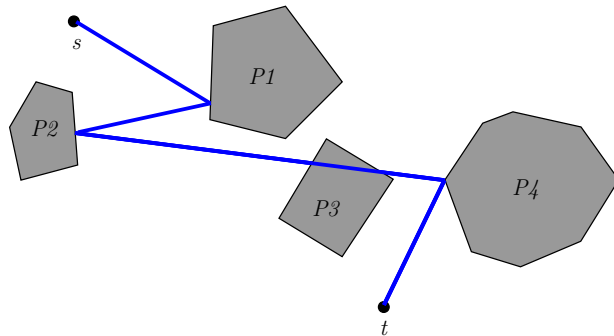
TPP



- Einfache Version:
- $O(nk \log \frac{n}{k})$
- Build(Query): $O(nk \log \frac{n}{k})$
- Kompl.: $O(n)$
- Query (festes s): $O(k \log \frac{n}{k})$

- Allgemeine Version:
- Zäune, Rand konvex, etc.
- $O(nk^2 \log n)$
- Build(Query): $O(nk^2 \log n)$
- Kompl.: $O(nk)$

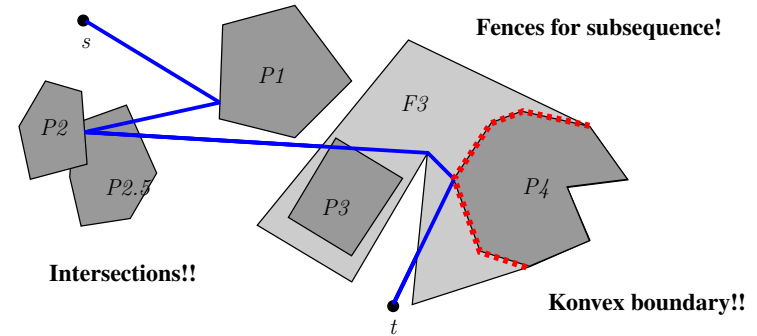
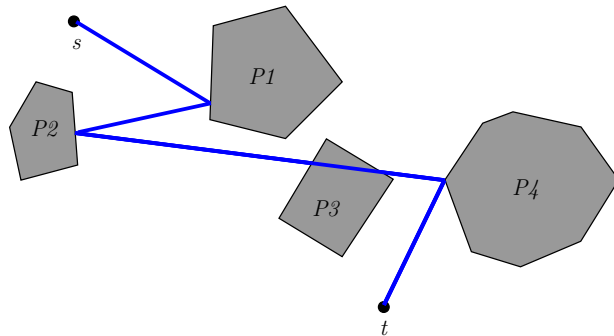
TPP



- Einfache Version:
- $O(nk \log \frac{n}{k})$
- Build(Query): $O(nk \log \frac{n}{k})$
- Kompl.: $O(n)$
- Query (festes s): $O(k \log \frac{n}{k})$

- Allgemeine Version:
- Zäune, Rand konvex, etc.
- $O(nk^2 \log n)$
- Build(Query): $O(nk^2 \log n)$
- Kompl.: $O(nk)$
- Query (festes s): $O(k \log n + m)$

TPP



- Einfache Version:
- $O(nk \log \frac{n}{k})$
- Build(Query): $O(nk \log \frac{n}{k})$
- Kompl.: $O(n)$
- Query (festes s): $O(k \log \frac{n}{k})$

- Allgemeine Version:
- Zäune, Rand konvex, etc.
- $O(nk^2 \log n)$
- Build(Query): $O(nk^2 \log n)$
- Kompl.: $O(nk)$
- Query (festes s): $O(k \log n + m)$

Ergebnisse von: Dror, Efrat, Lubiw, Mitchell 2003!!

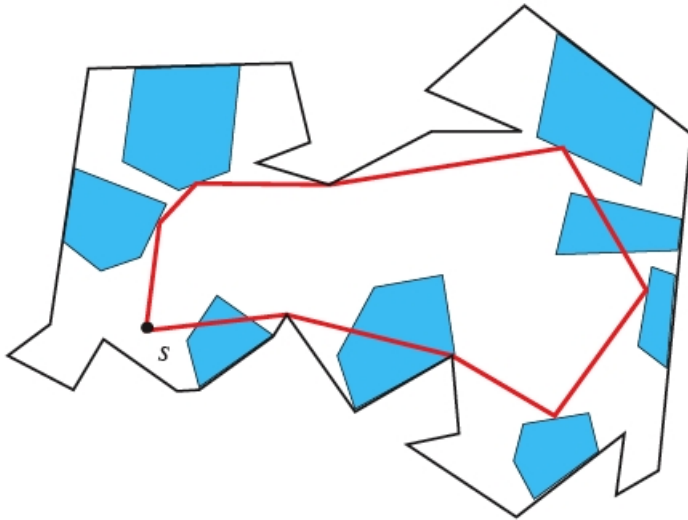
Anwendung: Safari-Problem

Anwendung: Safari-Problem

Kürzester Weg,
innerhalb eines Polygons, der eine
Menge von schnittfreien Polygonen
besucht.

Anwendung: Safari-Problem

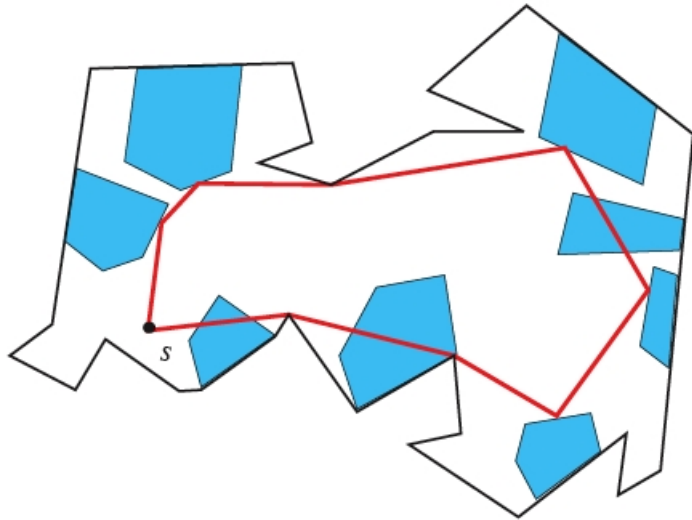
Kürzester Weg,
innerhalb eines Polygons, der eine
Menge von schnittfreien Polygonen
besucht.



Ordnung entlang des Randes!

Anwendung: Safari-Problem

Kürzester Weg,
innerhalb eines Polygones, der eine
Menge von schnittfreien Polygonen
besucht.



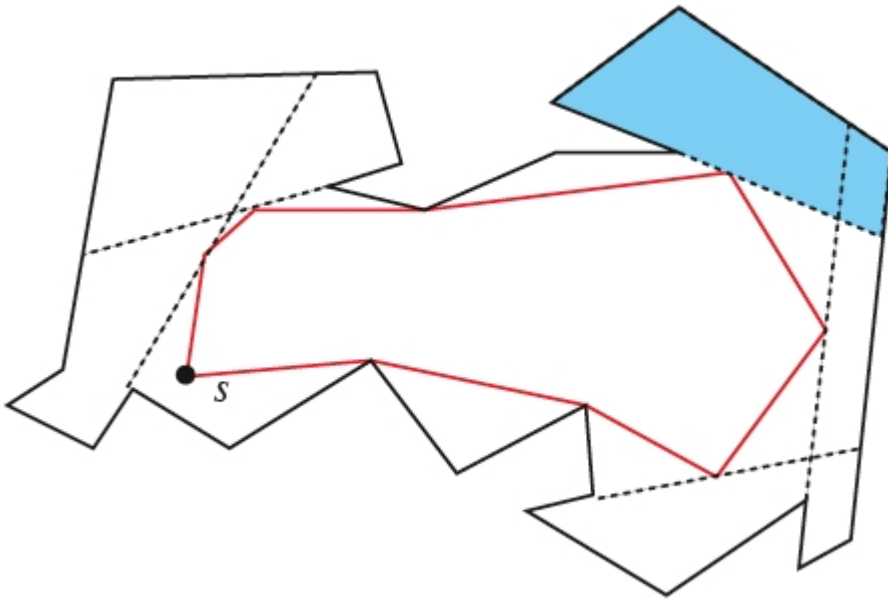
- $O(n^2)$ '92
- $O(n^3)$ '94
- $O(n^3)$ Tan Hirata '01
- $O(n^2 \log n)$ jetzt! Anpassen!!

Ordnung entlang des Randes!

Anwendung: SWR

Anwendung: SWR

Wesentlichen Teile!

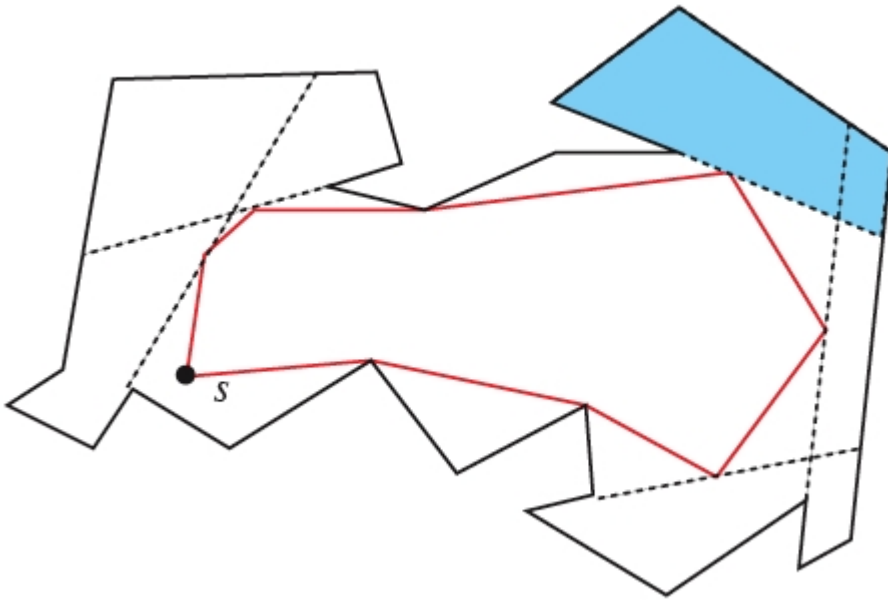


Anwendung: SWR

Wesentlichen Teile!

Ordnung entlang des Randes!

Ein gemeinsamer Zaun! Schnitte!

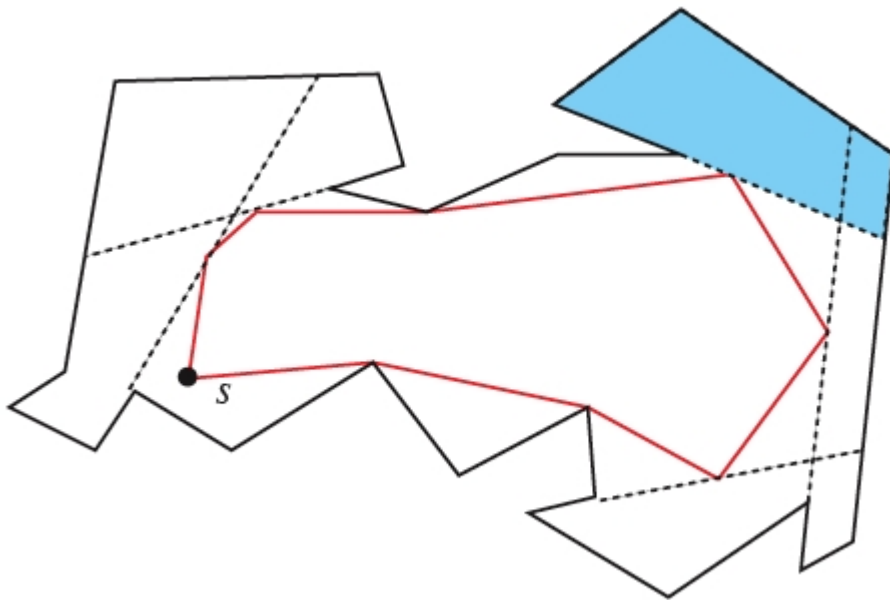


Anwendung: SWR

Wesentlichen Teile!

Ordnung entlang des Randes!

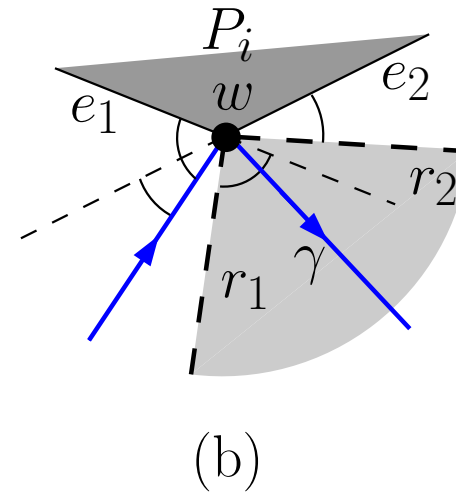
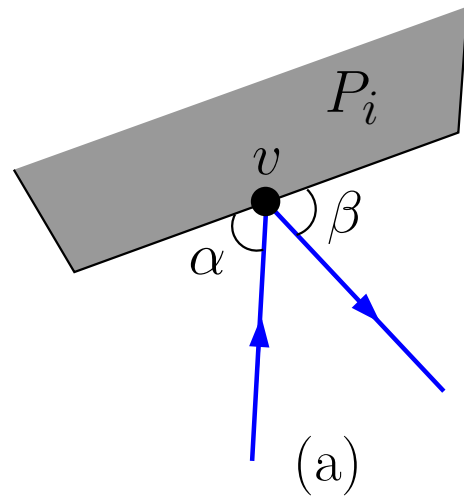
Ein gemeinsamer Zaun! Schnitte!



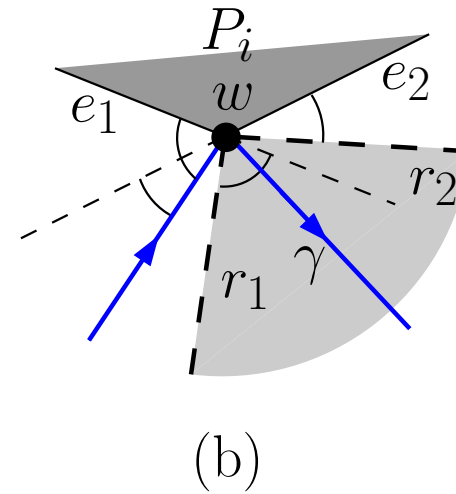
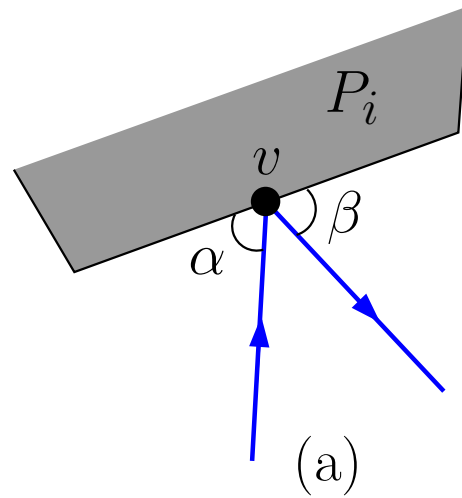
- $O(n^4)$ '91
- $O(n^4)$ Tan et al. '99
- $O(n^3 \log n)$ jetzt!

Lokale Eigenschaften: **Lemma 1.31(i)**

Lokale Eigenschaften: **Lemma 1.31(i)**

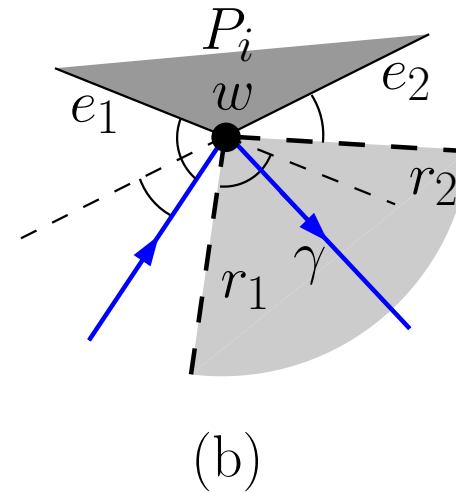
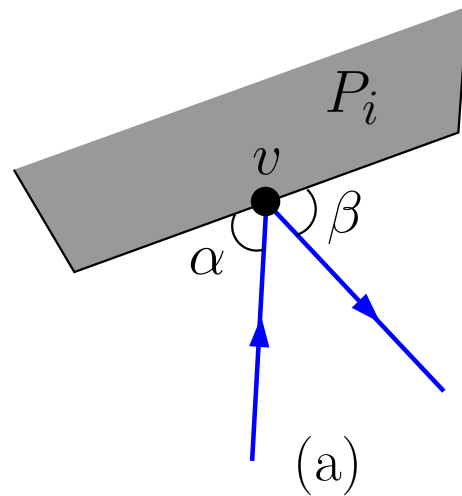


Lokale Eigenschaften: **Lemma 1.31(i)**



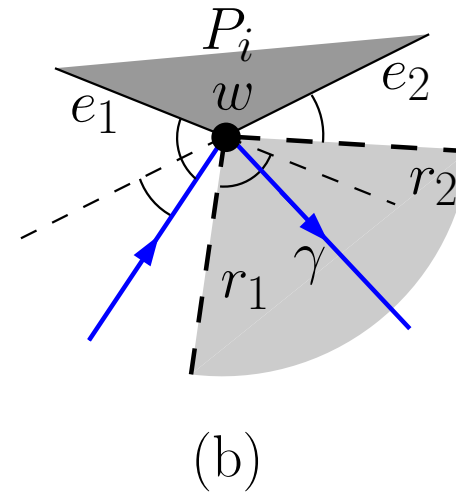
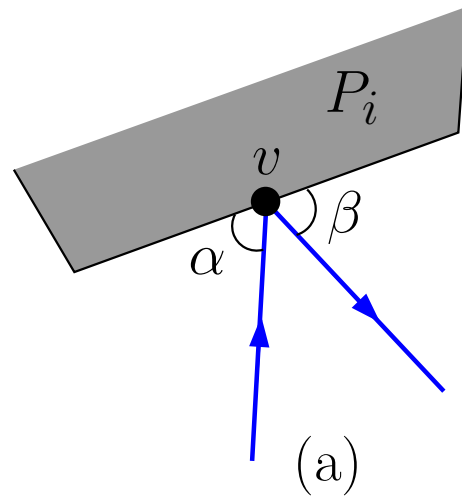
- (a) Reflektion an Kante e :

Lokale Eigenschaften: **Lemma 1.31(i)**



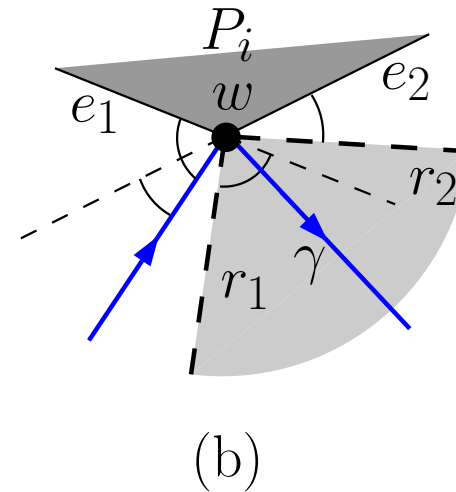
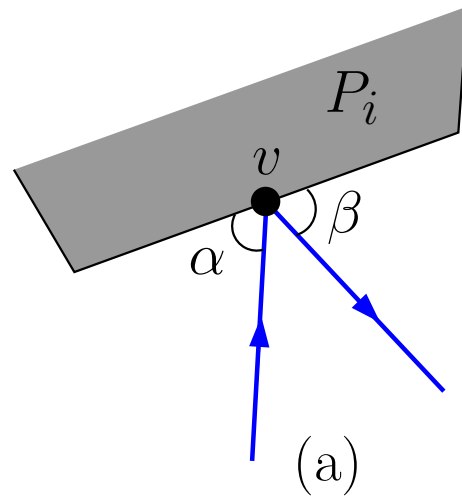
- (a) Reflektion an Kante e : $\alpha = \beta$

Lokale Eigenschaften: **Lemma 1.31(i)**



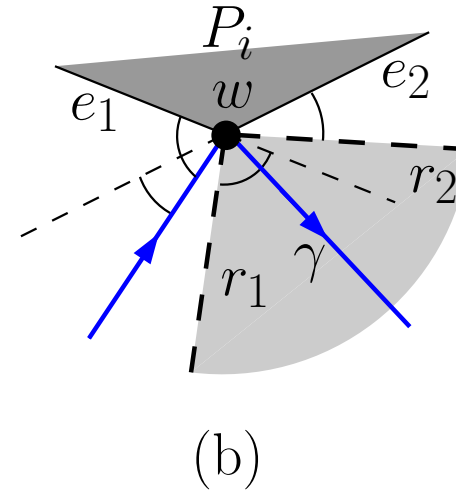
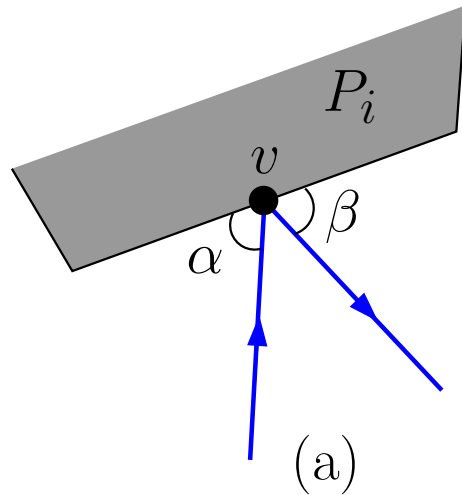
- (a) Reflektion an Kante e : $\alpha = \beta$
- (b) Reflektion an Knoten w :

Lokale Eigenschaften: **Lemma 1.31(i)**



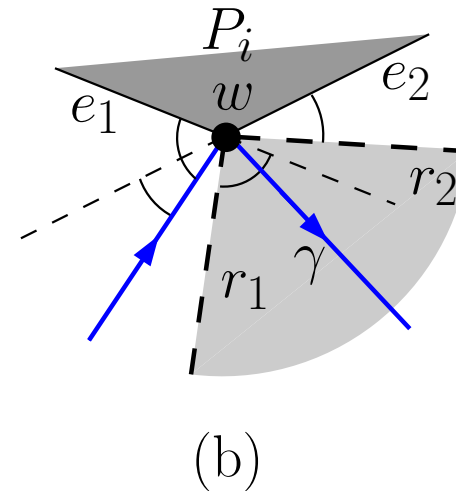
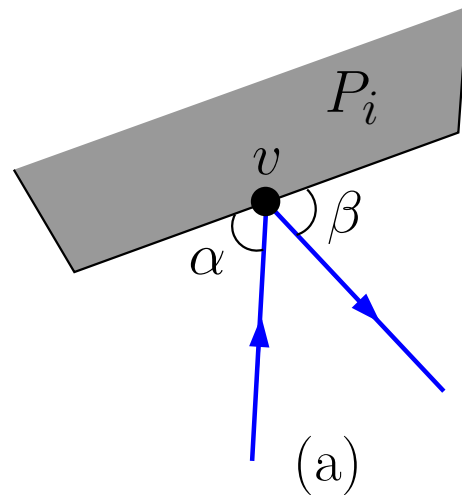
- (a) Reflektion an Kante e : $\alpha = \beta$
- (b) Reflektion an Knoten w : Innerhalb Winkelbereich γ

Lokale Eigenschaften: **Lemma 1.31(i)**



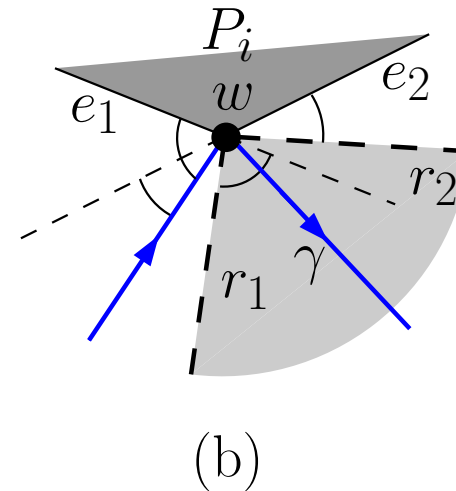
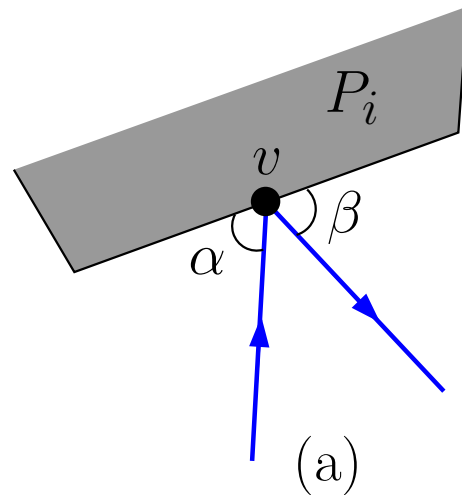
- (a) Reflektion an Kante e : $\alpha = \beta$
- (b) Reflektion an Knoten w : Innerhalb Winkelbereich γ
- Sonst nicht optimal

Lokale Eigenschaften: **Lemma 1.31(i)**



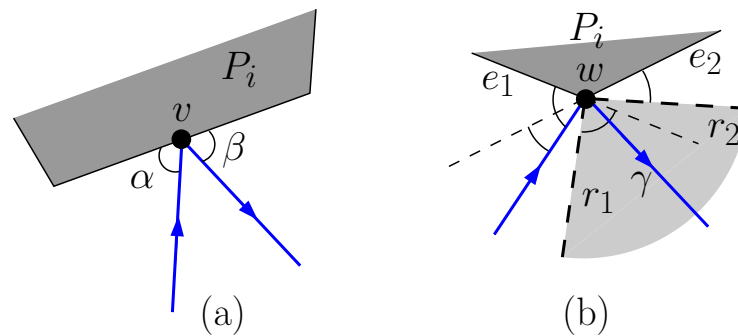
- (a) Reflektion an Kante e : $\alpha = \beta$
- (b) Reflektion an Knoten w : Innerhalb Winkelbereich γ
- Sonst nicht optimal
- Lemma 1.31: Local optimality \Rightarrow Global optimality

Lokale Eigenschaften: **Lemma 1.31(i)**



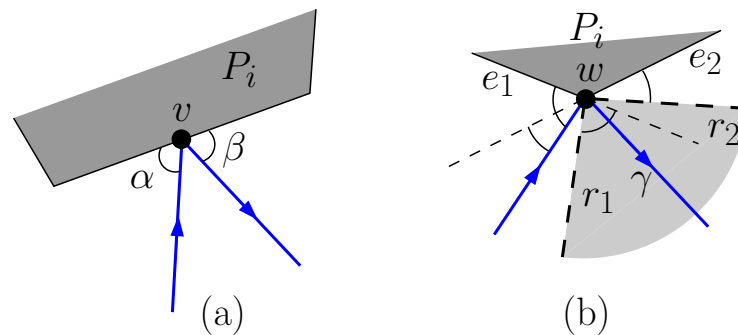
- (a) Reflektion an Kante e : $\alpha = \beta$
- (b) Reflektion an Knoten w : Innerhalb Winkelbereich γ
- Sonst nicht optimal
- Lemma 1.31: Local optimality \Rightarrow Global optimality
- Aufgabe: Berechne sukzessive lokal optimalen Pfad

Lemma 1.31 (ii) und (iii)



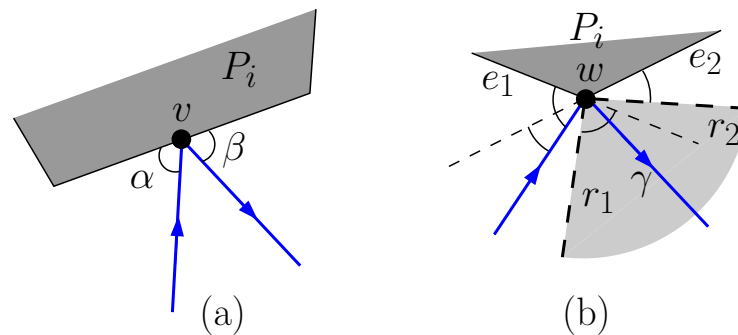
Lemma 1.31 (ii) und (iii)

- Bezeichnung $\pi_i(p)$:



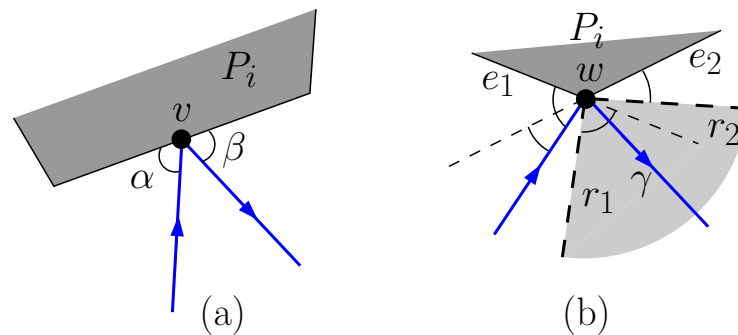
Lemma 1.31 (ii) und (iii)

- Bezeichnung $\pi_i(p)$: lokal opt. Weg von s über P_1, \dots, P_i bis p



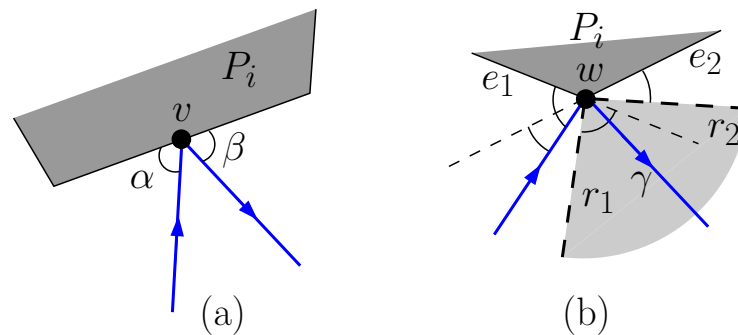
Lemma 1.31 (ii) und (iii)

- Bezeichnung $\pi_i(p)$: lokal opt. Weg von s über P_1, \dots, P_i bis p
- Lokal optimal: Erfüllt überall die Bedingungen (a) oder (b)



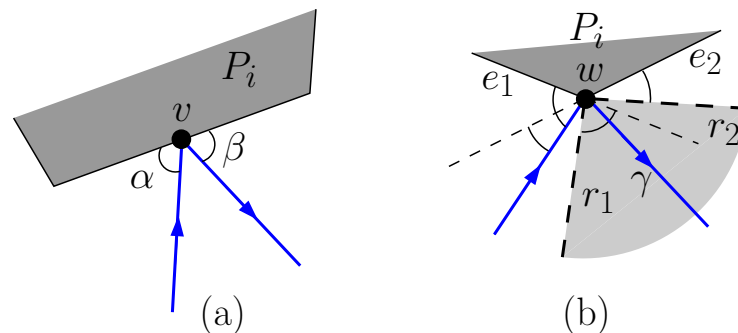
Lemma 1.31 (ii) und (iii)

- Bezeichnung $\pi_i(p)$: lokal opt. Weg von s über P_1, \dots, P_i bis p
- Lokal optimal: Erfüllt überall die Bedingungen (a) oder (b)
- Über Ecken und Kanten, lokal nicht verkürzbar



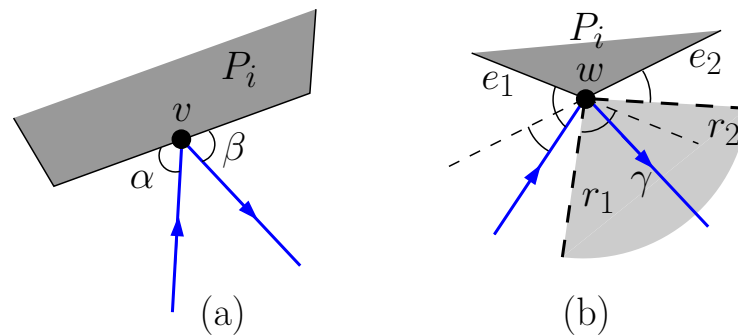
Lemma 1.31 (ii) und (iii)

- Bezeichnung $\pi_i(p)$: lokal opt. Weg von s über P_1, \dots, P_i bis p
- Lokal optimal: Erfüllt überall die Bedingungen (a) oder (b)
- Über Ecken und Kanten, lokal nicht verkürzbar
- (ii): $\pi_i(p)$ ist stets eindeutig! (Konstruktiv!)



Lemma 1.31 (ii) und (iii)

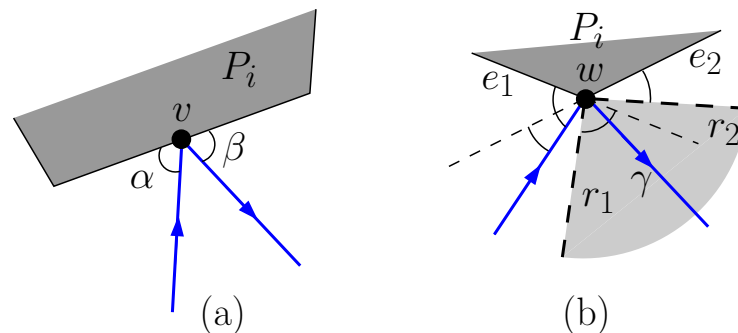
- Bezeichnung $\pi_i(p)$: lokal opt. Weg von s über P_1, \dots, P_i bis p
- Lokal optimal: Erfüllt überall die Bedingungen (a) oder (b)
- Über Ecken und Kanten, lokal nicht verkürzbar
- (ii): $\pi_i(p)$ ist stets eindeutig! (Konstruktiv!)
- (iii): $\pi_i(p)$ ist auch global optimal!



Lemma 1.31 (ii) und (iii)

- Bezeichnung $\pi_i(p)$: lokal opt. Weg von s über P_1, \dots, P_i bis p
- Lokal optimal: Erfüllt überall die Bedingungen (a) oder (b)
- Über Ecken und Kanten, lokal nicht verkürzbar
- (ii): $\pi_i(p)$ ist stets eindeutig! (Konstruktiv!)
- (iii): $\pi_i(p)$ ist auch global optimal!

Beweis (iii): Jeder global optimale Weg muss lokal optimal sein!



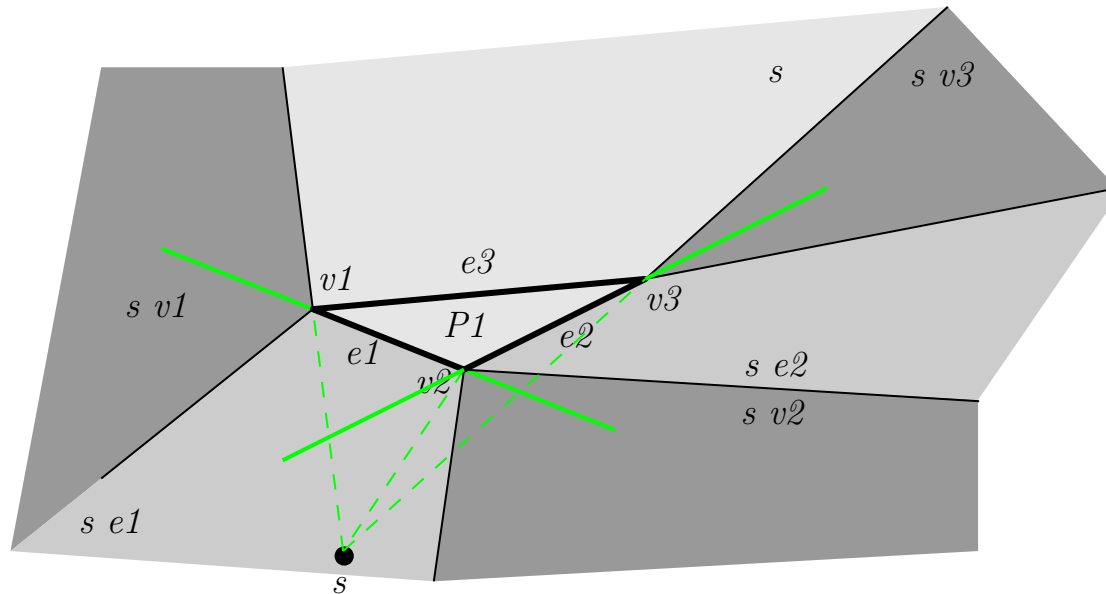
Full comb. shortest path map: Beispiel

Full comb. shortest path map: Beispiel

- Fixiere den Startpunkt s

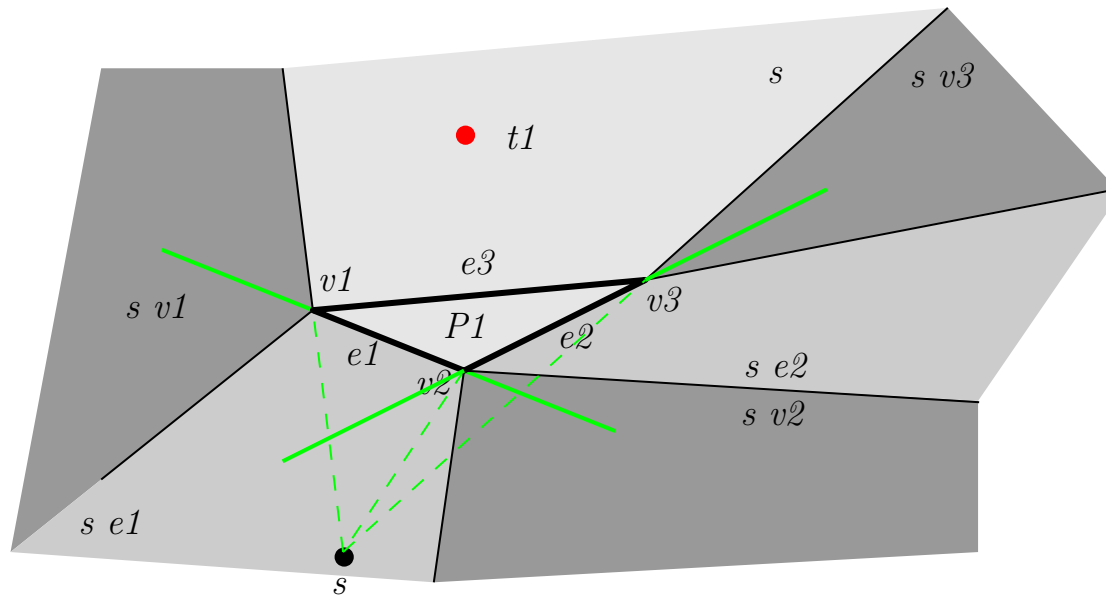
Full comb. shortest path map: Beispiel

- Fixiere den Startpunkt s
- Teile die Ebene in Regionen ein:
- *Kombinatorisch* gleiche Kürzeste Wege



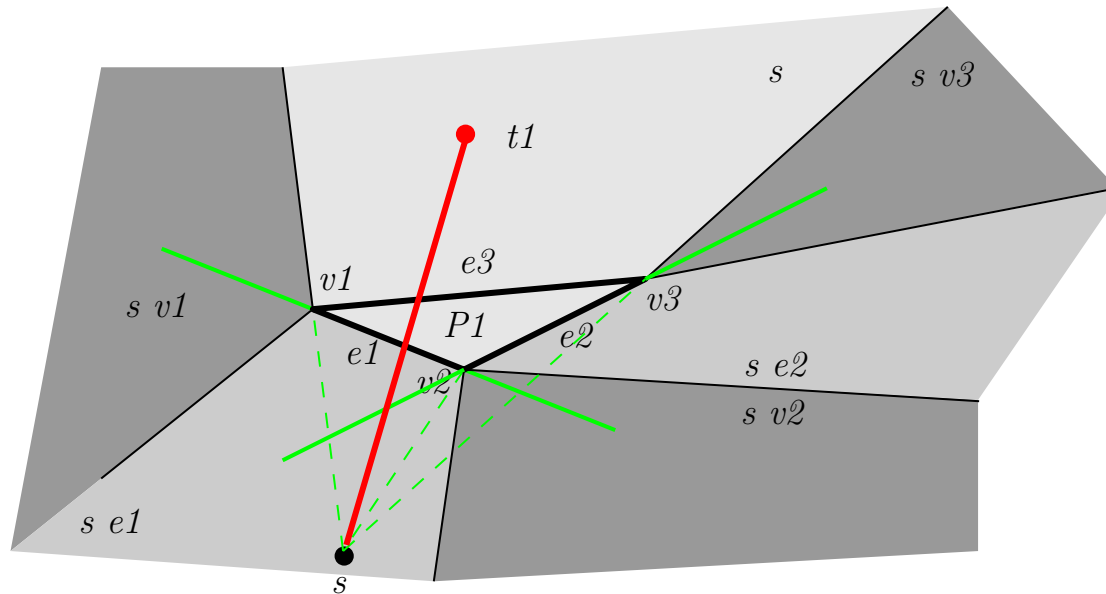
Full comb. shortest path map: Beispiel

- Fixiere den Startpunkt s
- Teile die Ebene in Regionen ein:
- *Kombinatorisch* gleiche Kürzeste Wege



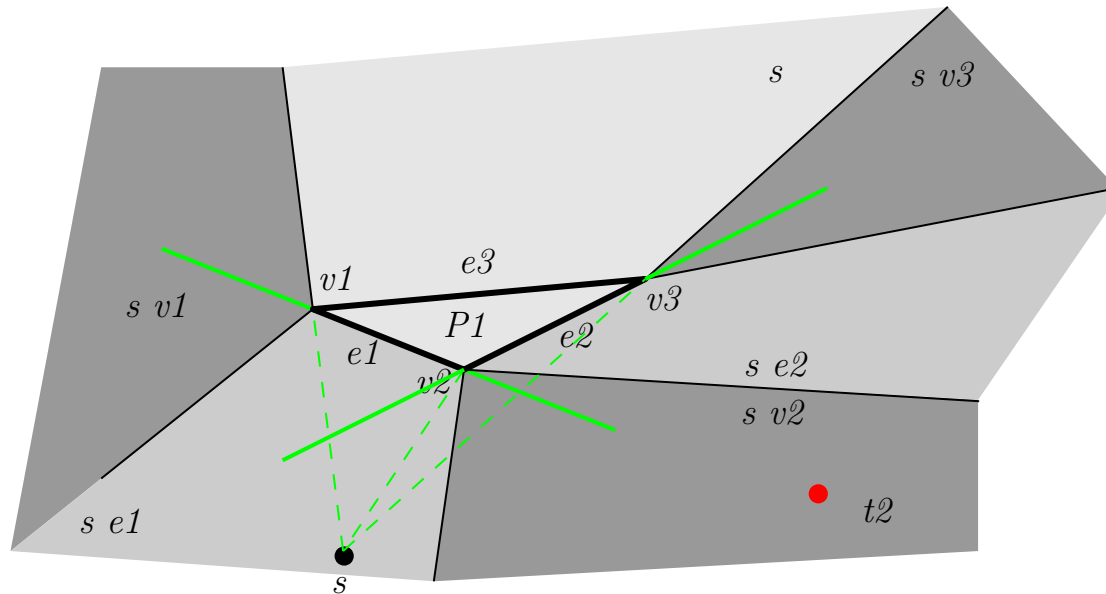
Full comb. shortest path map: Beispiel

- Fixiere den Startpunkt s
- Teile die Ebene in Regionen ein:
- *Kombinatorisch* gleiche Kürzeste Wege



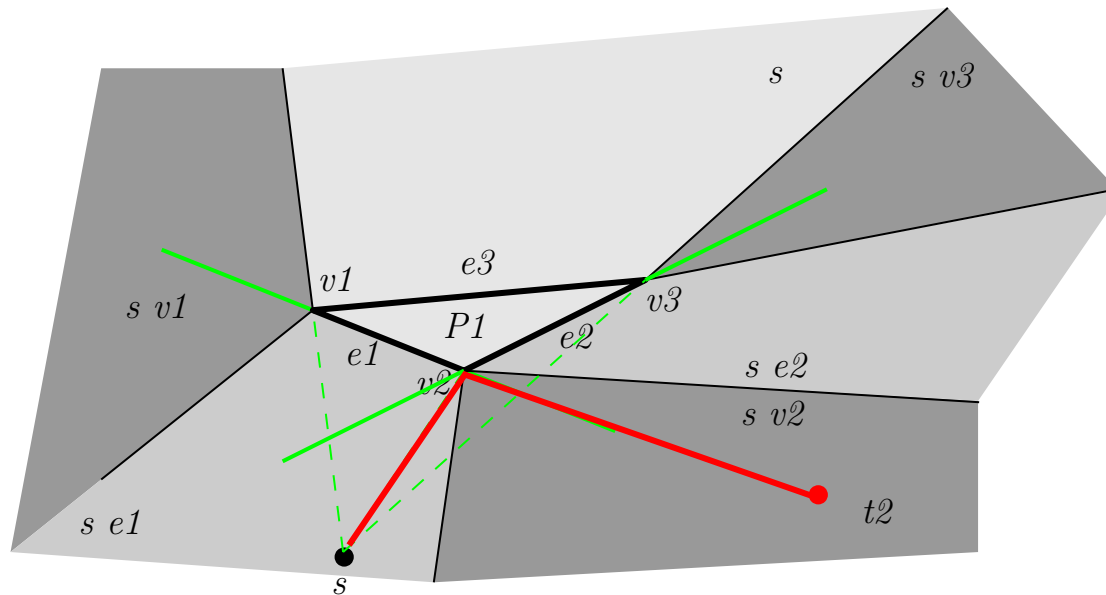
Full comb. shortest path map: Beispiel

- Fixiere den Startpunkt s
- Teile die Ebene in Regionen ein:
- *Kombinatorisch* gleiche Kürzeste Wege



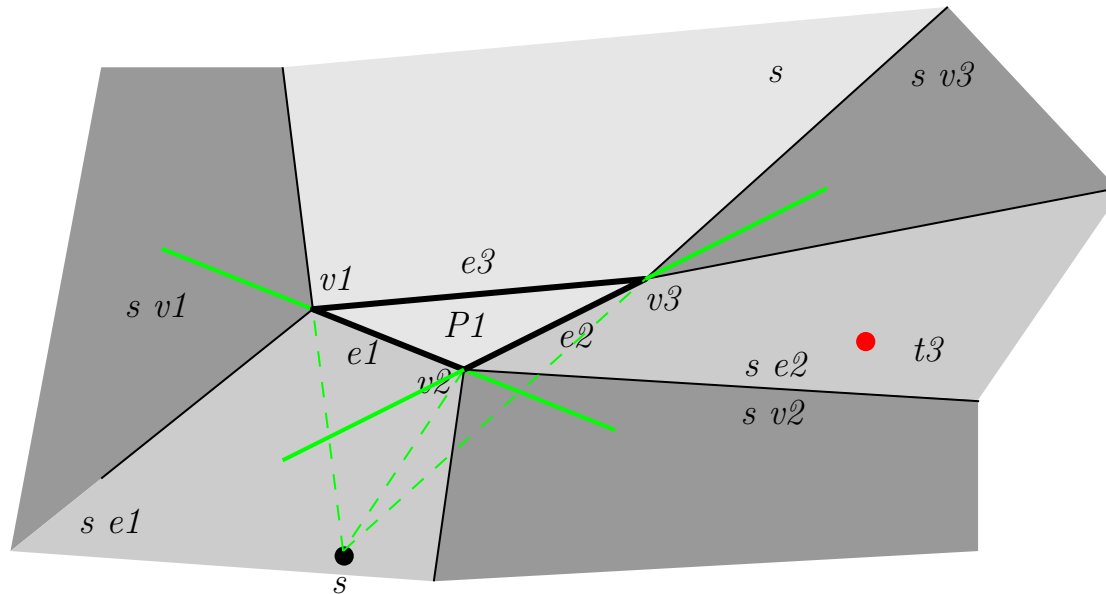
Full comb. shortest path map: Beispiel

- Fixiere den Startpunkt s
- Teile die Ebene in Regionen ein:
- *Kombinatorisch* gleiche Kürzeste Wege



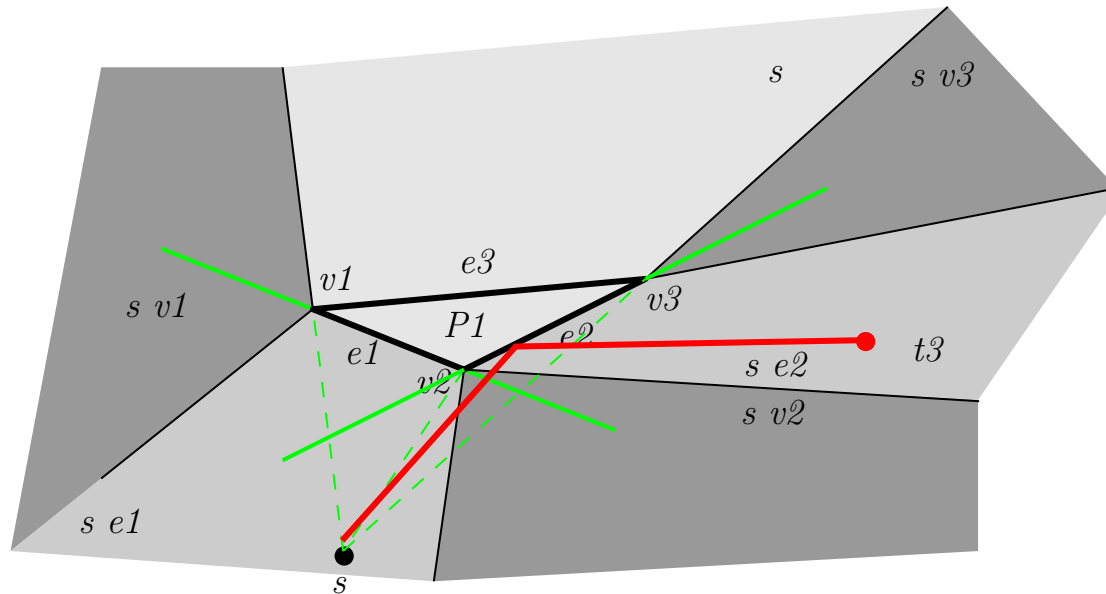
Full comb. shortest path map: Beispiel

- Fixiere den Startpunkt s
- Teile die Ebene in Regionen ein:
- *Kombinatorisch* gleiche Kürzeste Wege



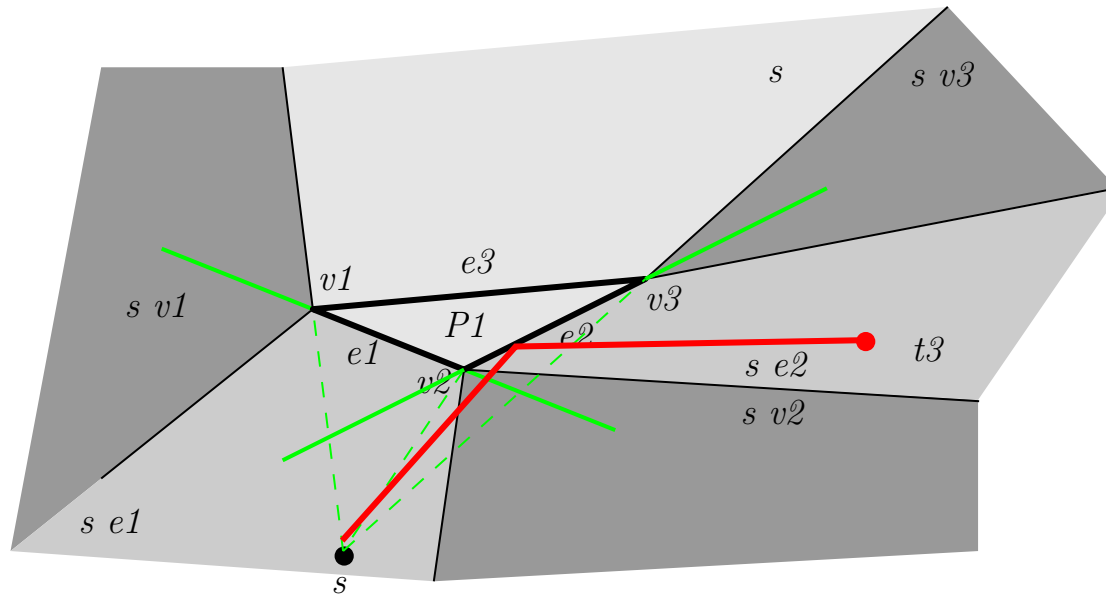
Full comb. shortest path map: Beispiel

- Fixiere den Startpunkt s
- Teile die Ebene in Regionen ein:
- *Kombinatorisch* gleiche Kürzeste Wege



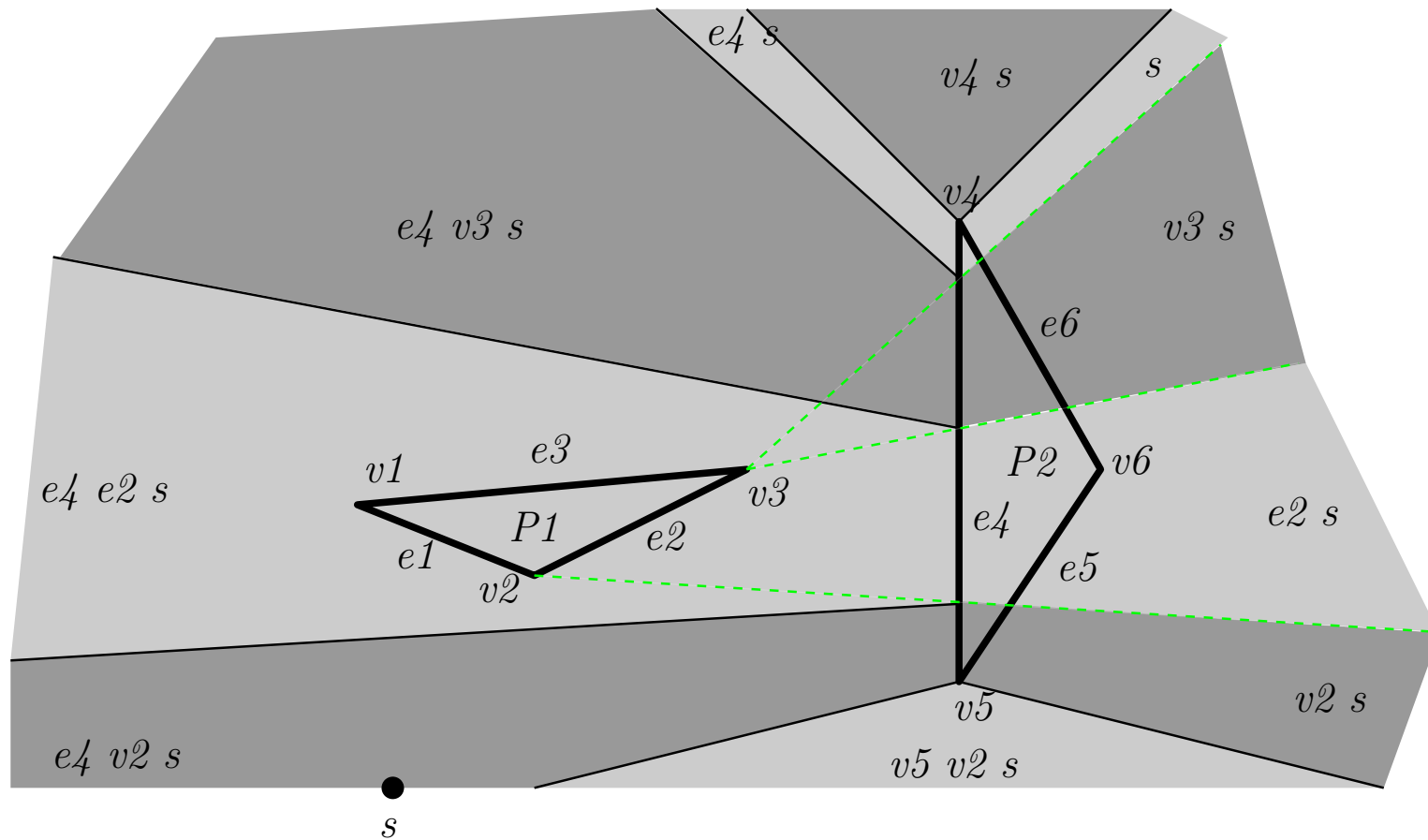
Full comb. shortest path map: Beispiel

- Fixiere den Startpunkt s
- Teile die Ebene in Regionen ein:
- *Kombinatorisch* gleiche Kürzeste Wege

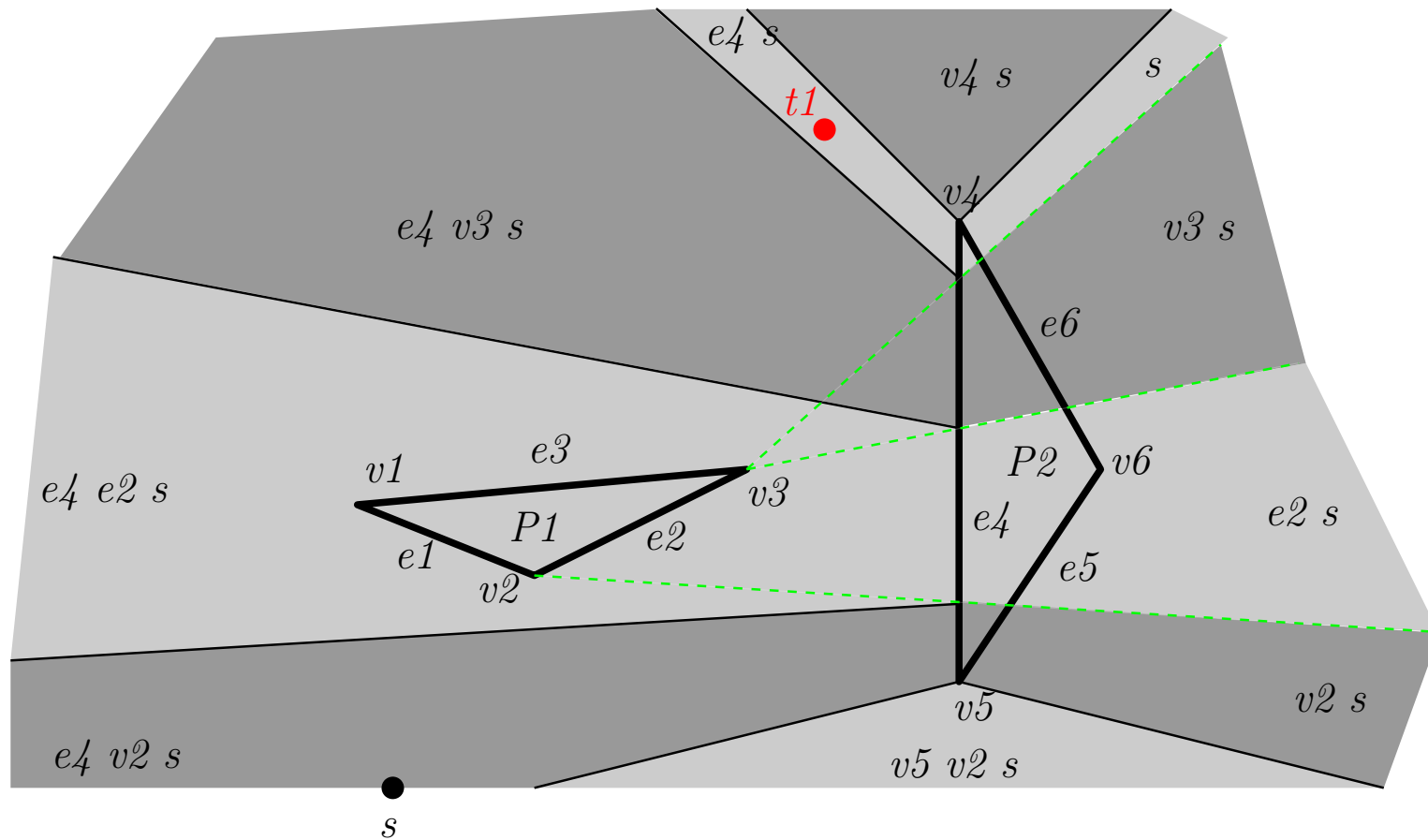


Full comb. shortest path map: Zwei Polygone

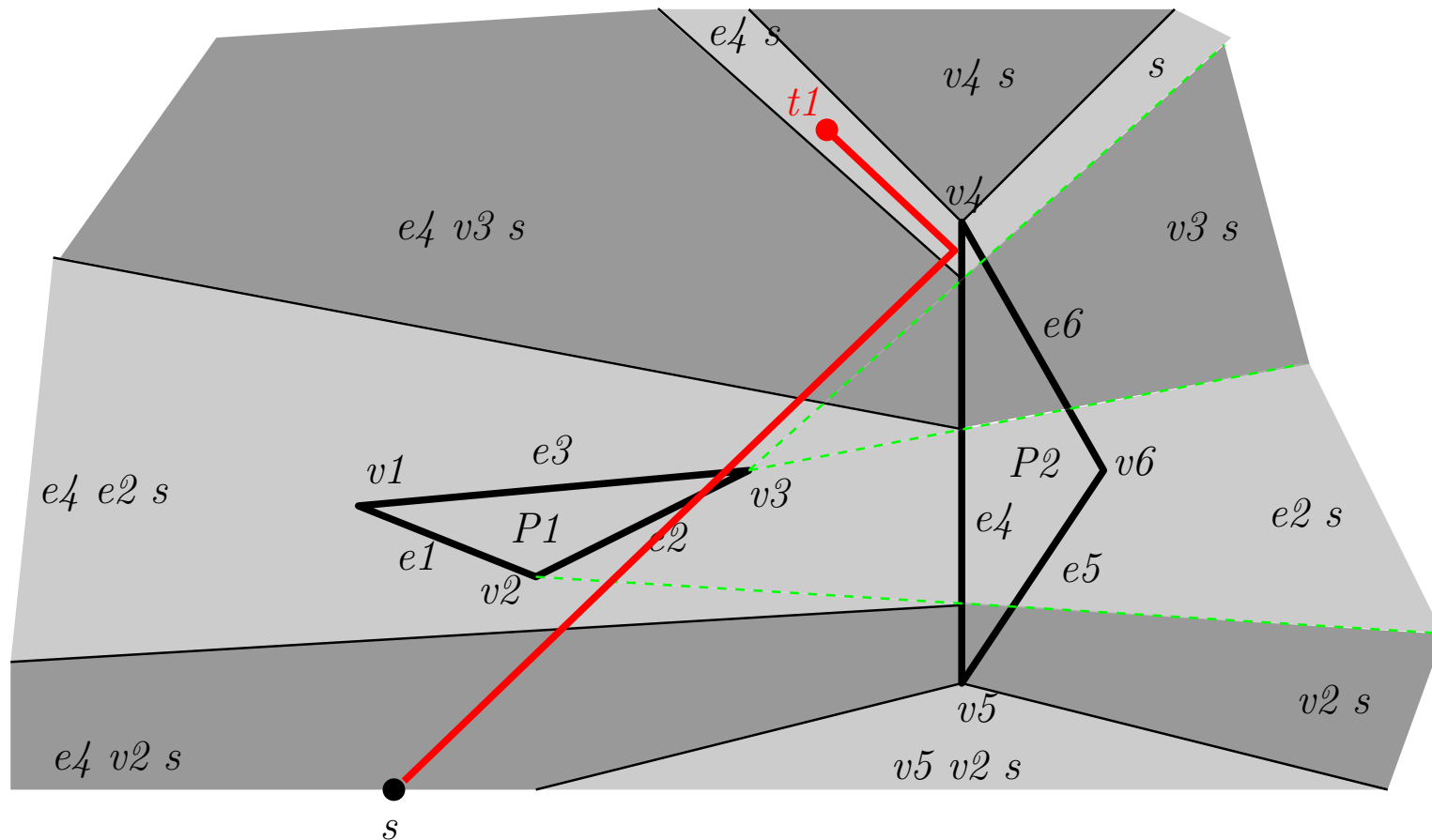
Full comb. shortest path map: Zwei Polygone



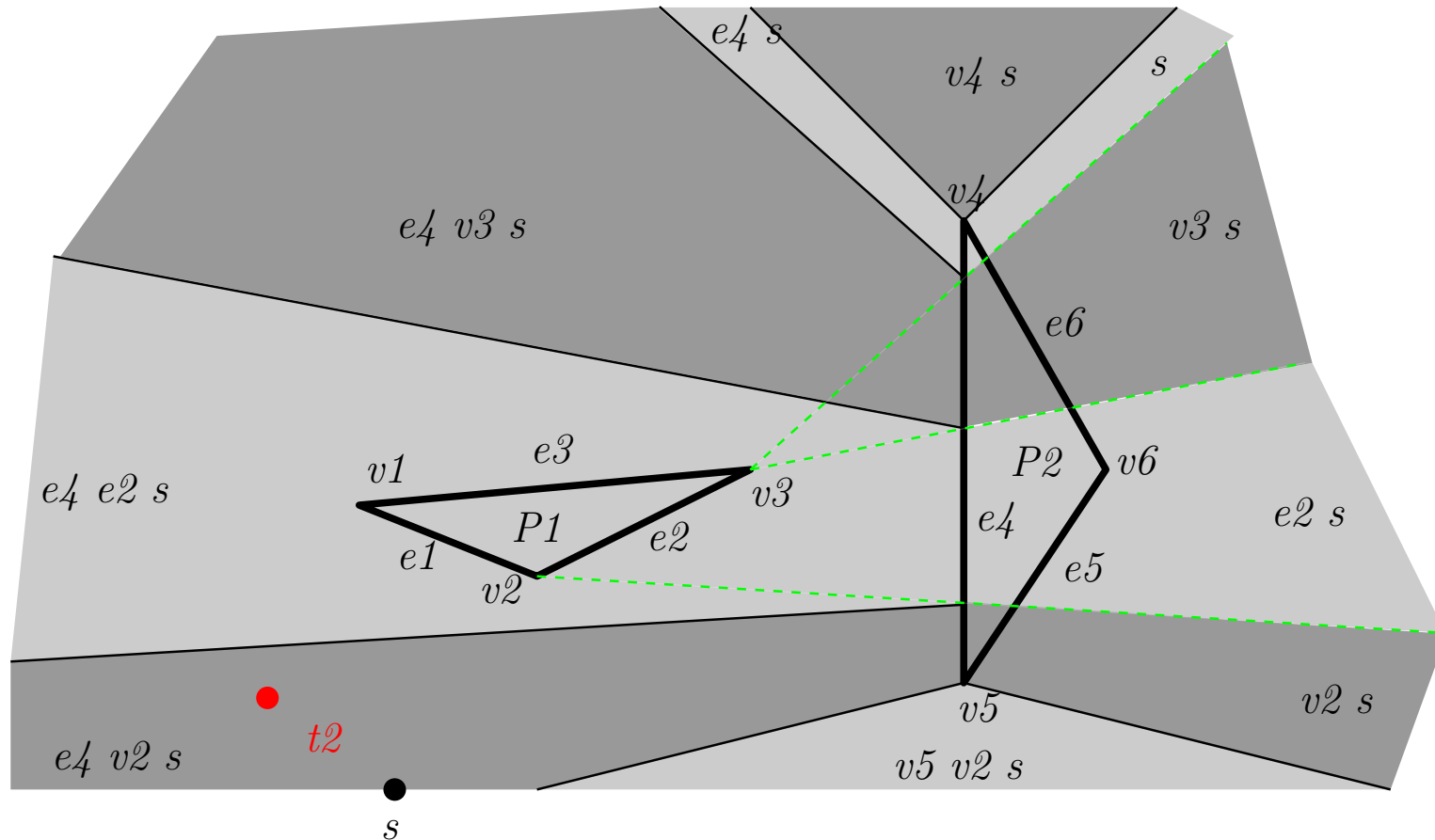
Full comb. shortest path map: Zwei Polygone



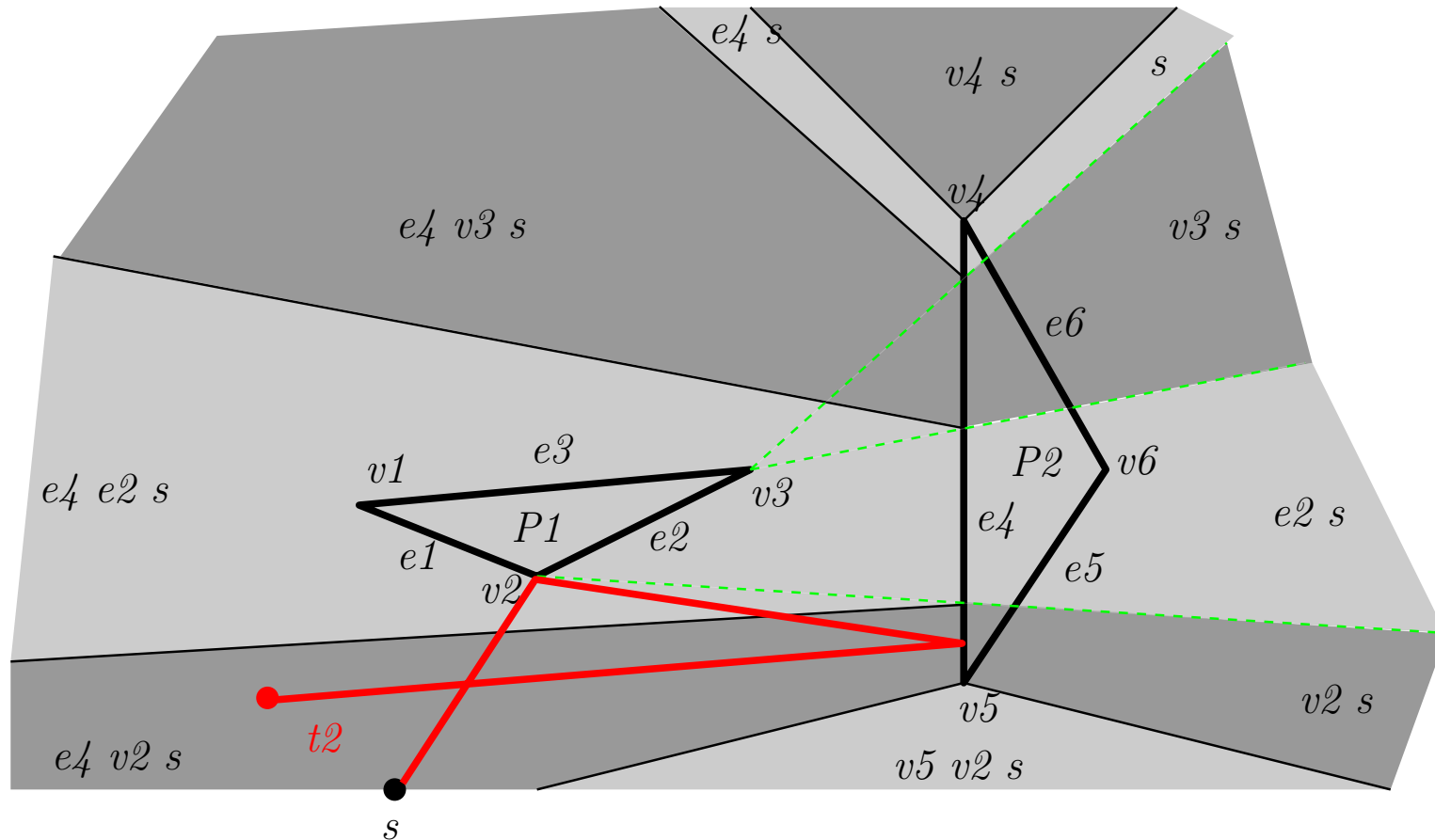
Full comb. shortest path map: Zwei Polygone



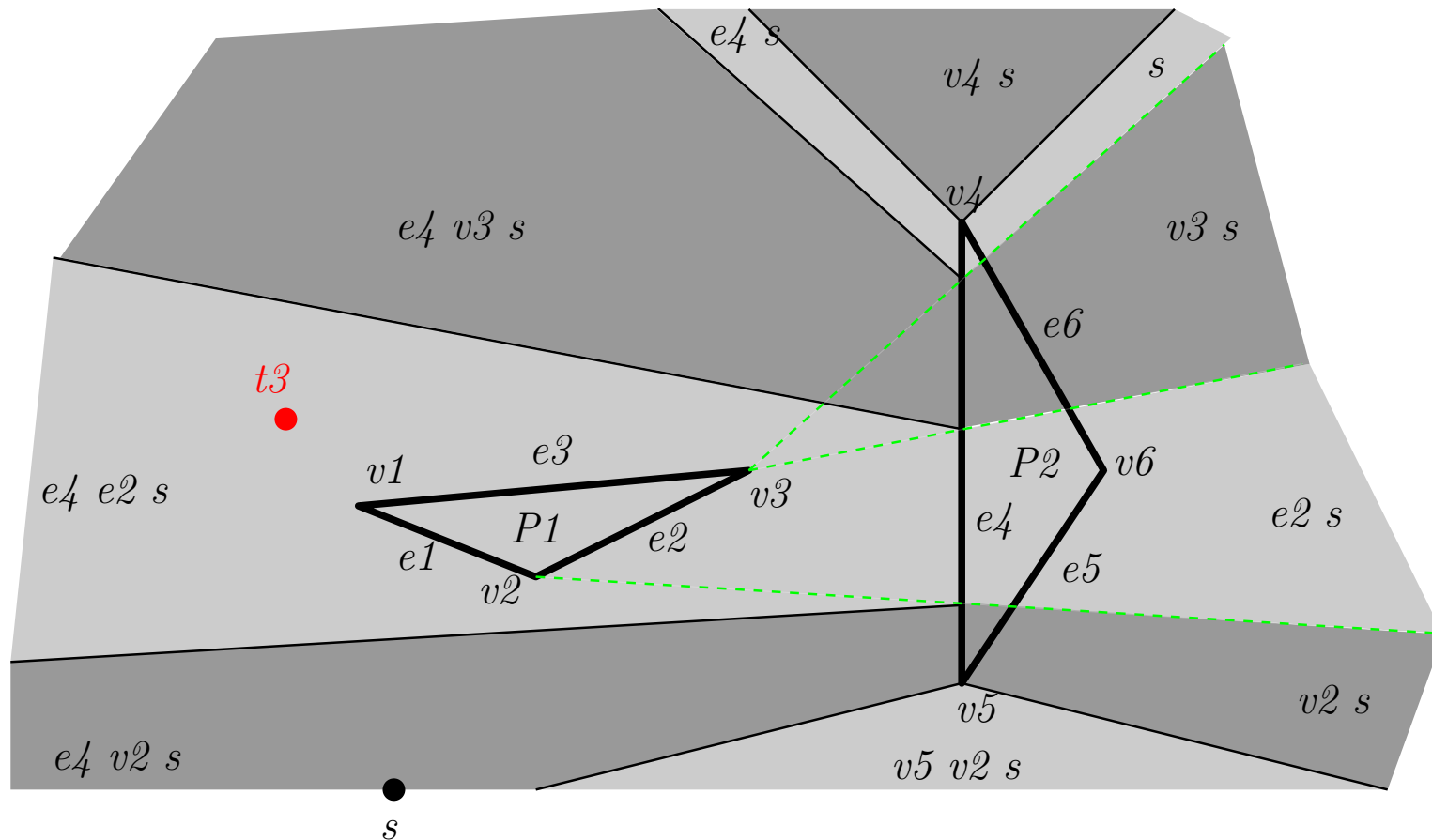
Full comb. shortest path map: Zwei Polygone



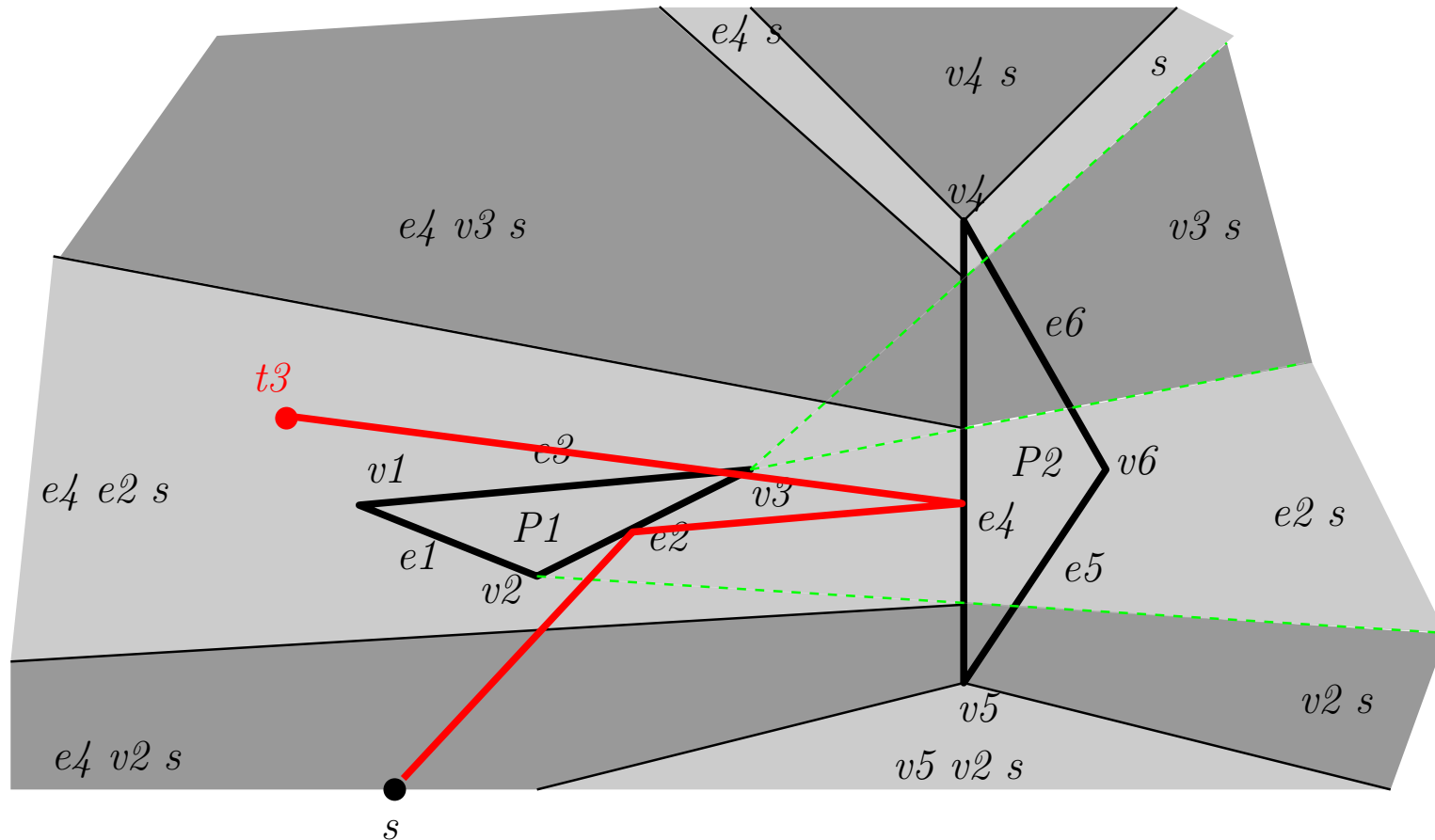
Full comb. shortest path map: Zwei Polygone



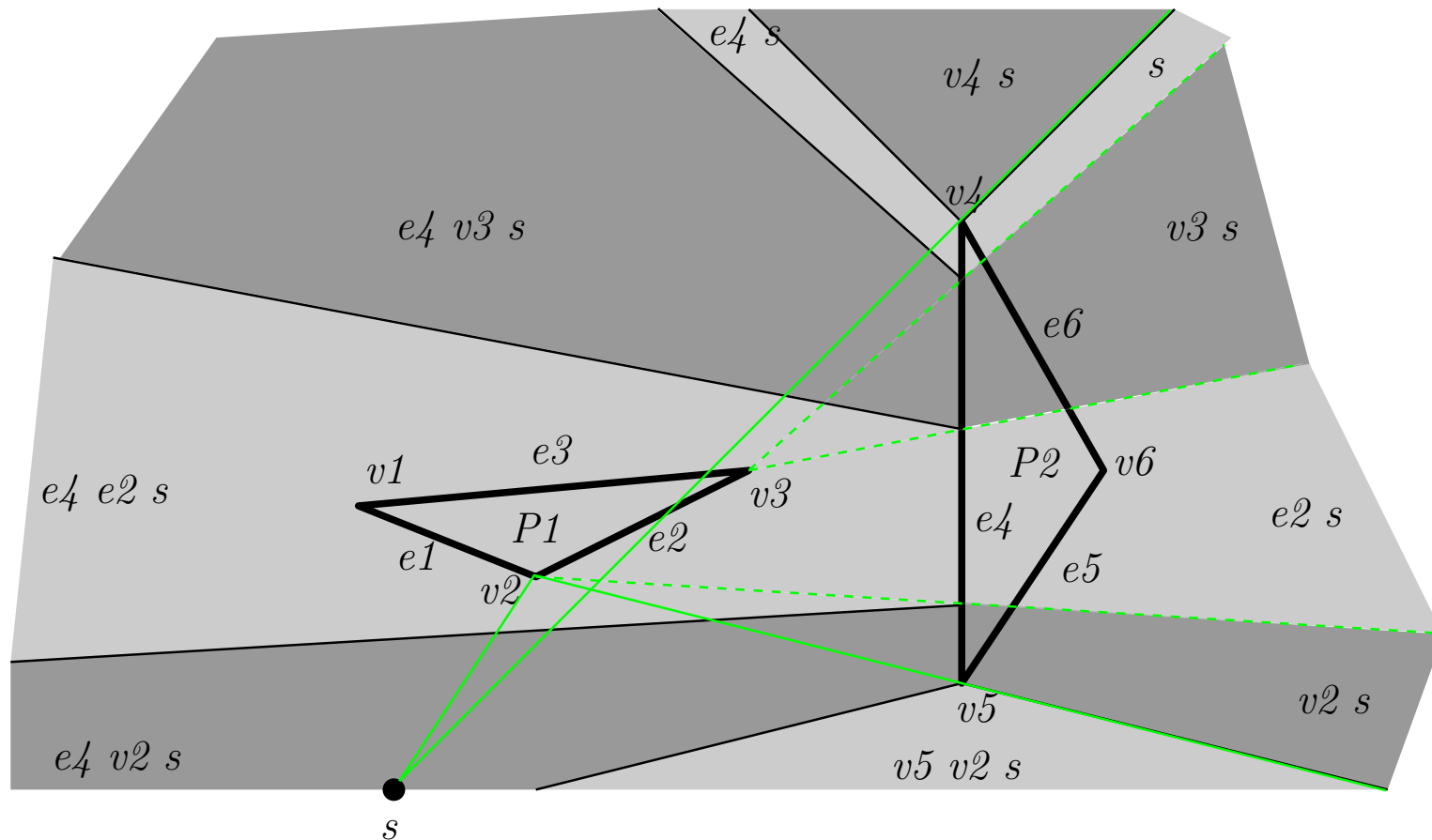
Full comb. shortest path map: Zwei Polygone



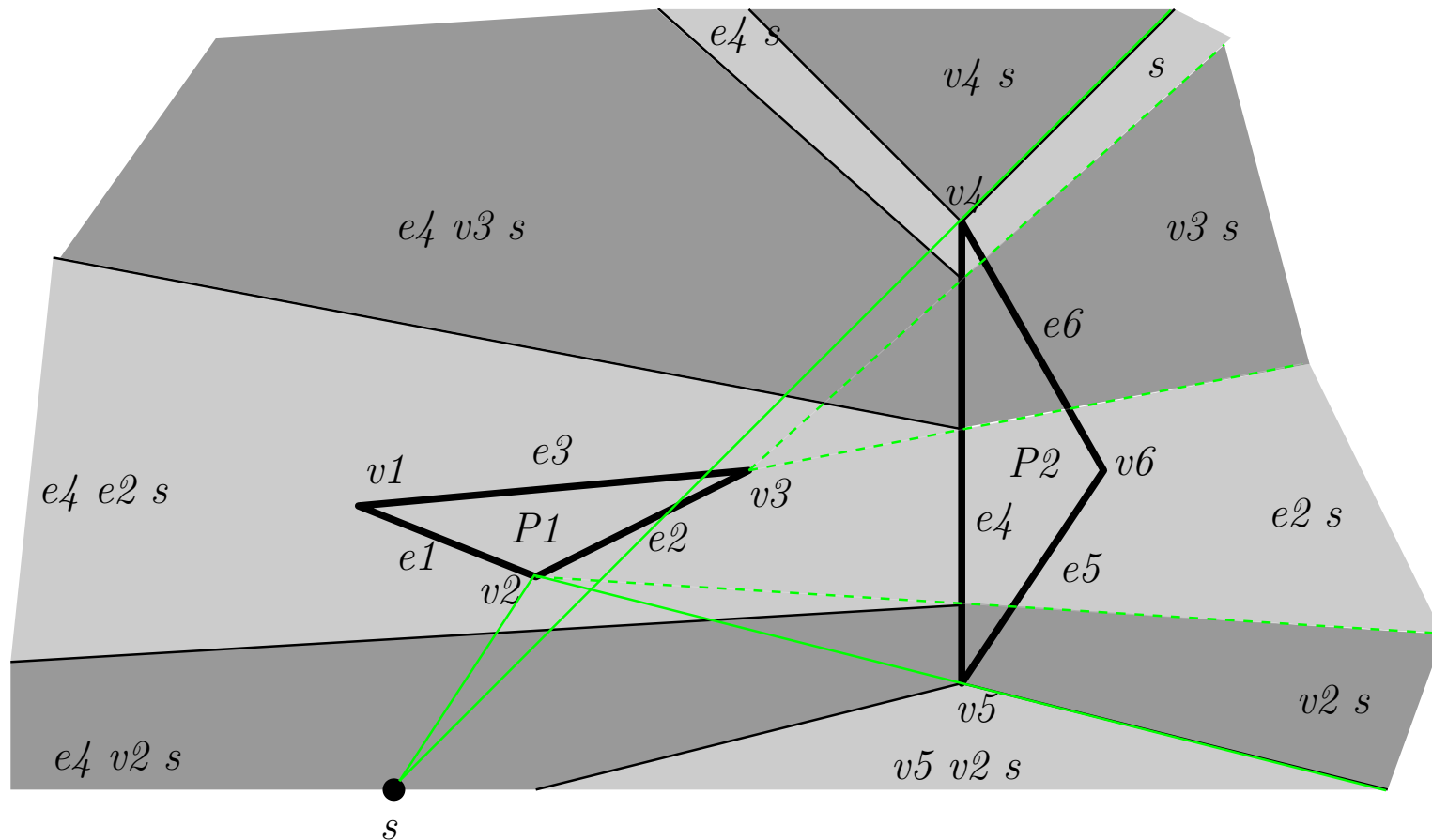
Full comb. shortest path map: Zwei Polygone



Full comb. shortest path map: Zwei Polygone



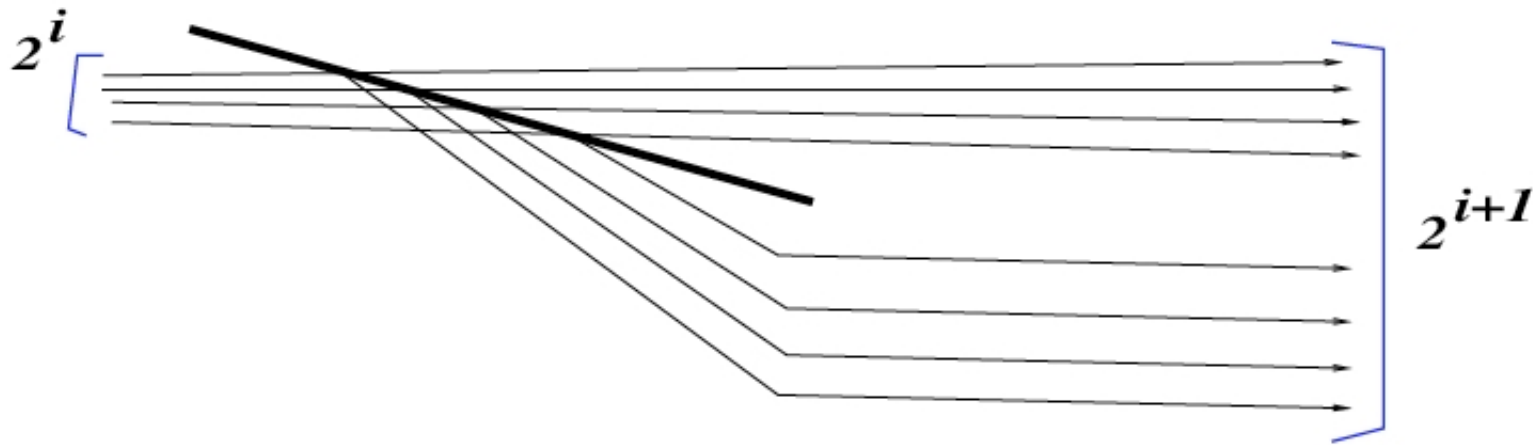
Full comb. shortest path map: Zwei Polygone



Komplexität der FC SPM

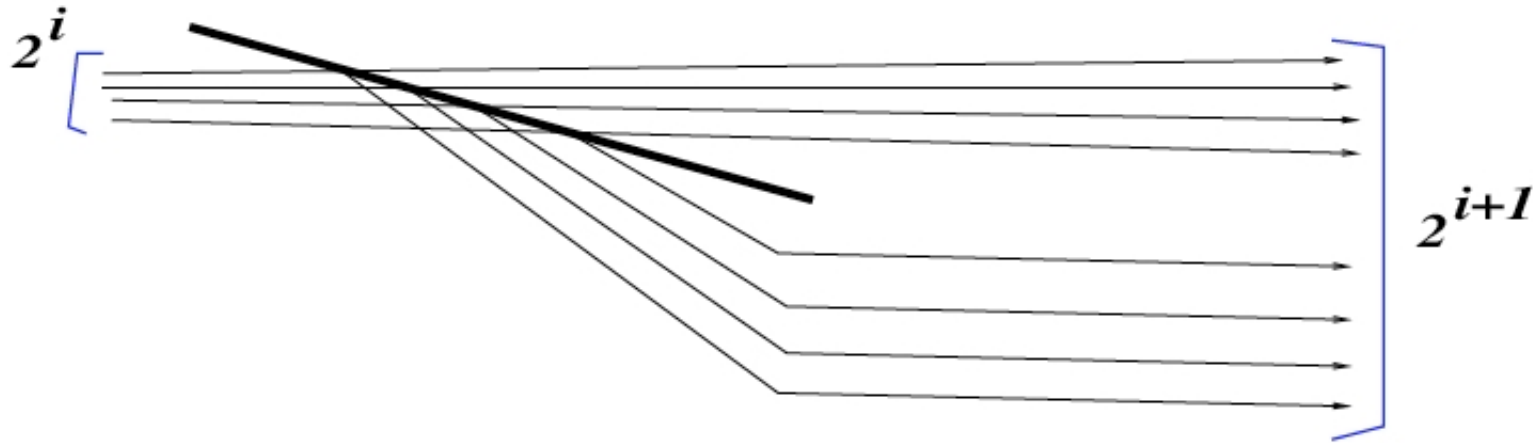
Komplexität der FC SPM

Anzahl der Kanten: Verdoppeln für jedes Polygon!



Komplexität der FC SPM

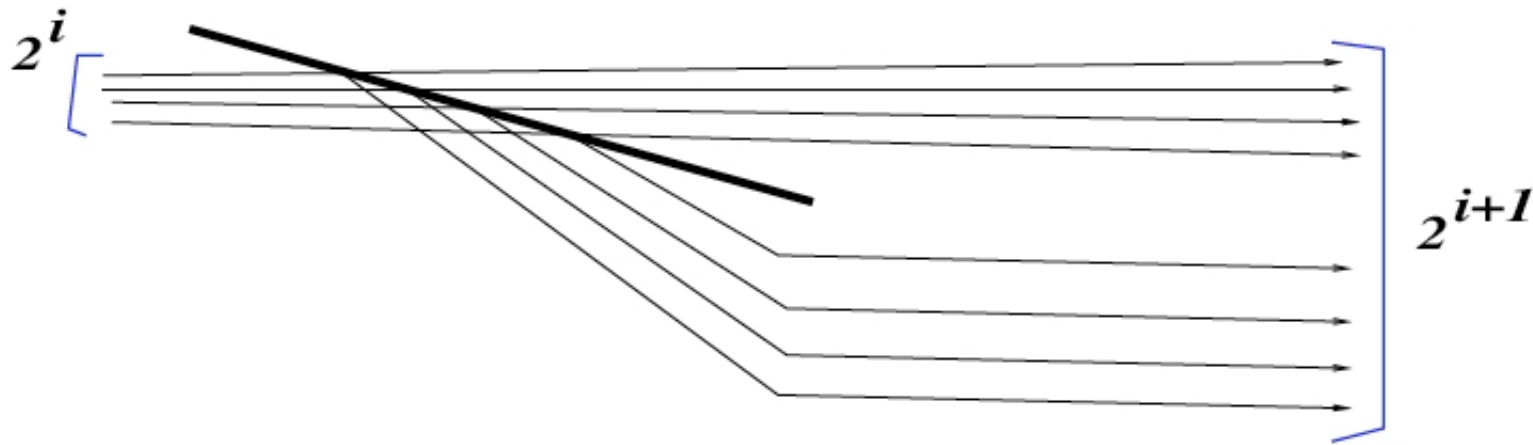
Anzahl der Kanten: Verdoppeln für jedes Polygon!



Mehr als $\Omega((n - k)2^k)$ Kanten!

Komplexität der FC SPM

Anzahl der Kanten: Verdoppeln für jedes Polygon!



Mehr als $\Omega((n - k)2^k)$ Kanten! Zu viele!

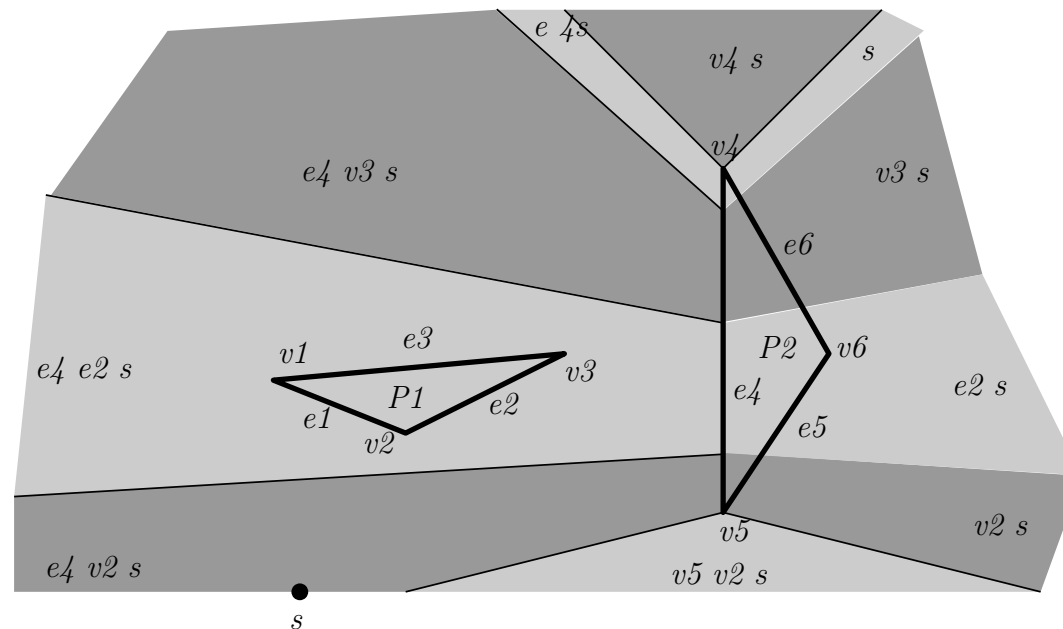
Last step shortest path map

Last step shortest path map

- Der letzte Schritt des Kürzesten Weges

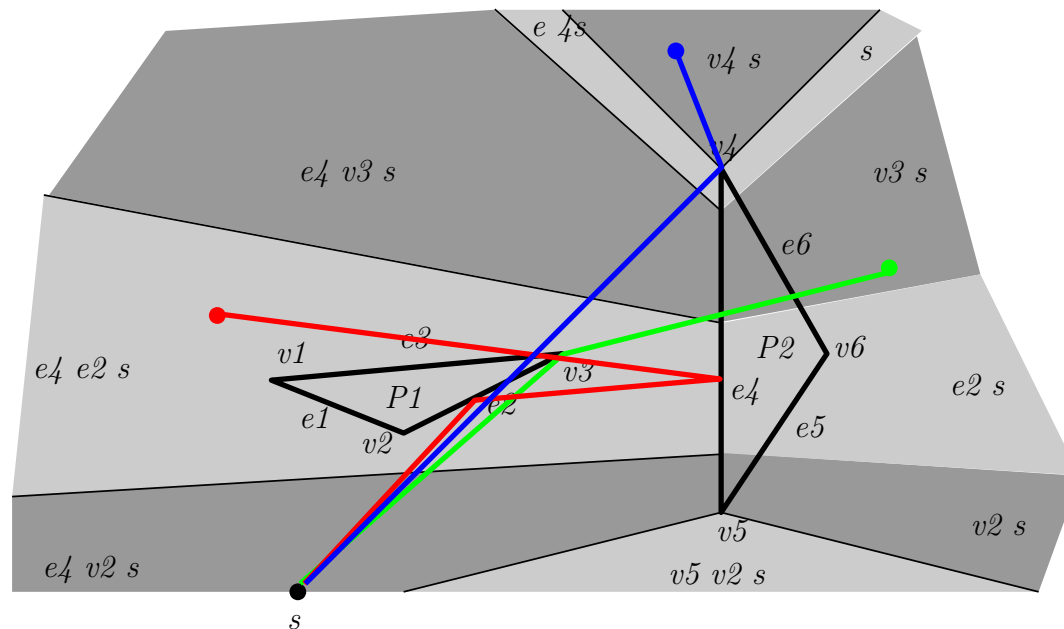
Last step shortest path map

- Der letzte Schritt des Kürzesten Weges



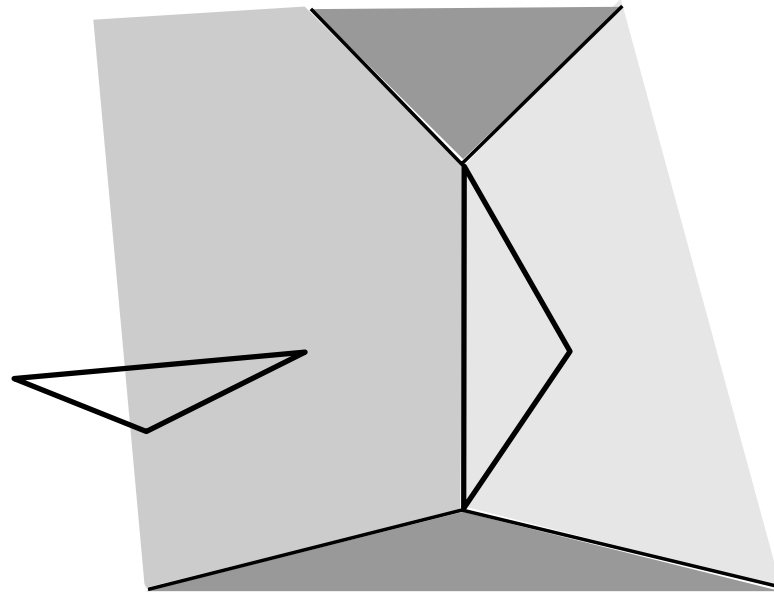
Last step shortest path map

- Der letzte Schritt des Kürzesten Weges



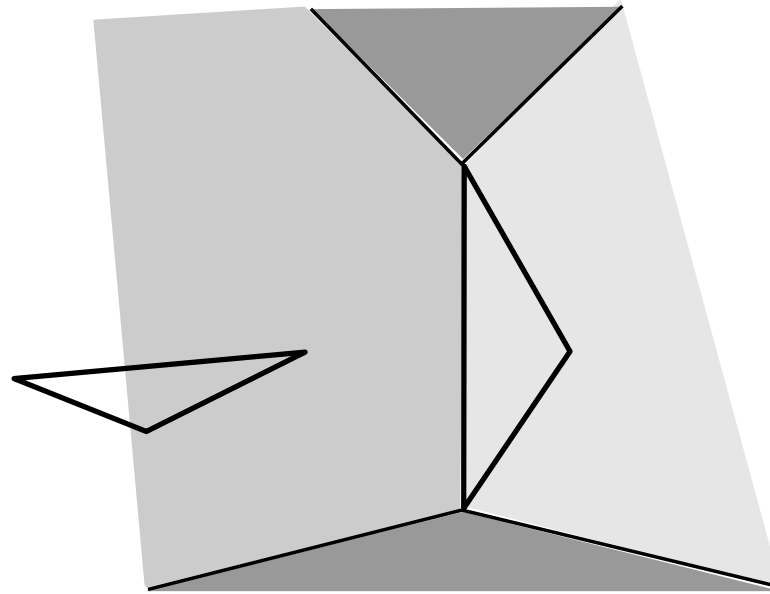
Last step shortest path map

- Der letzte Schritt des Kürzesten Weges
- Einfacher Teil der vollen SPM!

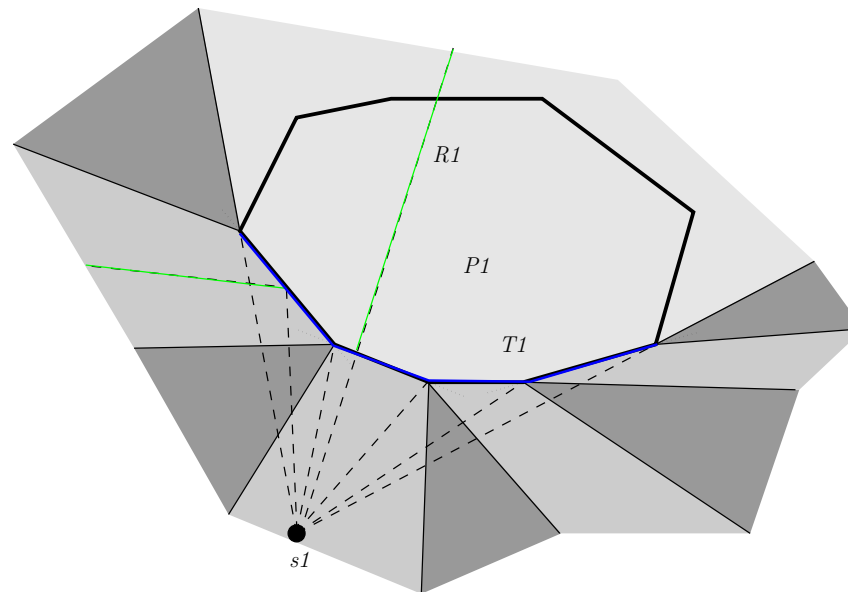


Last step shortest path map

- Der letzte Schritt des Kürzesten Weges
- Einfacher Teil der vollen SPM!
- Durchgang, Reflektion, Kante/Knoten

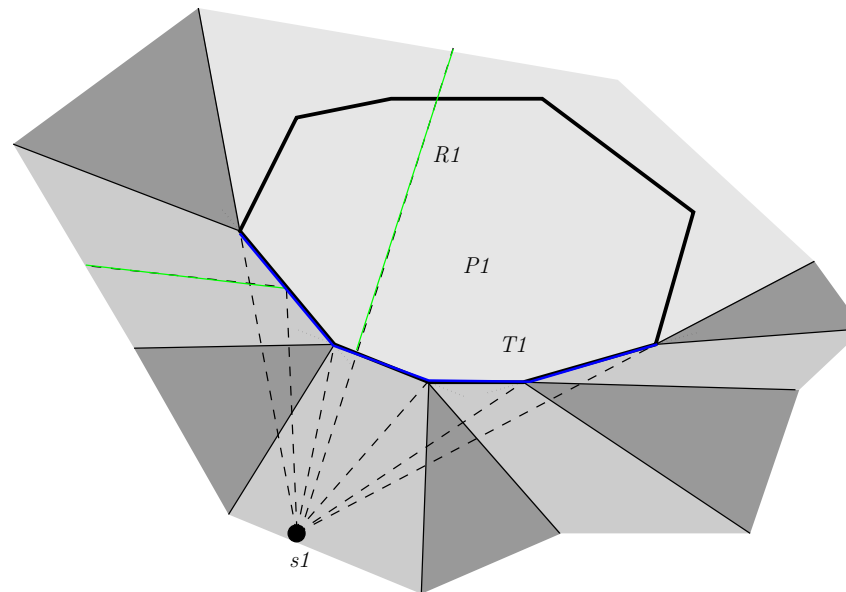


Last Step SPM S_1, S_2, \dots, S_k



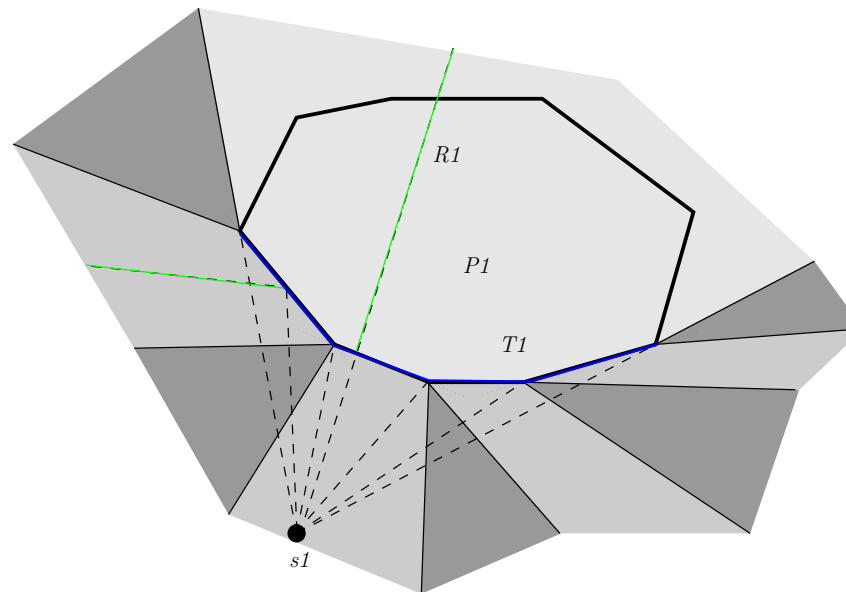
Last Step SPM S_1, S_2, \dots, S_k

- S_i gehört zur Sequenz P_1, P_2, \dots, P_i



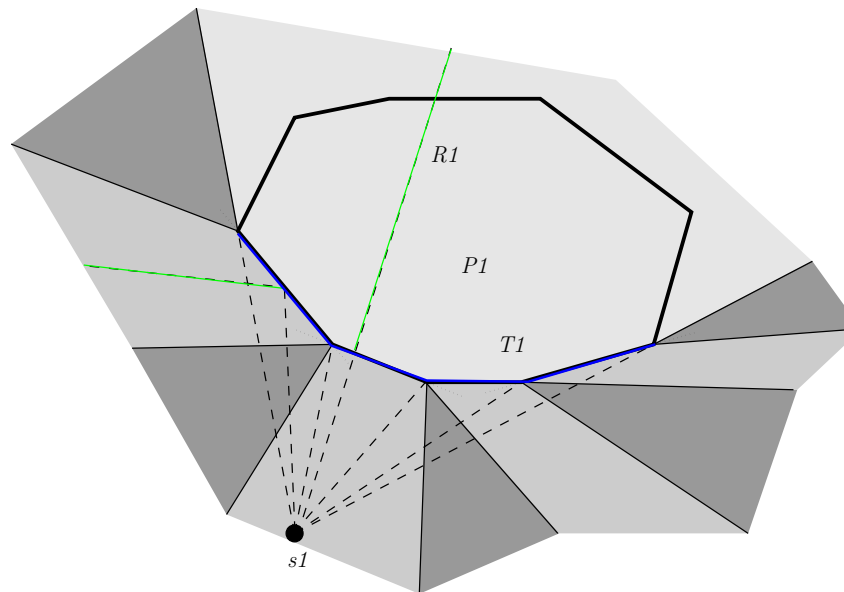
Last Step SPM S_1, S_2, \dots, S_k

- S_i gehört zur Sequenz P_1, P_2, \dots, P_i
- Beschaffenheit S_i : Reflektionsbereich T_i (konvexe Kette)



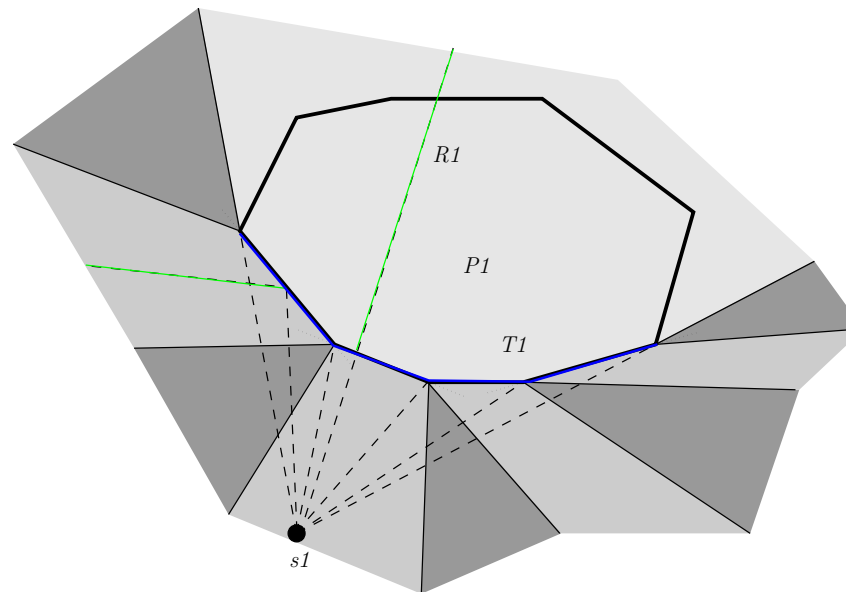
Last Step SPM S_1, S_2, \dots, S_k

- S_i gehört zur Sequenz P_1, P_2, \dots, P_i
- Beschaffenheit S_i : Reflektionsbereich T_i (konvexe Kette)
- Ausgehende Strahlen: Stern R_i , disjunkt



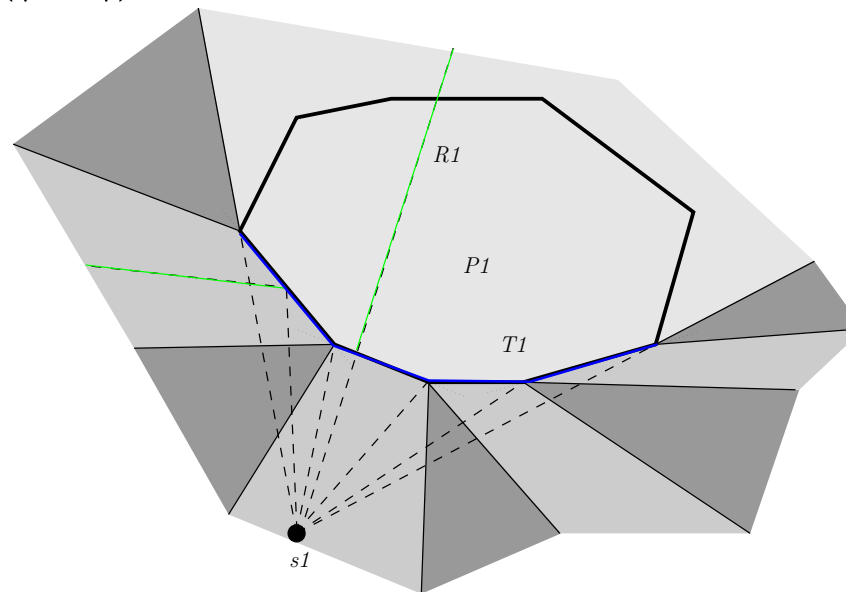
Last Step SPM S_1, S_2, \dots, S_k

- S_i gehört zur Sequenz P_1, P_2, \dots, P_i
- Beschaffenheit S_i : Reflektionsbereich T_i (konvexe Kette)
- Ausgehende Strahlen: Stern R_i , disjunkt
- Jeder Punkt wird von genau einem Strahl getroffen



Last Step SPM S_1, S_2, \dots, S_k

- S_i gehört zur Sequenz P_1, P_2, \dots, P_i
- Beschaffenheit S_i : Reflektionsbereich T_i (konvexe Kette)
- Ausgehende Strahlen: Stern R_i , disjunkt
- Jeder Punkt wird von genau einem Strahl getroffen
- Komplexität: $O(|P_i|)$



Lemma 1.32: Beweis!

Lemma 1.32: Beweis!

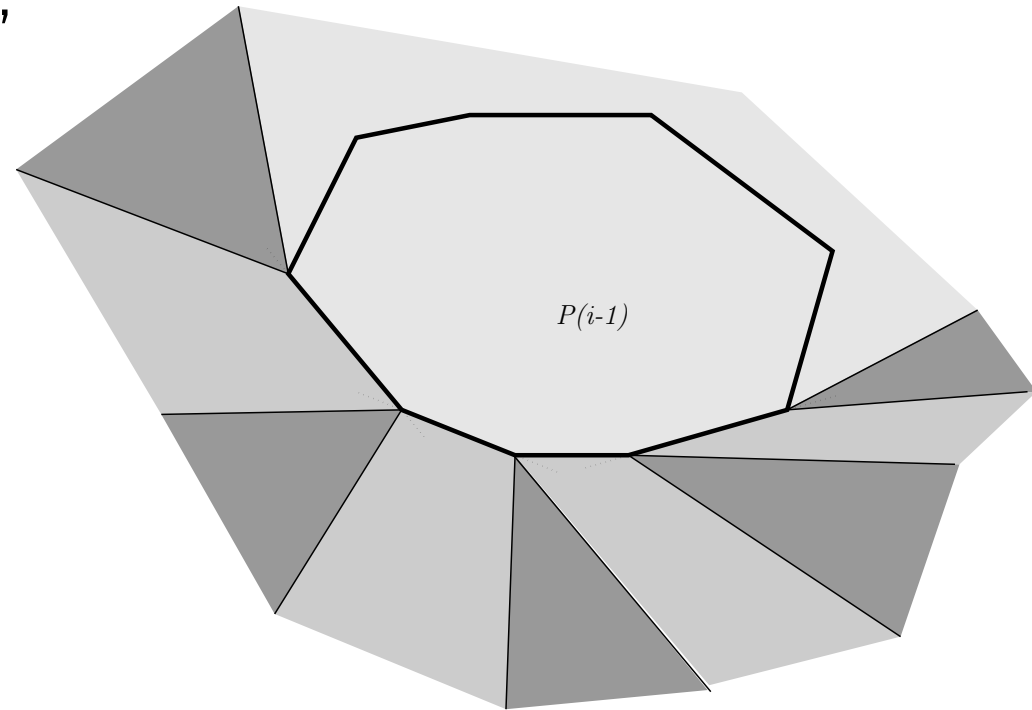
- Ind.: $O(|P_i|)$, Konvexe Kette T_i ,
Disjunkter Stern R_i

Lemma 1.32: Beweis!

- Ind.: $O(|P_i|)$, Konvexe Kette T_i ,
Disjunkter Stern R_i
- Gilt für S_1

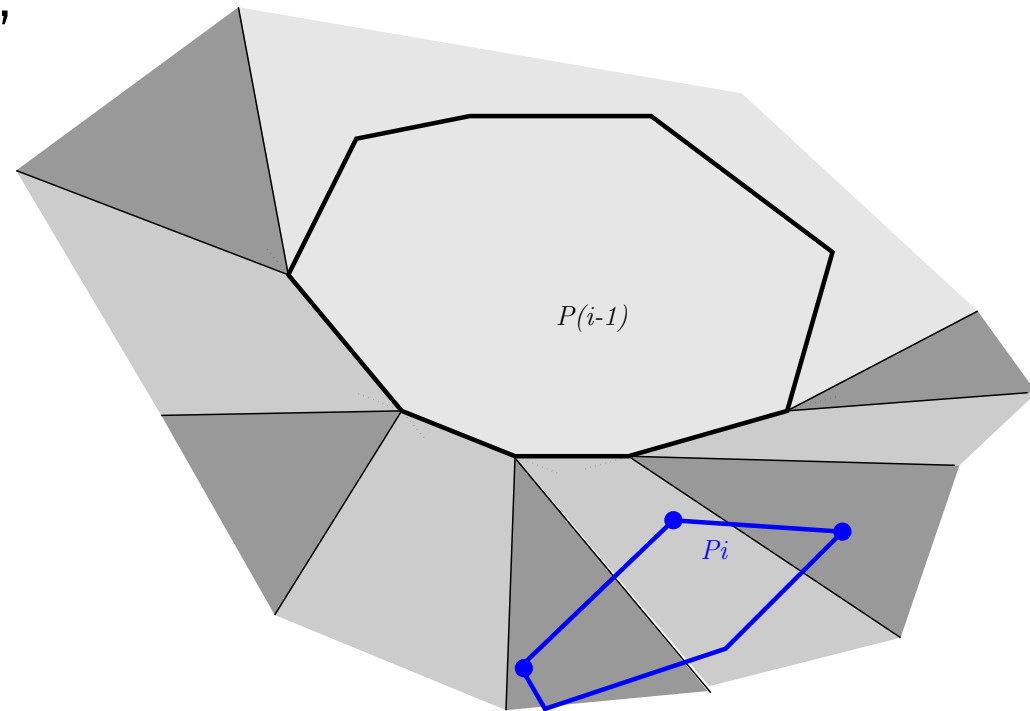
Lemma 1.32: Beweis!

- Ind.: $O(|P_i|)$, Konvexe Kette T_i ,
Disjunkter Stern R_i
- Gilt für S_1
- Annahme: Gilt für
 S_1, S_2, \dots, S_{i-1}



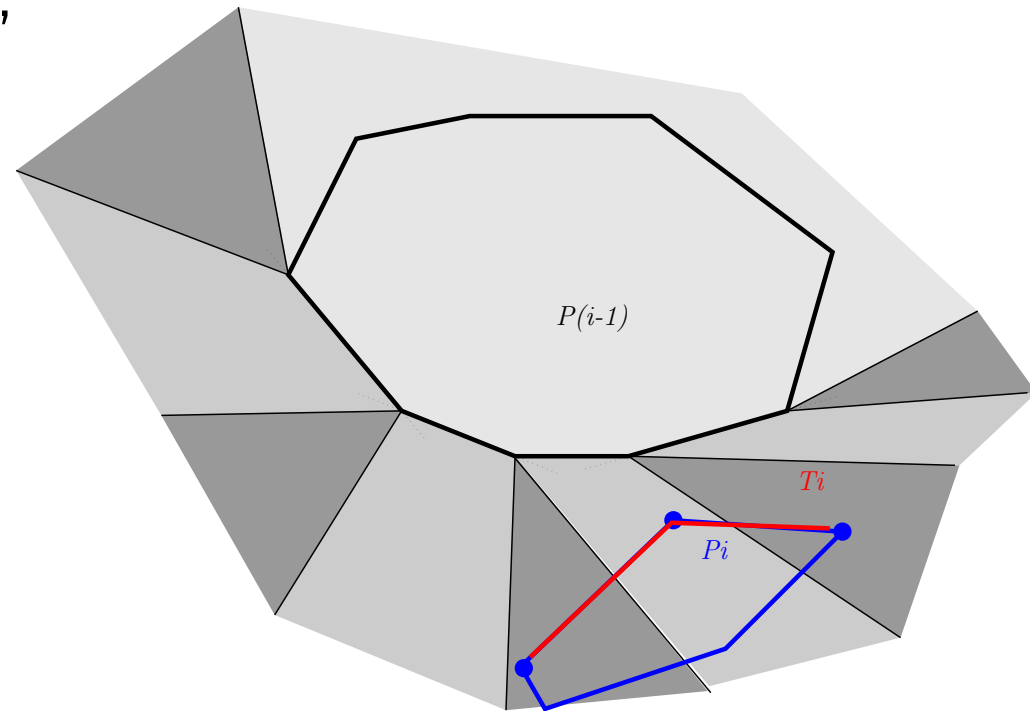
Lemma 1.32: Beweis!

- Ind.: $O(|P_i|)$, Konvexe Kette T_i ,
Disjunkter Stern R_i
- Gilt für S_1
- Annahme: Gilt für
 S_1, S_2, \dots, S_{i-1}
- Lege konvexes Polygon P_i
in S_{i-1}



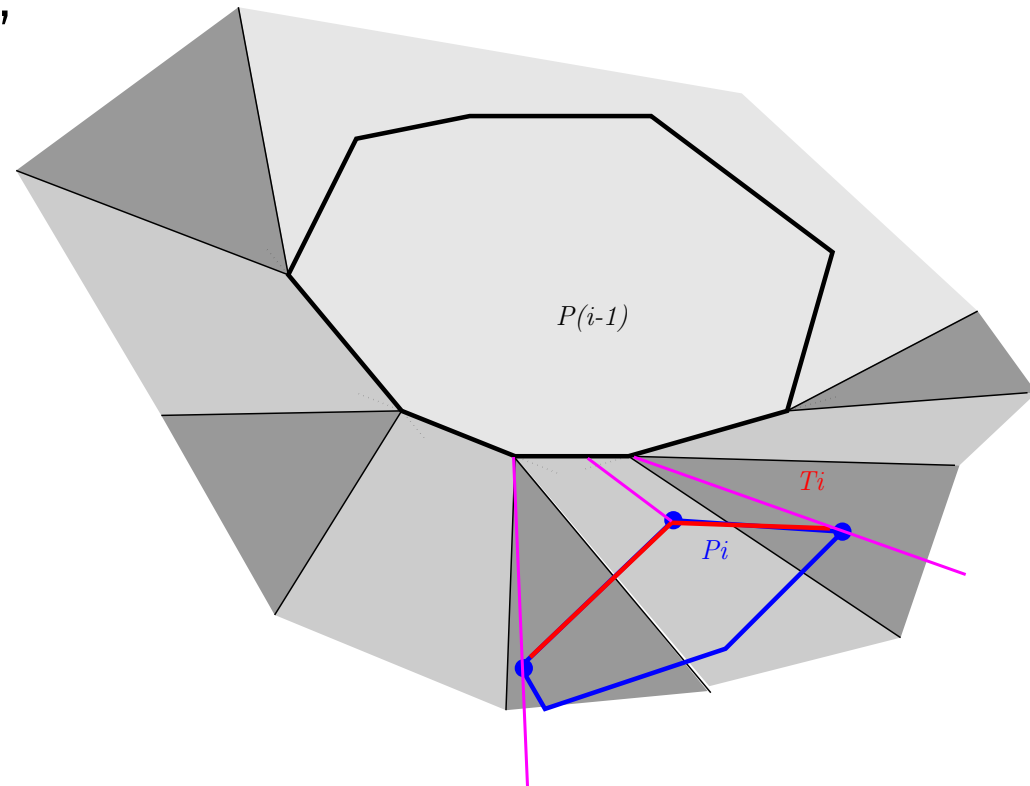
Lemma 1.32: Beweis!

- Ind.: $O(|P_i|)$, Konvexe Kette T_i ,
Disjunkter Stern R_i
- Gilt für S_1
- Annahme: Gilt für
 S_1, S_2, \dots, S_{i-1}
- Lege konvexes Polygon P_i
in S_{i-1}
- *Sichtbare* Eckpunkte
konvexer Kette von
disjunkten Strahlen getroffen



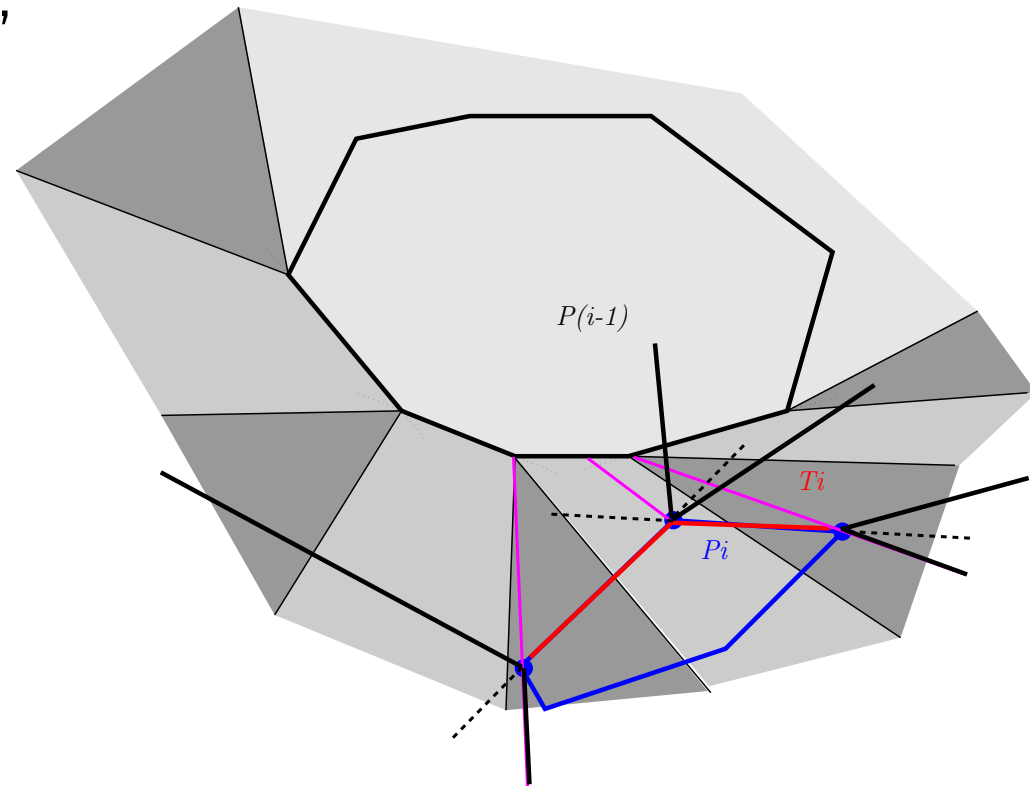
Lemma 1.32: Beweis!

- Ind.: $O(|P_i|)$, Konvexe Kette T_i ,
Disjunkter Stern R_i
- Gilt für S_1
- Annahme: Gilt für
 S_1, S_2, \dots, S_{i-1}
- Lege konvexes Polygon P_i
in S_{i-1}
- *Sichtbare* Eckpunkte
konvexer Kette von
disjunkten Strahlen getroffen



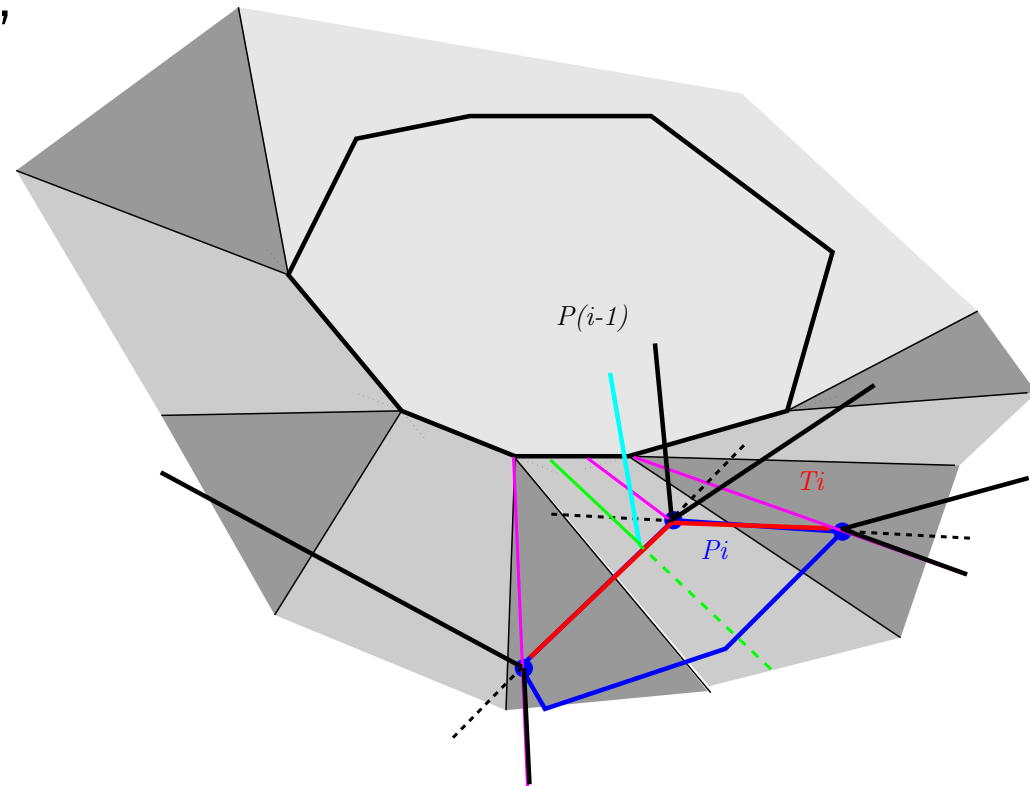
Lemma 1.32: Beweis!

- Ind.: $O(|P_i|)$, Konvexe Kette T_i ,
Disjunkter Stern R_i
- Gilt für S_1
- Annahme: Gilt für
 S_1, S_2, \dots, S_{i-1}
- Lege konvexes Polygon P_i
in S_{i-1}
- *Sichtbare* Eckpunkte
konvexer Kette von
disjunkten Strahlen getroffen
- Reflektionen disjunkt



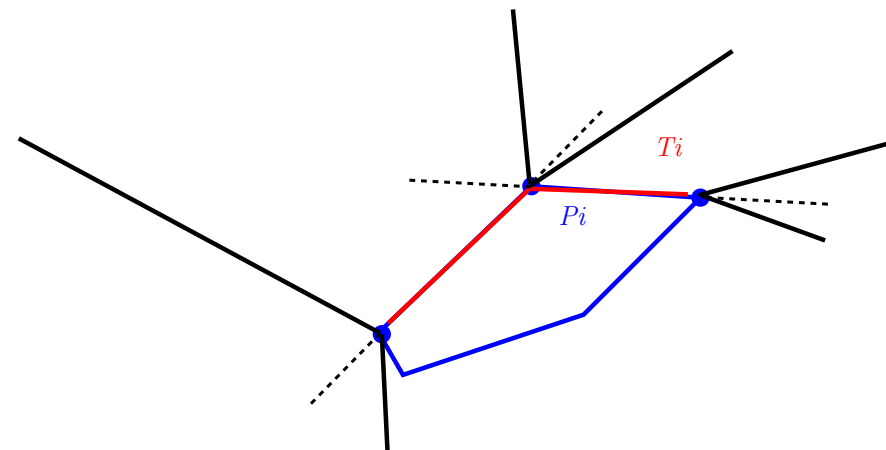
Lemma 1.32: Beweis!

- Ind.: $O(|P_i|)$, Konvexe Kette T_i ,
Disjunkter Stern R_i
- Gilt für S_1
- Annahme: Gilt für
 S_1, S_2, \dots, S_{i-1}
- Lege konvexes Polygon P_i
in S_{i-1}
- *Sichtbare* Eckpunkte
konvexer Kette von
disjunkten Strahlen getroffen
- Reflektionen disjunkt



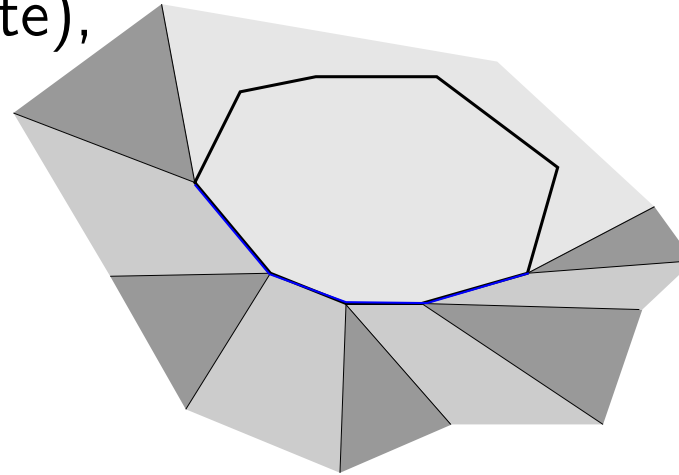
Lemma 1.32: Beweis!

- Ind.: $O(|P_i|)$, Konvexe Kette T_i ,
Disjunkter Stern R_i
- Gilt für S_1
- Annahme: Gilt für
 S_1, S_2, \dots, S_{i-1}
- Lege konvexes Polygon P_i
in S_{i-1}
- *Sichtbare* Eckpunkte
konvexer Kette von
disjunkten Strahlen getroffen
- Reflektionen disjunkt



DS und Komplexität für S_i !

- Vorbereitet für Lokalisation
- Reflektionsbereich(Knoten/Kante),
Durchgangsbereich
- Konvexe Kette,
 n_i disjunkte Strahlen
- Balancierter Baum:
Lokalization in $O(\log n_i)$
- Für alle S_i gleich



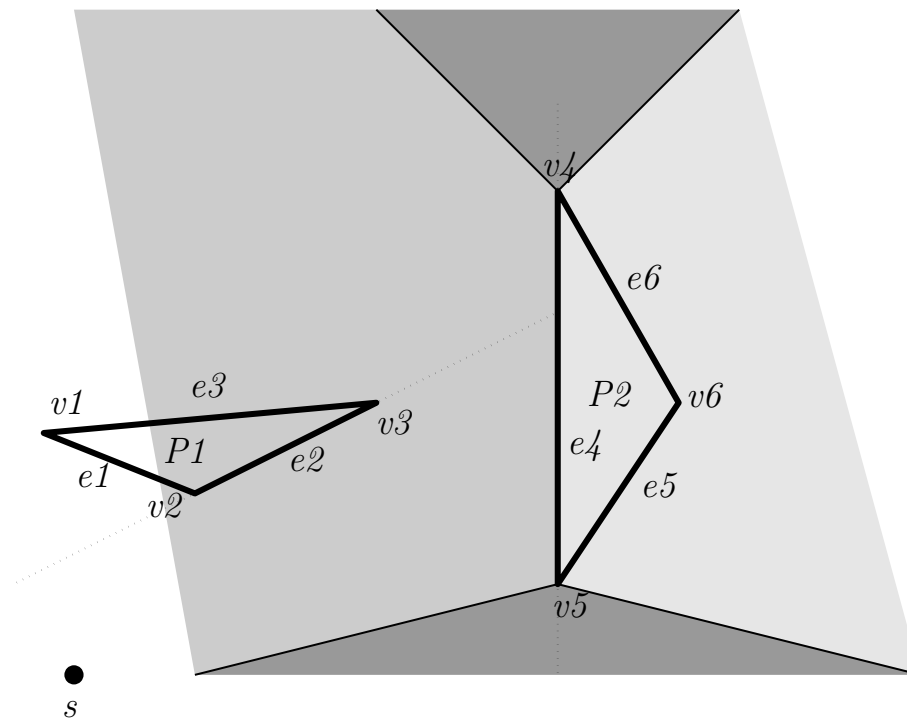
Benutzung der Last Step SPM: Alg. 1.10

Benutzung der Last Step SPM: Alg. 1.10

- Ziel t_1 : Nutze Last Step SPM von P_k

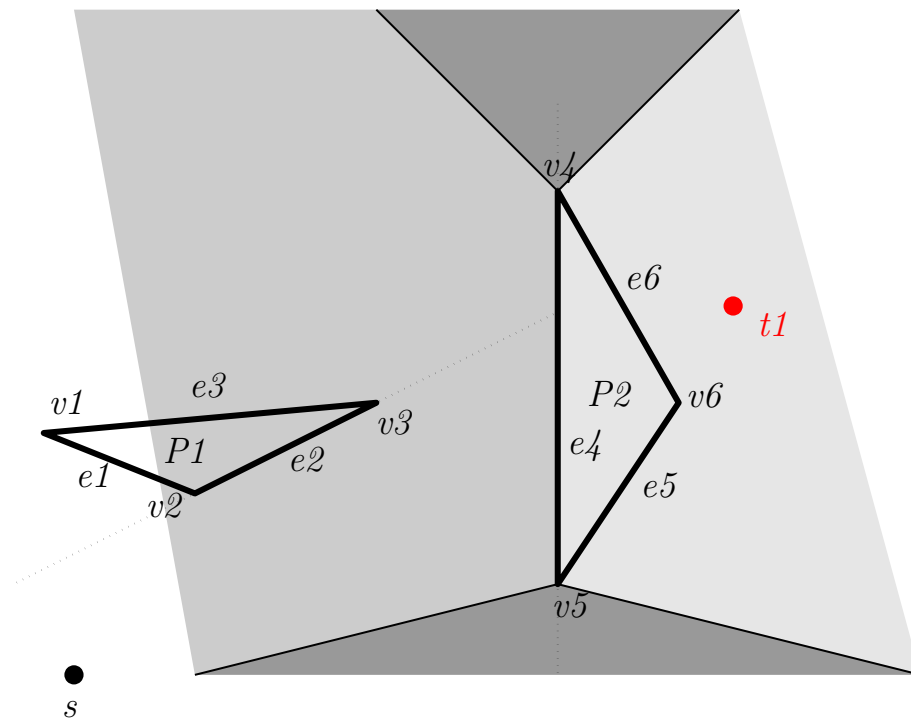
Benutzung der Last Step SPM: Alg. 1.10

- Ziel t_1 : Nutze Last Step SPM von P_k
- Induktiv nutze Last Step SPM von P_{k-1}



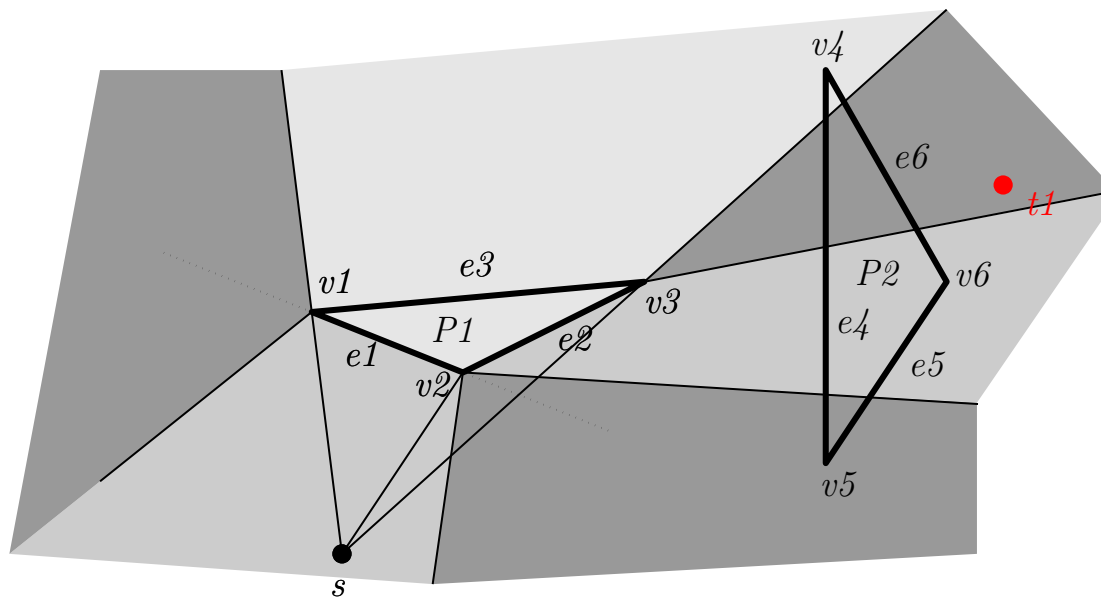
Benutzung der Last Step SPM: Alg. 1.10

- Ziel t_1 : Nutze Last Step SPM von P_k
- Induktiv nutze Last Step SPM von P_{k-1}



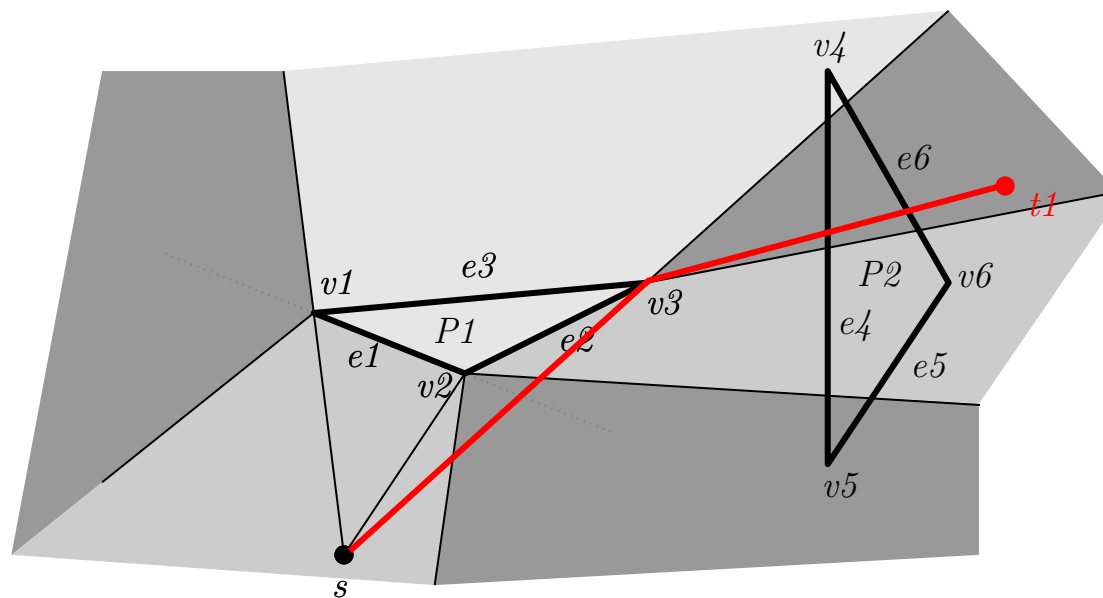
Benutzung der Last Step SPM: Alg. 1.10

- Ziel t_1 : Nutze Last Step SPM von P_k
- Induktiv nutze Last Step SPM von P_{k-1}



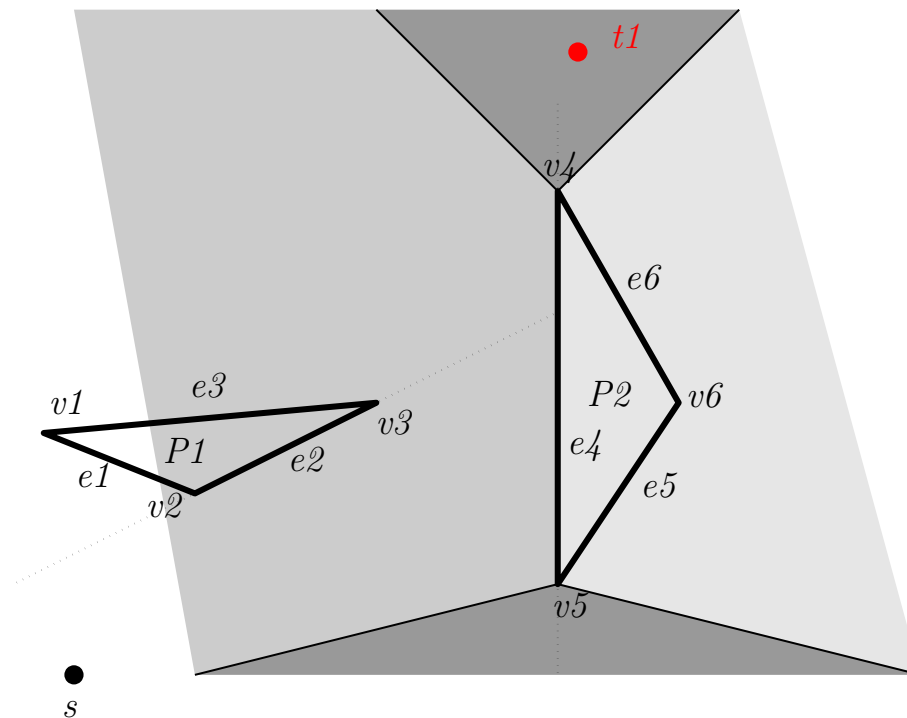
Benutzung der Last Step SPM: Alg. 1.10

- Ziel t_1 : Nutze Last Step SPM von P_k
- Induktiv nutze Last Step SPM von P_{k-1}



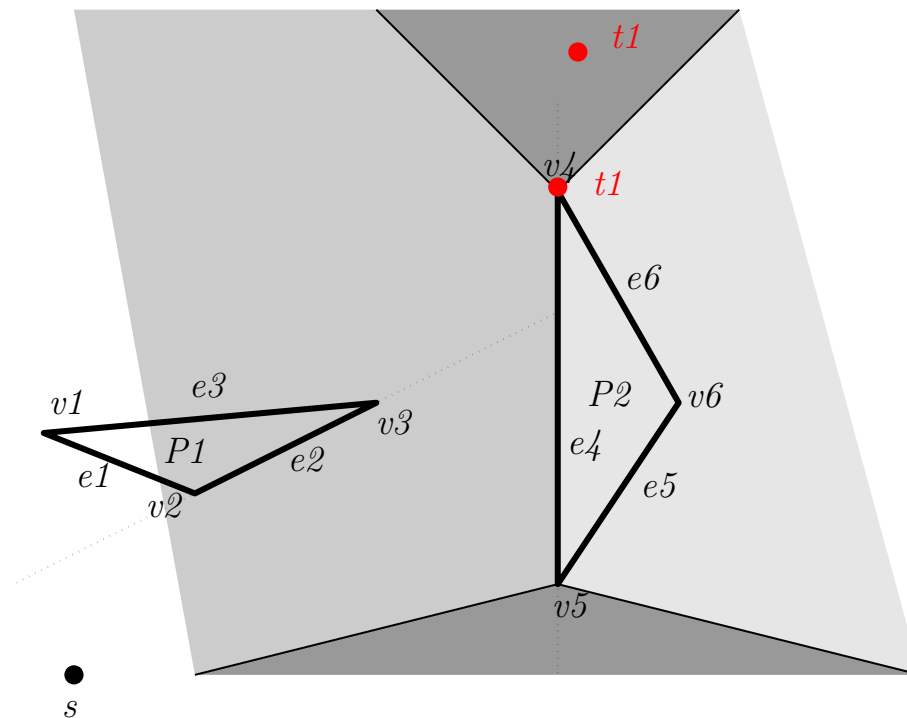
Benutzung der Last Step SPM: Alg. 1.10

- Ziel t_1 : Nutze Last Step SPM von P_k
- Induktiv nutze Last Step SPM von P_{k-1}



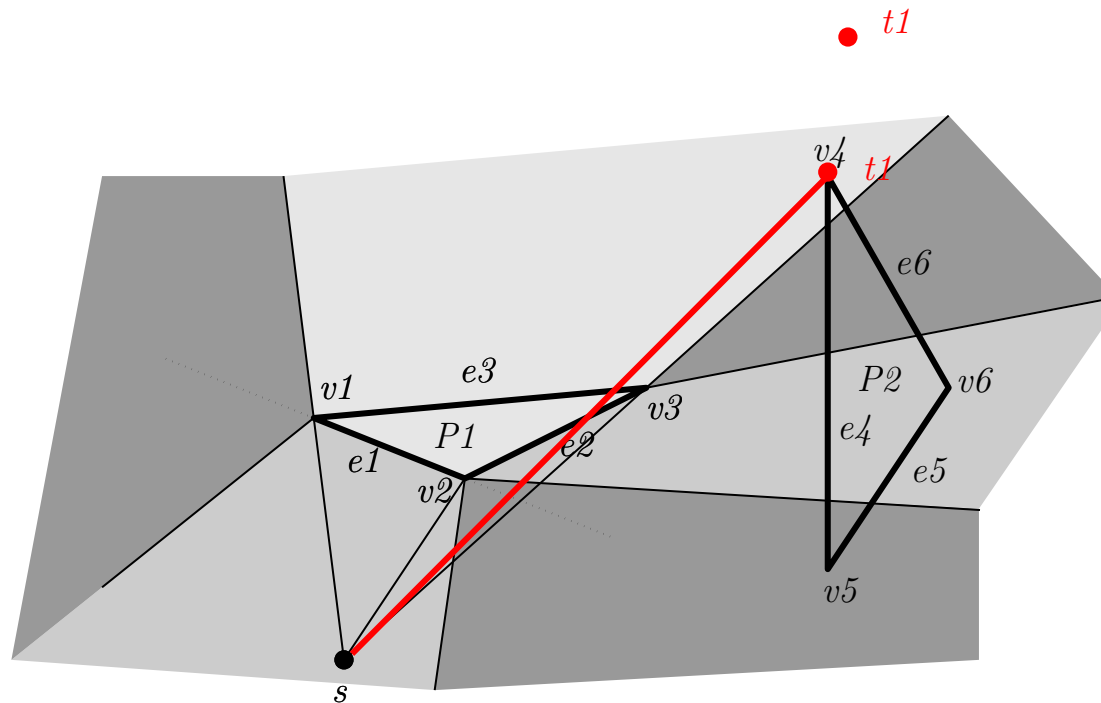
Benutzung der Last Step SPM: Alg. 1.10

- Ziel t_1 : Nutze Last Step SPM von P_k
- Induktiv nutze Last Step SPM von P_{k-1}



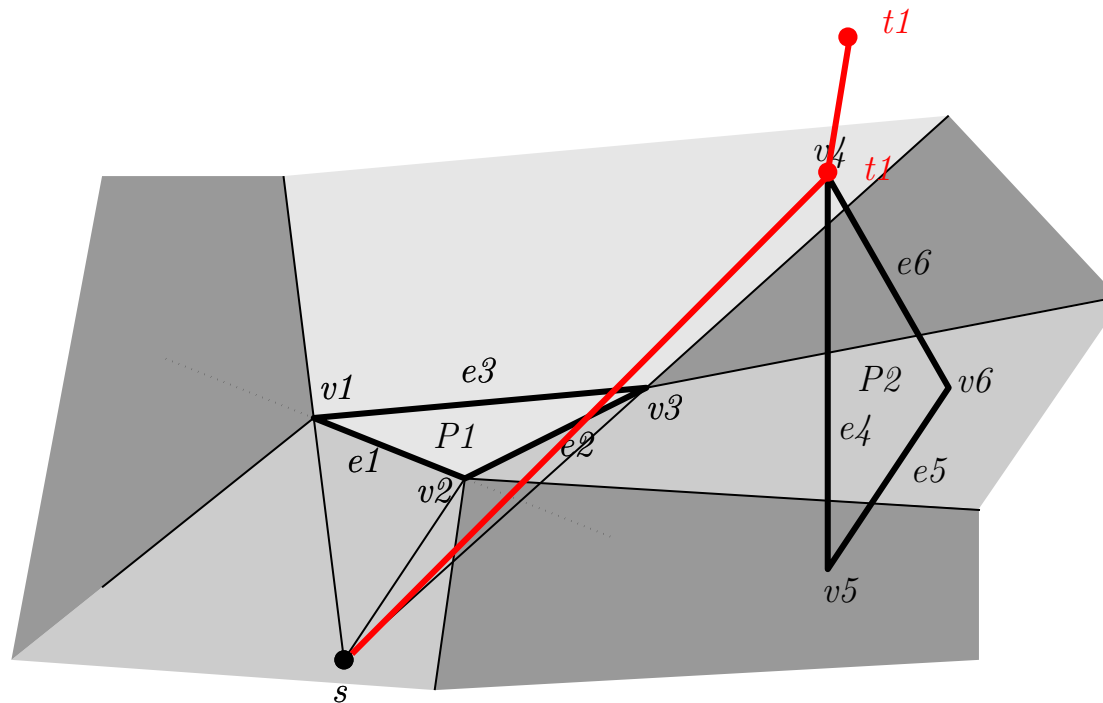
Benutzung der Last Step SPM: Alg. 1.10

- Ziel t_1 : Nutze Last Step SPM von P_k
- Induktiv nutze Last Step SPM von P_{k-1}



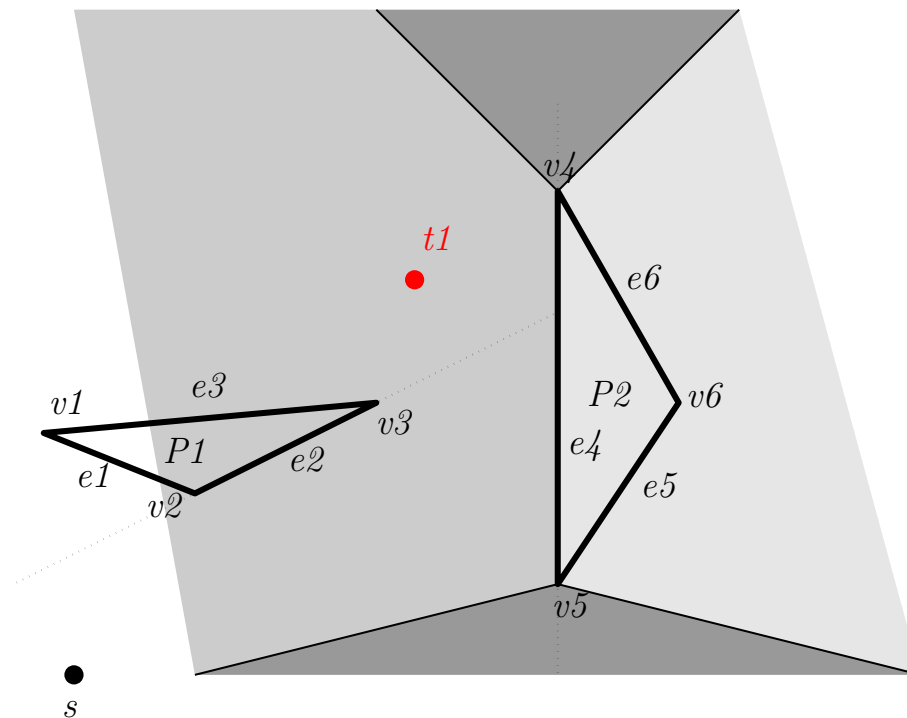
Benutzung der Last Step SPM: Alg. 1.10

- Ziel t_1 : Nutze Last Step SPM von P_k
- Induktiv nutze Last Step SPM von P_{k-1}



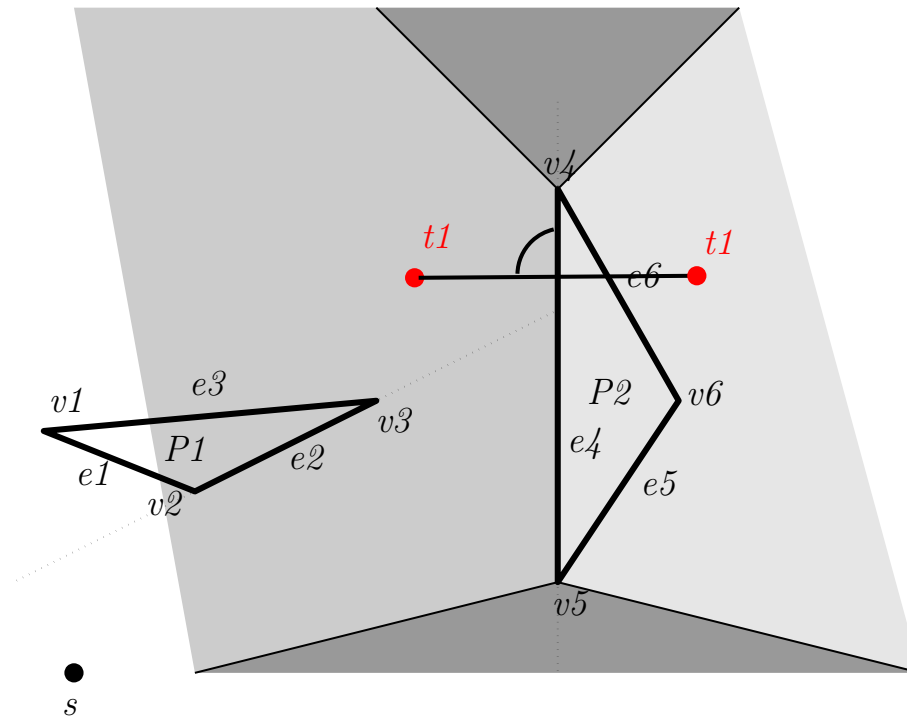
Benutzung der Last Step SPM: Alg. 1.10

- Ziel t_1 : Nutze Last Step SPM von P_k
- Induktiv nutze Last Step SPM von P_{k-1}



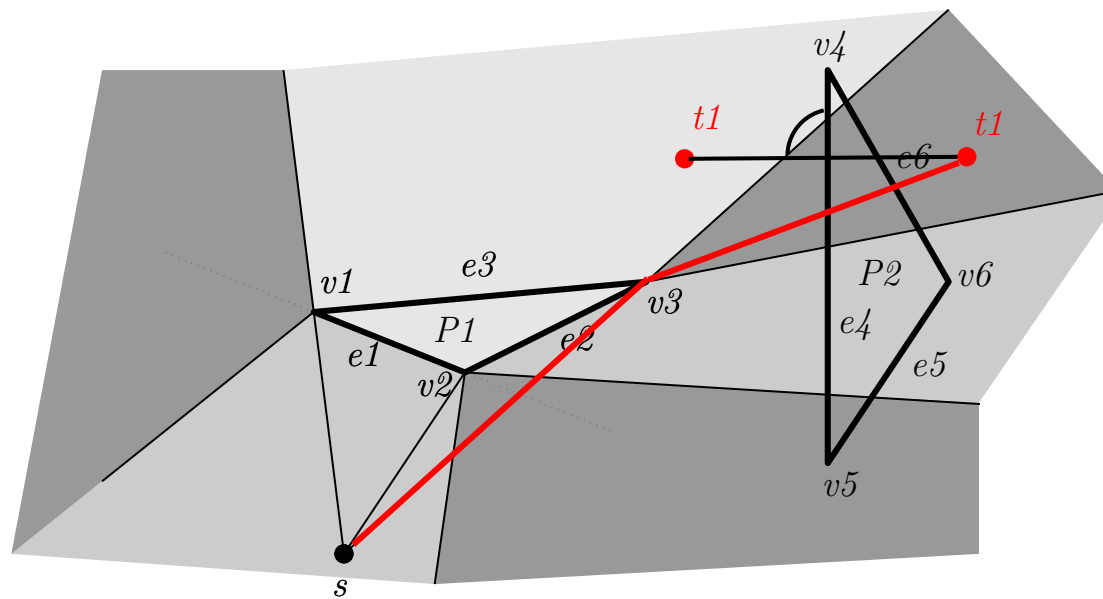
Benutzung der Last Step SPM: Alg. 1.10

- Ziel t_1 : Nutze Last Step SPM von P_k
- Induktiv nutze Last Step SPM von P_{k-1}



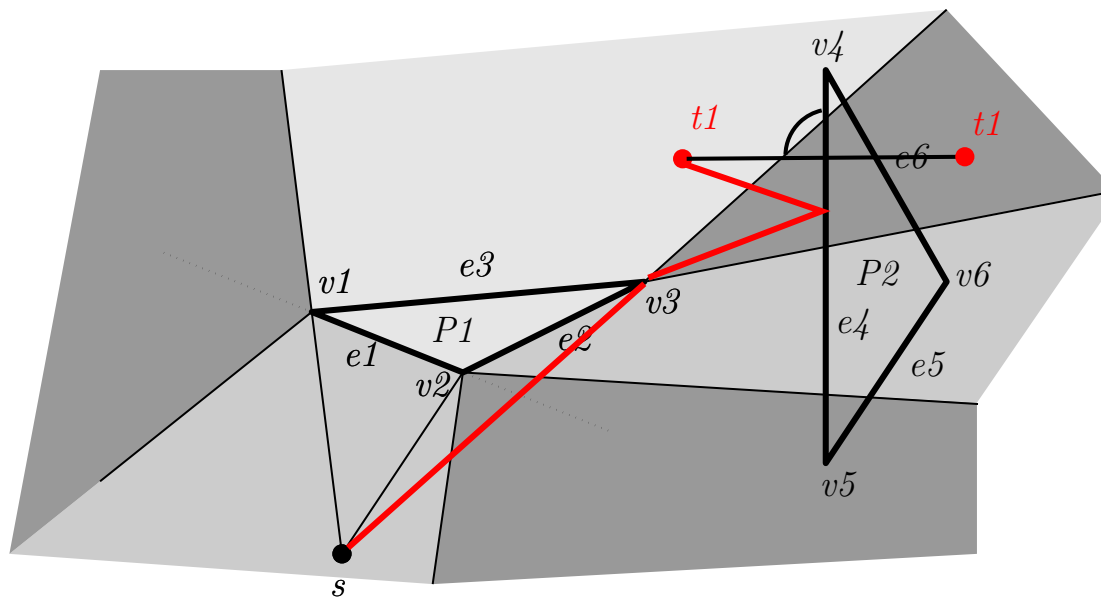
Benutzung der Last Step SPM: Alg. 1.10

- Ziel t_1 : Nutze Last Step SPM von P_k
- Induktiv nutze Last Step SPM von P_{k-1}



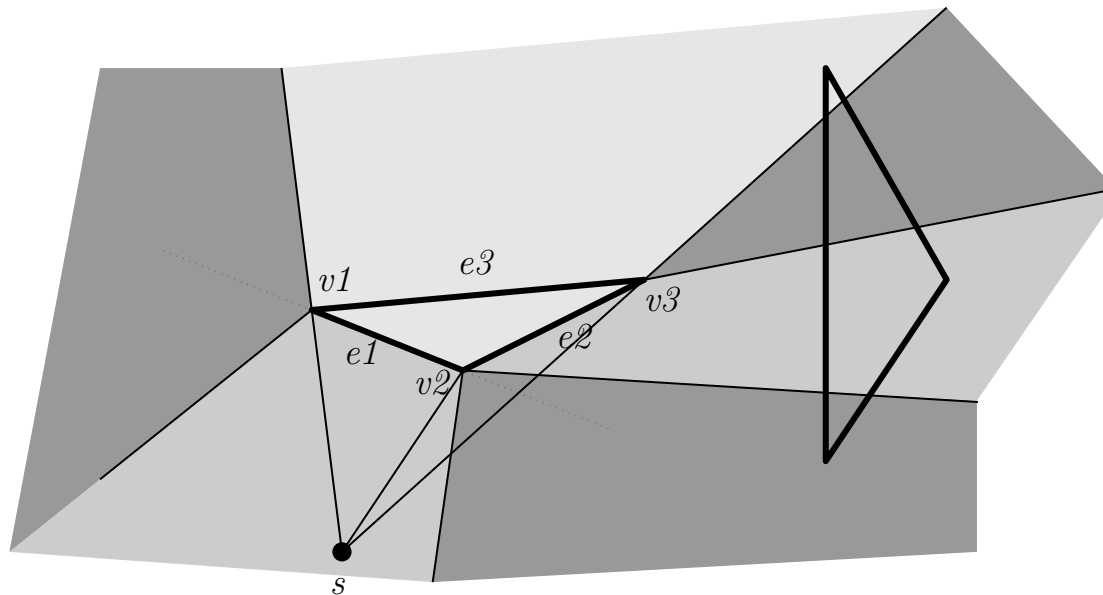
Benutzung der Last Step SPM: Alg. 1.10

- Ziel t_1 : Nutze Last Step SPM von P_k
- Induktiv nutze Last Step SPM von P_{k-1}



Benutzung der Last Step SPM: Alg. 1.10

- Ziel t_1 : Nutze Last Step SPM von P_k
- Induktiv nutze Last Step SPM von P_{k-1}



Analyse der Query! **Lemma 1.33**

- Annahme: Alle Last Step SPM sind gegeben

-

$$\sum_{i=1}^k \log n_i \quad \text{mit} \quad \sum_{i=1}^k n_i = n$$

- Worst-Case:

$$n_i = \frac{n}{k}$$

- Alles zusammen:

$$O\left(k \log \frac{n}{k}\right)$$

Berechnung S_1, \dots, S_k : Alg. 1.11

Berechnung S_1, \dots, S_k : Alg. 1.11

- Queries: Backward

Berechnung S_1, \dots, S_k : Alg. 1.11

- Queries: Backward
- Berechnung: Forward

Berechnung S_1, \dots, S_k : Alg. 1.11

- Queries: Backward
- Berechnung: Forward
- SPM für P_1

Berechnung S_1, \dots, S_k : Alg. 1.11

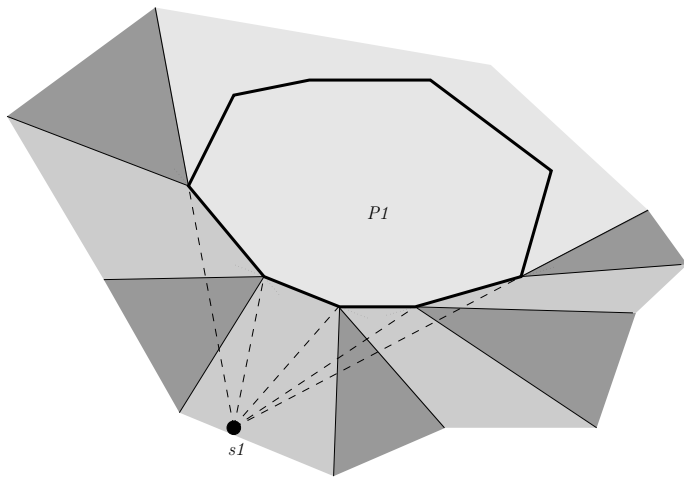
- Queries: Backward
- Berechnung: Forward
- SPM für P_1
- Sichtbare konvexe Kette/Baum der Strahlen: $O(n_1)$

Berechnung S_1, \dots, S_k : Alg. 1.11

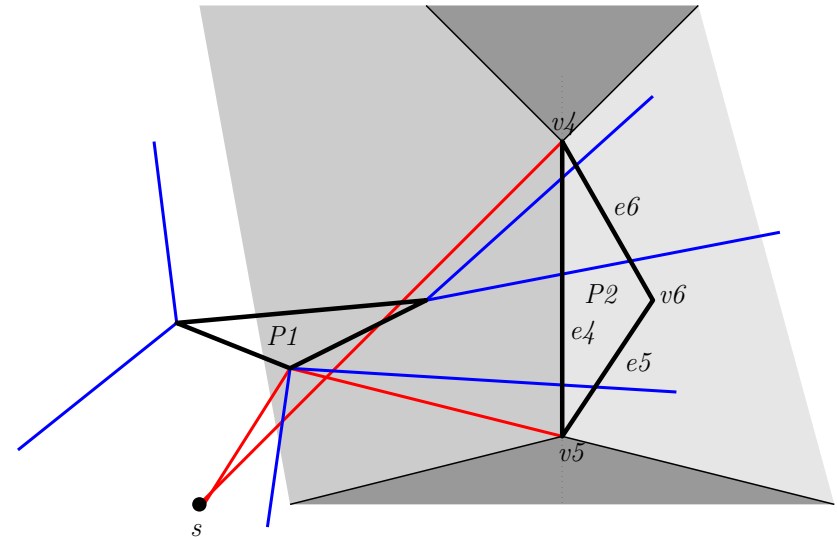
- Queries: Backward
- Berechnung: Forward
- SPM für P_1
- Sichtbare konvexe Kette/Baum der Strahlen: $O(n_1)$
- Disjunkte Reflektionen!!

Berechnung S_1, \dots, S_k : Alg. 1.11

- Queries: Backward
- Berechnung: Forward
- SPM für P_1
- Sichtbare konvexe Kette/Baum der Strahlen: $O(n_1)$
- Disjunkte Reflektionen!!

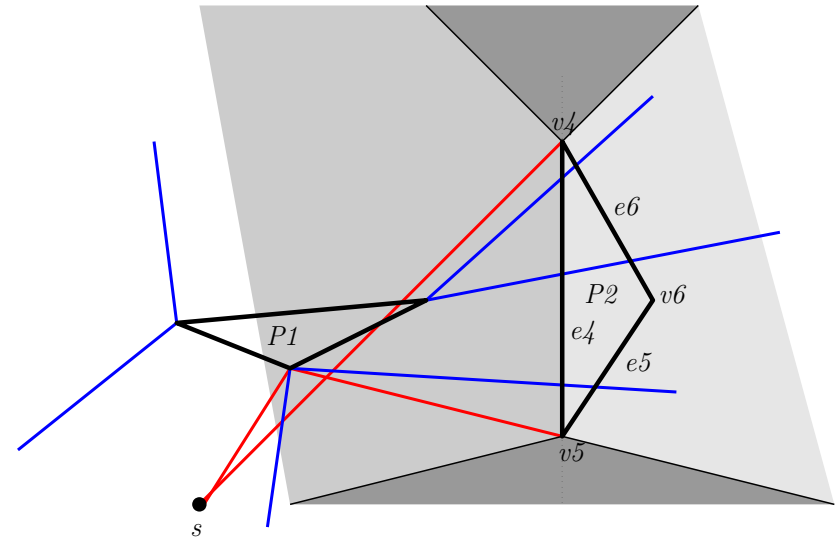


Berechnung S_1, \dots, S_k : Alg. 1.11



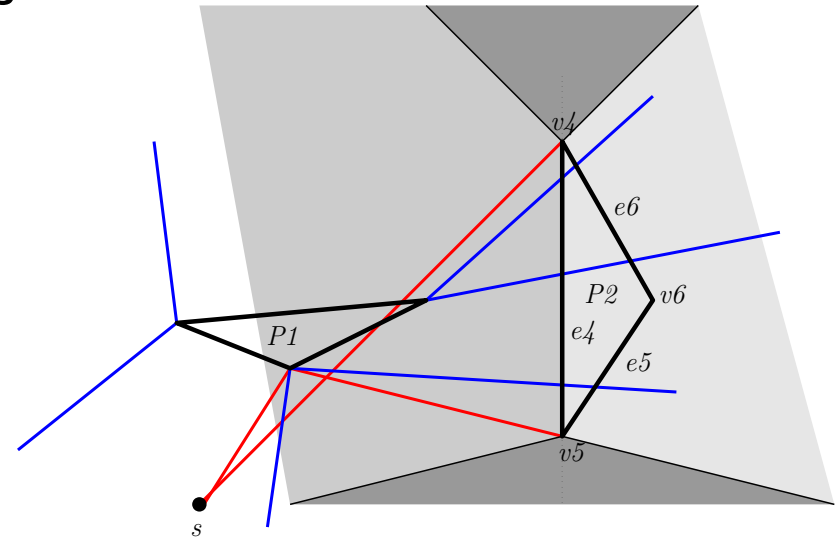
Berechnung S_1, \dots, S_k : Alg. 1.11

- SPM von P_i aus SPM of P_{i-1}, \dots, P_1



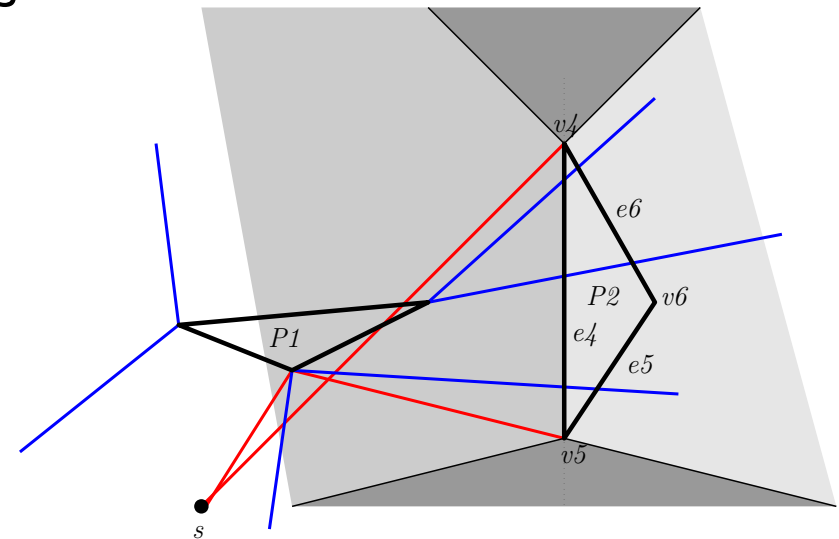
Berechnung S_1, \dots, S_k : Alg. 1.11

- SPM von P_i aus SPM of P_{i-1}, \dots, P_1
- Letztes Segment des Kürzesten Weges von s zu Knoten von P_i



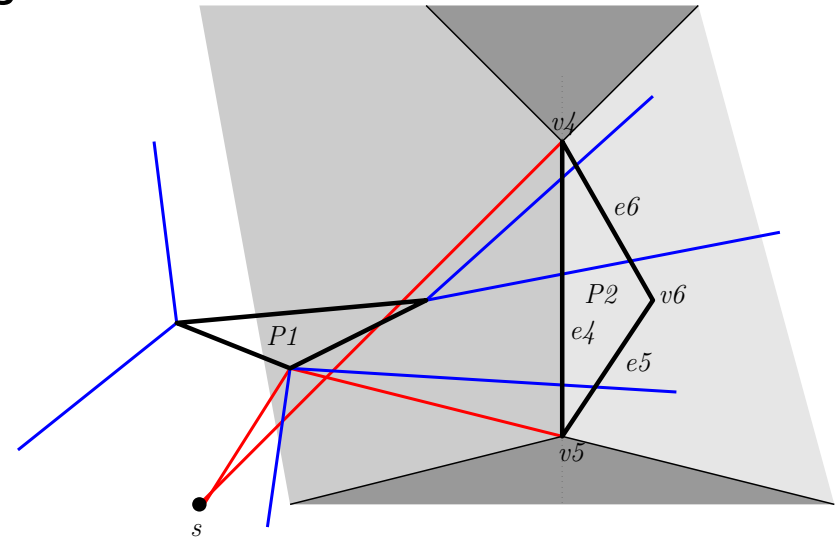
Berechnung S_1, \dots, S_k : Alg. 1.11

- SPM von P_i aus SPM of P_{i-1}, \dots, P_1
- Letztes Segment des Kürzesten Weges von s zu Knoten von P_i
- Query: Nutze SPM P_{i-1}, \dots, P_1



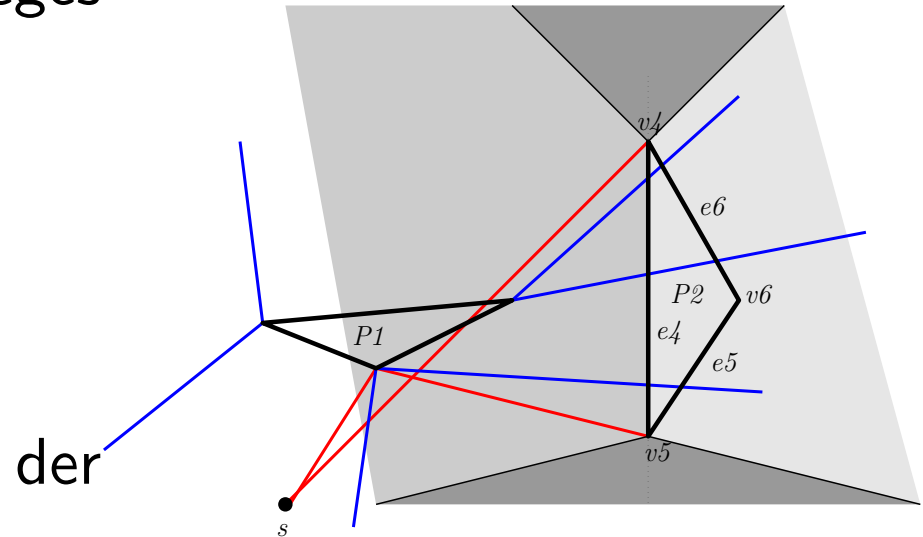
Berechnung S_1, \dots, S_k : Alg. 1.11

- SPM von P_i aus SPM of P_{i-1}, \dots, P_1
- Letztes Segment des Kürzesten Weges von s zu Knoten von P_i
- Query: Nutze SPM P_{i-1}, \dots, P_1
- Laufzeit: $O\left(n_i(i-1) \log \frac{N_{i-1}}{i-1}\right)$
mit $N_j := \sum_{l=1}^j n_l$



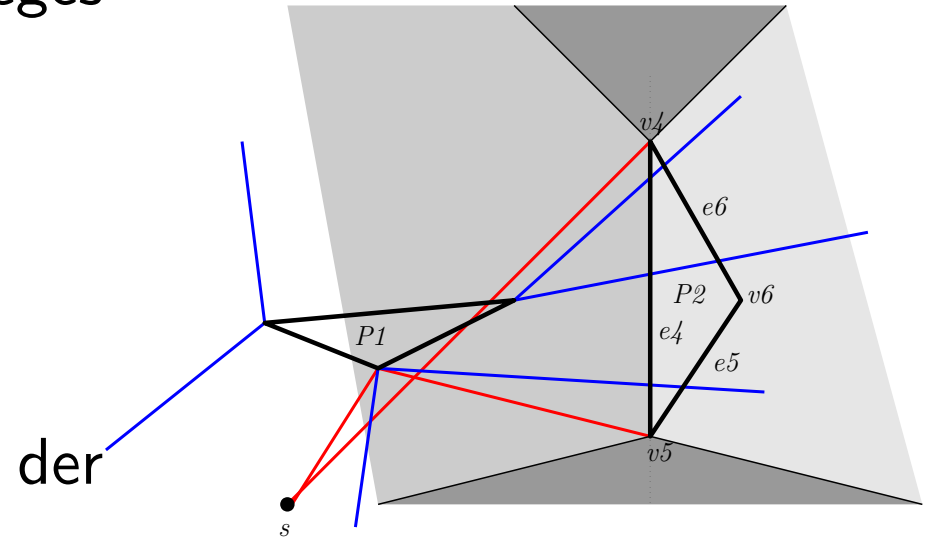
Berechnung S_1, \dots, S_k : Alg. 1.11

- SPM von P_i aus SPM of P_{i-1}, \dots, P_1
- Letztes Segment des Kürzesten Weges von s zu Knoten von P_i
- Query: Nutze SPM P_{i-1}, \dots, P_1
- Laufzeit: $O\left(n_i(i-1) \log \frac{N_{i-1}}{i-1}\right)$
mit $N_j := \sum_{l=1}^j n_l$
- Sichtbare konvexe Kette/Baum der Strahlen



Berechnung S_1, \dots, S_k : Alg. 1.11

- SPM von P_i aus SPM of P_{i-1}, \dots, P_1
- Letztes Segment des Kürzesten Weges von s zu Knoten von P_i
- Query: Nutze SPM P_{i-1}, \dots, P_1
- Laufzeit: $O\left(n_i(i-1) \log \frac{N_{i-1}}{i-1}\right)$
mit $N_j := \sum_{l=1}^j n_l$
- Sichtbare konvexe Kette/Baum der Strahlen
- Disjunkt wegen konvexer Kette!!



Analyse Berechnung S_1, \dots, S_k : **Theorem 1.34**

Analyse Berechnung S_1, \dots, S_k : **Theorem 1.34**

Rekursiv: P_2, \dots, P_k

Analyse Berechnung S_1, \dots, S_k : **Theorem 1.34**

Rekursiv: P_2, \dots, P_k

Gesamtlaufzeit:

Analyse Berechnung S_1, \dots, S_k : **Theorem 1.34**

Rekursiv: P_2, \dots, P_k

Gesamtlaufzeit:

$$\sum_{i=2}^k n_i (i-1) \log \frac{N_{i-1}}{i-1}$$

Analyse Berechnung S_1, \dots, S_k : **Theorem 1.34**

Rekursiv: P_2, \dots, P_k

Gesamtlaufzeit:

$$\sum_{i=2}^k n_i(i-1) \log \frac{N_{i-1}}{i-1}$$

$$n_i(i-1) \log \frac{N_{i-1}}{i-1} \leq n_i k \log \frac{n}{k}$$

Analyse Berechnung S_1, \dots, S_k : **Theorem 1.34**

Rekursiv: P_2, \dots, P_k

Gesamtlaufzeit:

$$\sum_{i=2}^k n_i (i-1) \log \frac{N_{i-1}}{i-1}$$

$$n_i (i-1) \log \frac{N_{i-1}}{i-1} \leq n_i k \log \frac{n}{k}$$

Gesamtlaufzeit:

Analyse Berechnung S_1, \dots, S_k : **Theorem 1.34**

Rekursiv: P_2, \dots, P_k

Gesamtlaufzeit:

$$\sum_{i=2}^k n_i (i-1) \log \frac{N_{i-1}}{i-1}$$

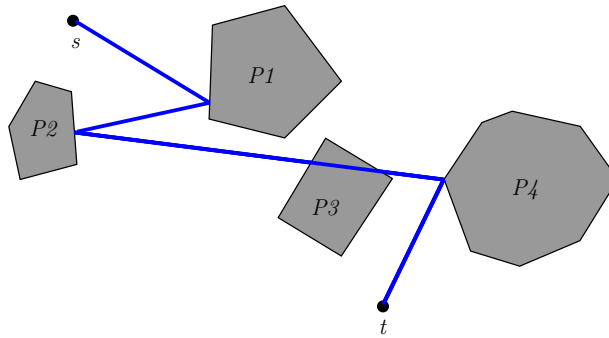
$$n_i (i-1) \log \frac{N_{i-1}}{i-1} \leq n_i k \log \frac{n}{k}$$

Gesamtlaufzeit:

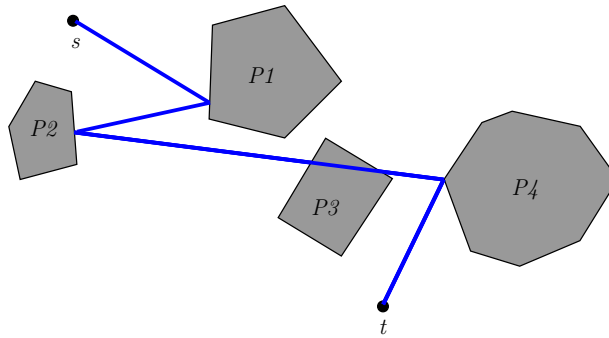
$$O\left(kn \log \frac{n}{k}\right)!$$

Zusammenfassung!!

Zusammenfassung!!



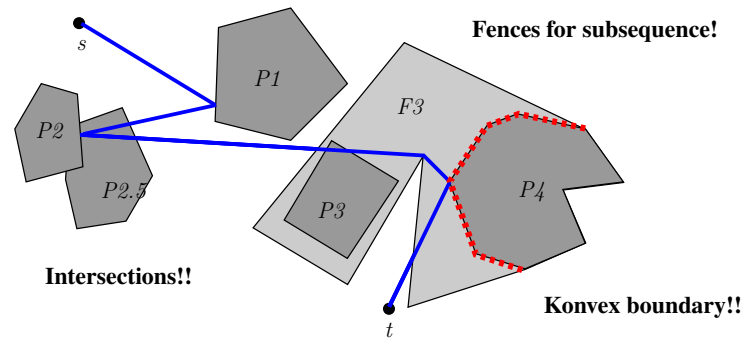
Zusammenfassung!!



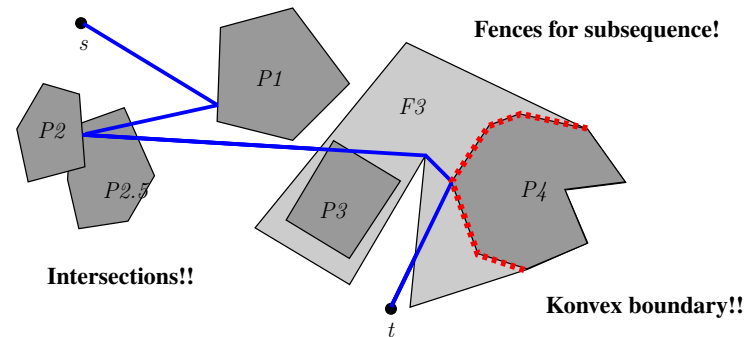
- Einfache Version:
- Disjunkte, konvexe Polygone, keine Zäune
- $O(nk \log \frac{n}{k})$
- Build(Query): $O(nk \log \frac{n}{k})$
- Komplexität: $O(n)$
- Query (festes s): $O(k \log \frac{n}{k})$
- **Theorem 1.34**

Erweiterung!

Erweiterung!

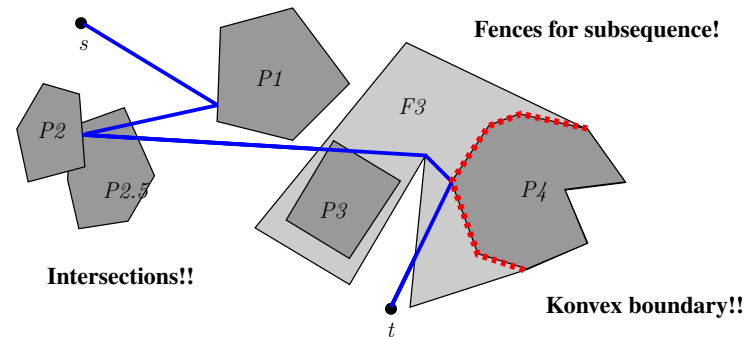


Erweiterung!



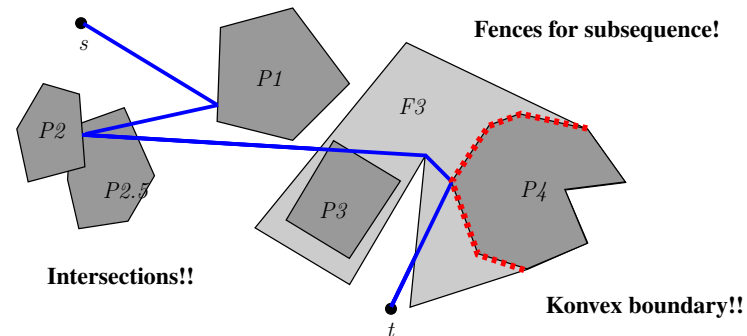
- Komplexe Version:

Erweiterung!



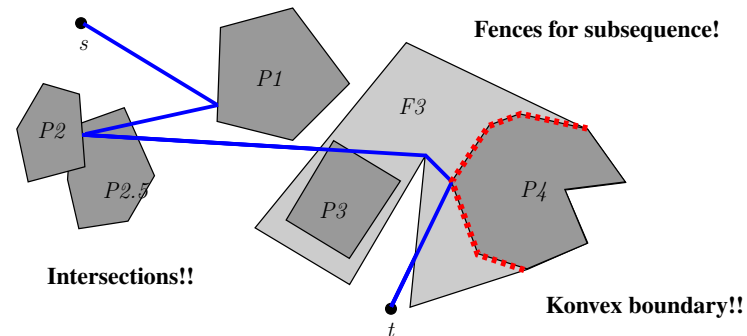
- Komplexe Version:
- Nicht-disjunkte, konvexe Polygone, Zäune

Erweiterung!



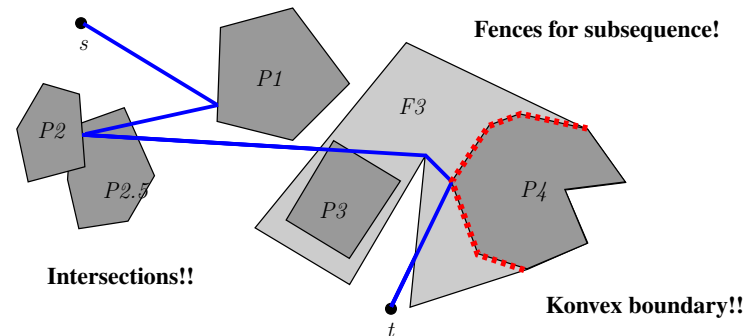
- Komplexe Version:
- Nicht-disjunkte, konvexe Polygone, Zäune
- $O(nk^2 \log n)$ insgesamt

Erweiterung!



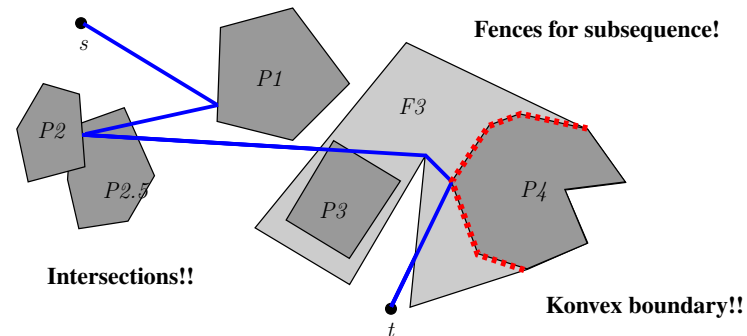
- Komplexe Version:
- Nicht-disjunkte, konvexe Polygone, Zäune
- $O(nk^2 \log n)$ insgesamt
- Build(Query): $O(nk^2 \log n)$

Erweiterung!



- Komplexe Version:
- Nicht-disjunkte, konvexe Polygone, Zäune
- $O(nk^2 \log n)$ insgesamt
- Build(Query): $O(nk^2 \log n)$
- Komplexität: $O(kn)$

Erweiterung!



- Komplexe Version:
- Nicht-disjunkte, konvexe Polygone, Zäune
- $O(nk^2 \log n)$ insgesamt
- Build(Query): $O(nk^2 \log n)$
- Komplexität: $O(kn)$
- Query (festes s): $O(kn)$

- Lemma 1.35: Reflexionsbereich ist Baum

- Lemma 1.35: Reflexionsbereich ist Baum
- Theorem 1.36/Theorem 1.37