Offline Bewegungsplanung: Reine Translation

Elmar Langetepe
University of Bonn

Folgerungen!! Theorem 2.16

- Konvexer Roboter R mit $|R| = m$, polygonale Szene, n Kanten
- Komplexität von C_{verb} in $O(mn)$, Beweis!!
- Trianguliere alle P_j, $T_1, T_2, \ldots T_l$ mit $O(n)$ Kanten
- $CT_i = T_i \oplus -R(0,0)$ Familie von Pseudokreisen Lem. 2.12, $O((m+3)n)$ Kanten Lem. 2.15

$$C_{\text{verb}} = \underbrace{\bigcup_i T_i \oplus -R(0,0)}_{\text{Komplexität } O(mn)}.$$

Benutze Theorem 2.13

Folgerungen!! Theorem 2.16

- Algorithmus 2.2: Berechnung von C_{verb}
  - Unterteile Hindernisse in gleichgroße Teilmengen von Dreiecken CP_1 und CP_2
  - Berechne rek. $C_{\text{verb}}(CP_1)$ und $C_{\text{verb}}(CP_2)$, je $O(mn)$ Kanten
  - Berechne Vereinigung $C_{\text{verb}}(CP_1) \cup C_{\text{verb}}(CP_2)$ in Zeit $O(mn \log(mn))$ mit Sweep (Alg. Geom.)

Laufzeit insgesamt: $O(mn \log^2 mn)$

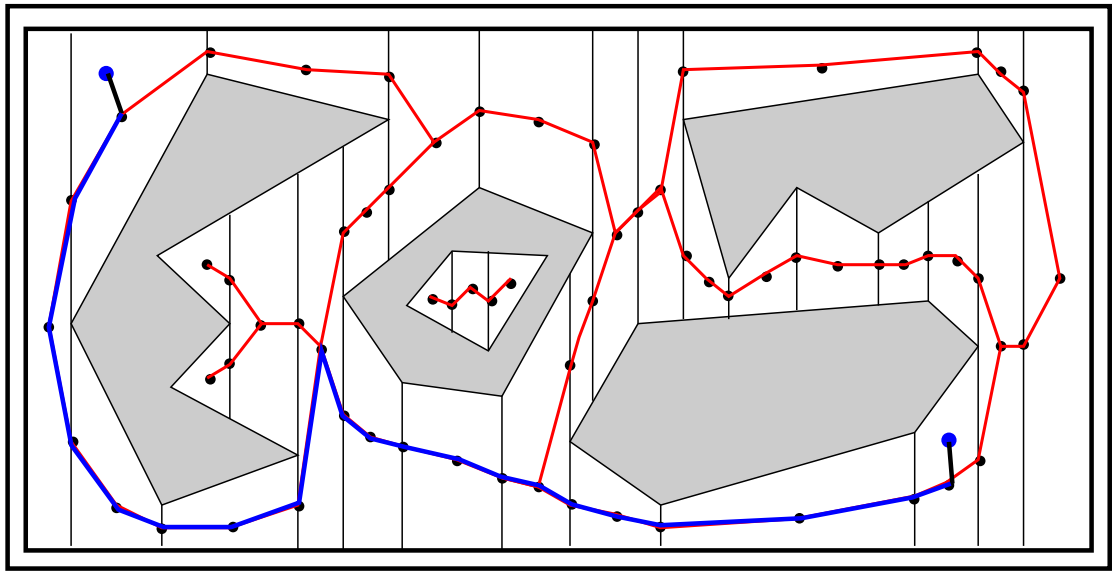
Laufzeitanalyse: Theorem 2.16

Arrangement mit $O(nm)$ Kanten!!

$$
\begin{aligned}
T(m,n) &\leq 2T\left(m,\frac{n}{2}\right) \text{(Rek.)} + C \times mn \log mn \text{(Merge)} \\
&\leq 2\left(2T\left(m,\frac{n}{4}\right) + C\frac{mn}{2}\log\frac{mn}{2}\right) + C \times mn \log mn \\
&\vdots \quad \vdots \\
&\leq nT(m,1) + Cmn \sum_{i=0}^{\log n} \log \frac{mn}{2^i} \\
&\in O(mn \log^2 mn)
\end{aligned}
$$

C_{frei} **verwenden! Roadmap!!**

- Zerlegung in Trapezen (Seidel)
- Benachbarte Trapeze verbinden (Schwerpunkt/Mittelpunkt)
- Zusammenhangsgraph, Roadmap
- Lokalisation s und t und verbinden (BFS)



Algorithmus 2.3: C_{frei} gegeben!

Vorbereitung:

- Trapezzerlegung C_{frei}/Point–Location Struktur, (randomisiert!!) $O(mn \log(mn))$
- Roadmap-Konstruktion: Verbindung adjazenter Trapeze, $O(mn)$

Query:

- Lokalisiere s und t Trapeze, $O(\log(mn))$
- Finde Weg der Trapeze mit BFS, $O(mn)$

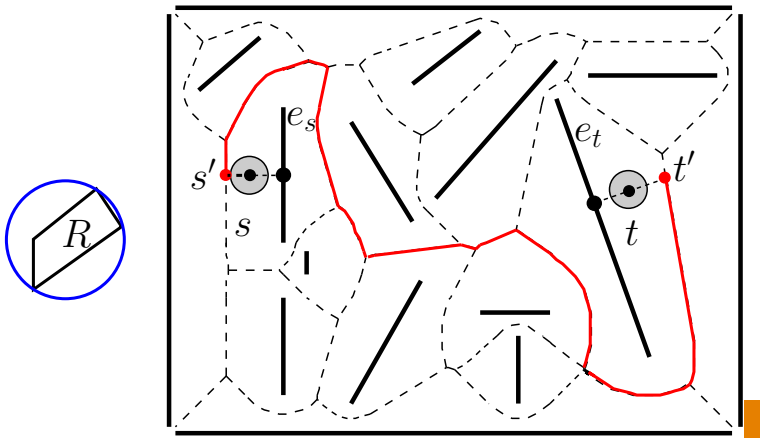
Roadmap!! Einfach aber effizient!! Probabilistisch!

Ergebnis: Theorem 2.17

Translationsbewegungen eines konvexen, polygonalen Roboters mit m Ecken in einer Umgebung mit polygonalen Hindernissen mit insgesamt n Ecken können nach $O(mn \log^2(mn))$ (randomisierter) Vorbereitungszeit in Zeit $O(mn)$ geplant werden.

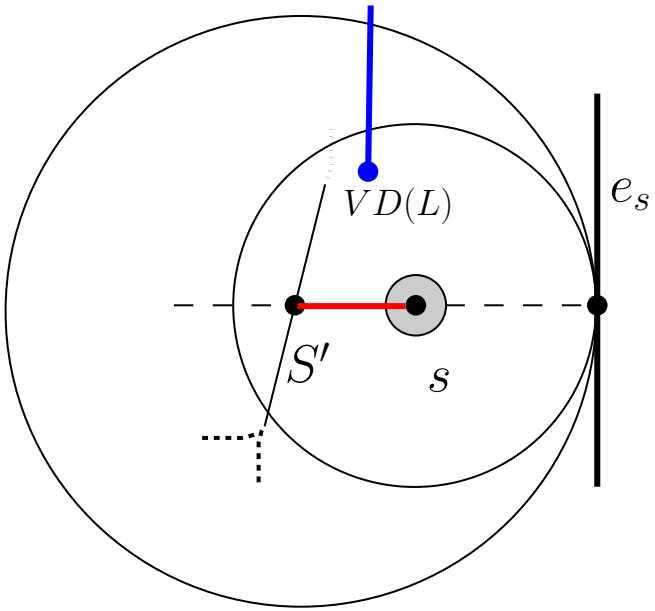
Kreisförmiger! Voronoi Diagramme 2.1.1

- Verwende kleinsten Kreis um Roboter
- Voronoi Diagramm der Segmente der Hindernisse
- Weg auf Bisektoren: Möglichst großer Abstand



Start s' kann stets angelaufen werden

- s in Region von e_s, Kreis frei
- Kürzester Weg zu e_s, Strahl Richtung Bisector
- Trifft Bisector bei s', Weg ist frei!!



Algorithmus 2.1: Kreisförmiger Roboter

Vorbereitung:

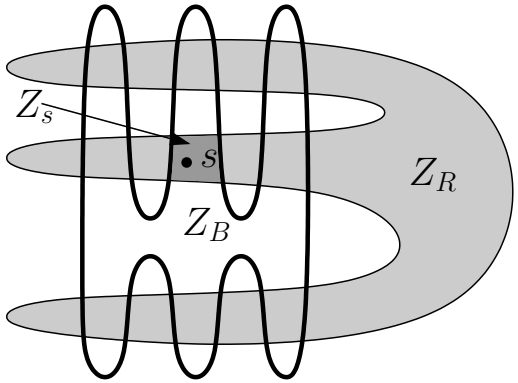
- Konstruiere das Voronoi–Diagramm $VD(L)$, Kanten der Hindernisse als Liniensegmente: $O(n \log n)$
- Point–Location Trapezzerlegung: $O(n \log n)$

Query für zwei Punkte s, t:

- Lokalisiere s, t im VD: $O(\log n)$
- s gehört zu e_s, t gehört zu e_t
- Bestimme mit Strahlen Punkt t' und s' auf $VD(L)$: $O(1)$
- Berechne (BFS) in $VD(L)$ Pfad von s' nach t' mit zul. Mindestabstand oder berichte, dass es keinen gibt: $O(n)$

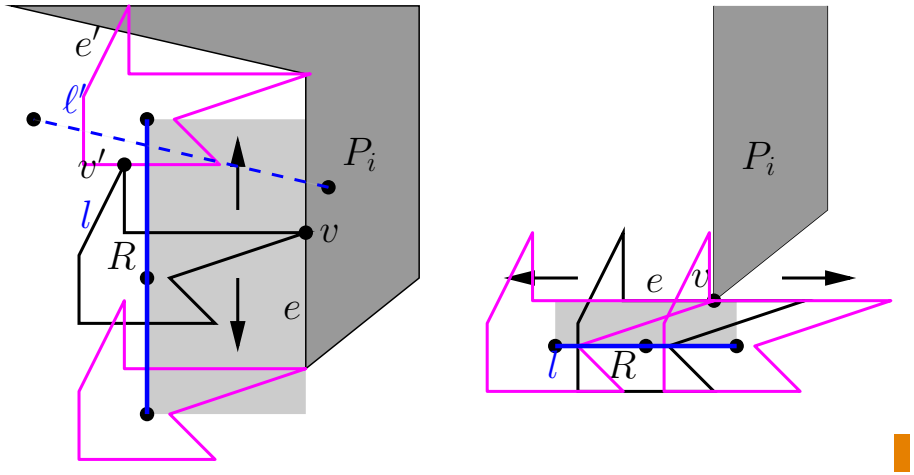
Kreisförmig, konvex! Beliebiger Roboter!

- Bisher: Divide and Conquer Konfigurationsraum
- Nicht konvex: Komplexität $\Theta((nm)^2)$
- Merge komplett: $\Theta((nm)^2)$
- Beobachtung: Zelle Z_s nicht so komplex??



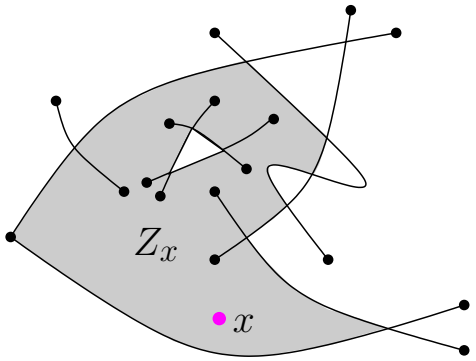
Begrenzung einer Zelle!!

- Nicht-konvexer Roboter R mit $|R| = m$. Polygonale Szene n Ecken
- Ecke Roboter, Kante Hindernis
- Ecke Hindernis, Kante Roboter
- $\Theta(mn)$ viele Kanten



Allgemeiner!!

- Menge von Segmenten
- Je zwei schneiden sich s mal
- Spezieller Punkt x
- Komplexität der Zelle Z_x



Fahrplan!!

- Divide and Conquer!!
- Teile Segmente in zwei gleichgroße Mengen Z_1, Z_2
- Berechne Z_{1x} und Z_{2x}
- Merge zu $\{Z_1 \cup Z_2\}_x$
- Spezieller Merge wegen Schnitt mit x
- RED BLUE Merge
- Merge: Komplexität des Ergebnisses

Exkurs: Davenport-Schinzel-Sequenzen

Alphabet $\Sigma = \{A, B, C, \ldots\}$ über n Buchstaben.

Davenport–Schinzel–Sequenz der Ordnung s

- Wort w über Σ
- in w keine benachbarten Buchstaben gleich
- keine zwei verschiedenen Buchstaben wechseln mehr als s mal

Bsp.: ABRAKADABRA; max. 4 Wechsel (A und B)

$\lambda_s(n)$ maximale Länge eines solchen Wortes

Davenport-Schinzel-Sequenzen: Th. A9

Fast linear!! (Ohne Beweis!)

$$
\begin{aligned}
\lambda_1(n) &= n \\
\lambda_2(n) &= 2n-1 \\
\lambda_3(n) &\in \Theta(n\,\alpha(n)) \\
\lambda_4(n) &\in \Theta(n \cdot 2^{\alpha(n)}) \\
\lambda_s(n) &\in O(n \log^*(n)) \in O(n^2)
\end{aligned}
$$

$\alpha(n)$ Inverse Ackermann Fkt.

$\log^*(n)$

Untere Kontur: Def. A10

$f_1, f_2, \ldots, f_n$ reellwertige Funktionen, entweder

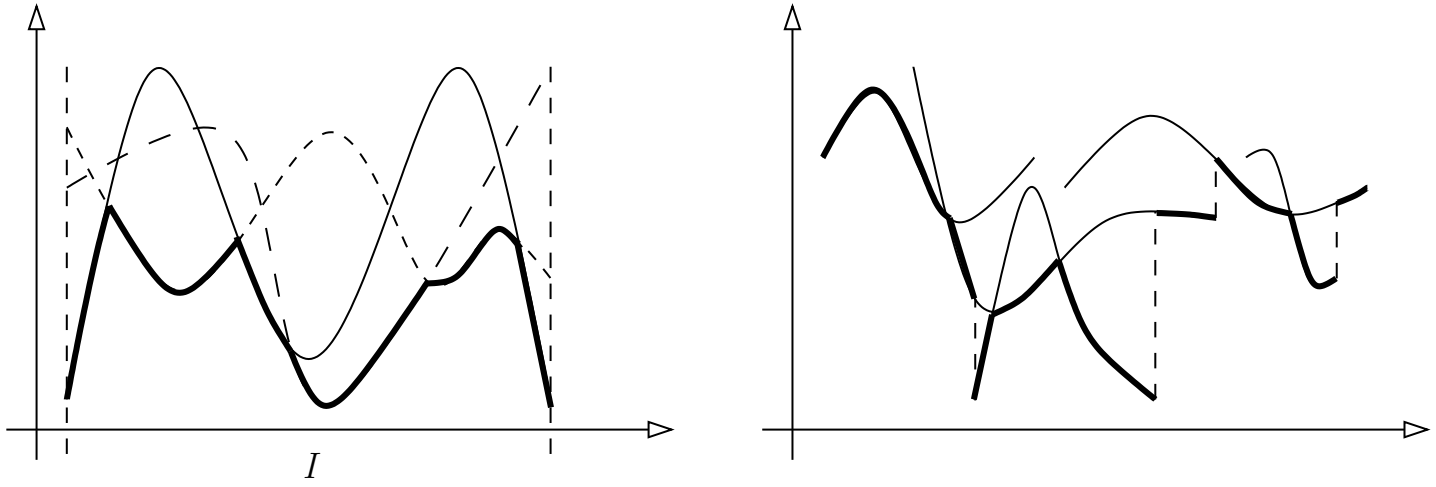
- über einem gemeinsamen Intervall I oder
- über je einem Intervall $I_j \subseteq I$, $1 \leq j \leq n$.

Je zwei Funktionen max. s gemeinsame Schnittpunkte

$L(x) := \min \{ f_i(x) | 1 \leq i \leq n \}$

Lower envelope der Funktionsgraphen

Untere Kontur: Th. A12



(1) (2)

$L(x)$ besteht aus maximal

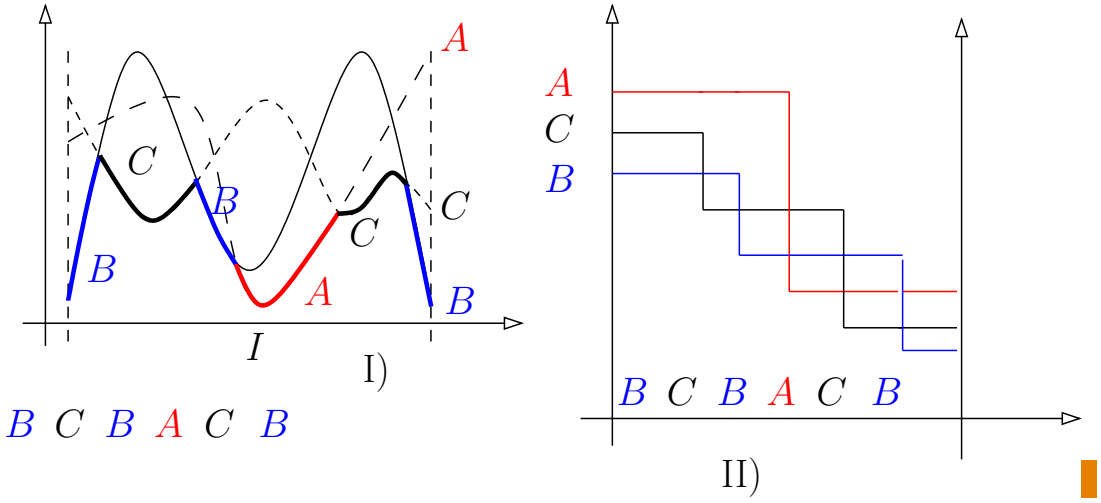
1. $\lambda_s(n)$
2. $\lambda_{s+2}(n)$

vielen Teilstücken

Beweis: Th. A12

1. $\lambda_s(n)$
2. $\lambda_{s+2}(n)$

1.: I) Übersetzen in Buchst.-folge oder II) Buchst.-folge übersetzen

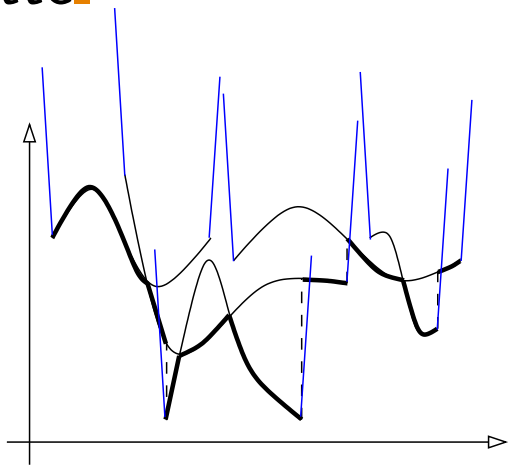


Beweis: Th. A12

1. $\lambda_s(n)$
2. $\lambda_{s+2}(n)$

2. Verlängern auf gesamtes Intervall, dann 1. verwenden

Max zwei zusätzliche Schnitte

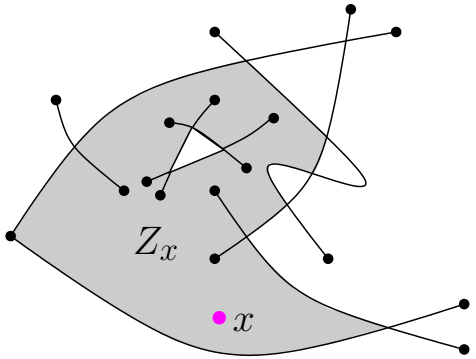


(2)

Zellen: Th. 2.18

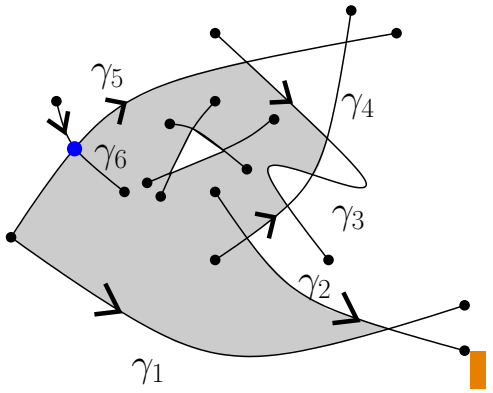
A Arrangement von n Kurvenstücken von denen sich zwei nur s mal schneiden. Jede Zelle von A hat Komplexität $O(\lambda_{s+2}(n))$.

Weniger als $\Omega(n^2)$!! Beweis!!



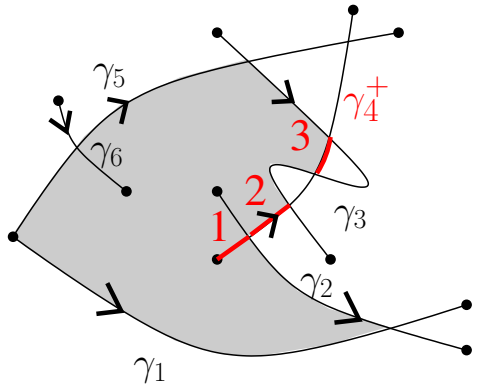
Beweis: Th. 2.18

- Rand zerfällt in mehere Zyklen
- Analyse: Äußerer Zyklus C reicht! $\lambda_s(n)$, n Segmente
- Bezeichnen, orientieren: links nach rechts
- Zyklische Abfolge
  $S =< \gamma_5^- \; \gamma_1^+ \; \gamma_2^- \; \gamma_4^- \; \gamma_4^+ \; \gamma_2^- \; \gamma_2^+ \; \gamma_4^+ \; \gamma_3^- \; \gamma_4^+ \; \gamma_3^- \; \gamma_5^- \; \gamma_6^+ \; \gamma_6^- >$
- $2n$ Segmente γ_i^- ,γ_i^+



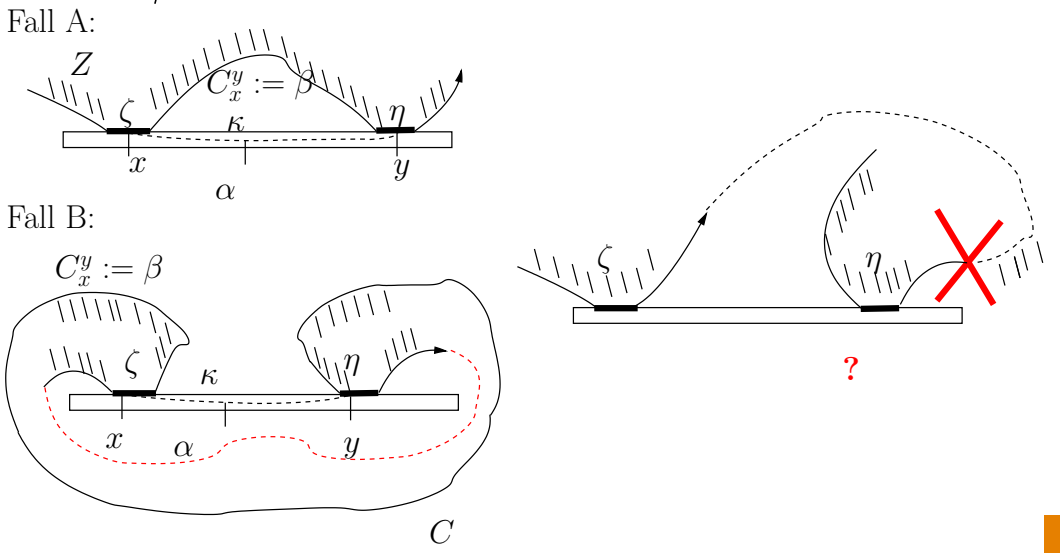
Bogenteile in Reihenfolge! Lem. 2.19

- Zyklische *Folge* des Zykels C:
  $S =< \gamma_5^- \; \gamma_1^+ \; \gamma_2^- \; \gamma_4^- \; \gamma_4^+ \; \gamma_2^- \; \gamma_2^+ \; \gamma_4^+ \; \gamma_3^- \; \gamma_4^+ \; \gamma_3^- \; \gamma_5^- \; \gamma_6^+ \; \gamma_6^- >$
- Bogen γ_i: Segmente von $\gamma_i^+(\gamma_i^-)$ erscheinen in C in derselben Reihenfolge wie entlang von $\gamma_i^+(\gamma_i^-)$
- Beispiel γ_4^+! Beweis!



Beweis: Konsistenzlemma 2.19

- γ_i etwas aufblasen; $\zeta, \eta \in \gamma_i^*$ konsekutive Subsegmente in C!
- Nur zwei Fälle gemäß Innerem; sonst keine Verbindung!!
- Betrachte α und $\beta := C_x^y$
- β berührt γ_i^* nicht: Konsekutiv in C!
- C schnittfrei: $\alpha \cup \beta$ trennt κ von C ab



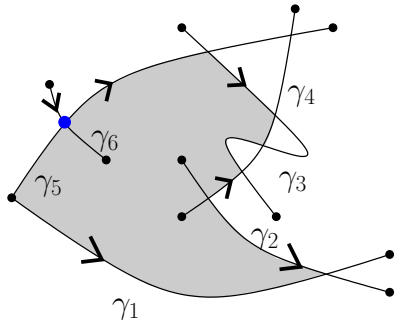
Lineare Sequenz S'' bilden!

- Zyklische Sequenz:

  $S =< \gamma_5^- \; \gamma_1^+ \; \gamma_2^- \; \gamma_4^- \; \gamma_4^+ \; \gamma_2^- \; \gamma_2^+ \; \gamma_4^+ \; \gamma_3^- \; \gamma_4^+ \; \gamma_3^- \; \gamma_5^- \; \gamma_6^+ \; \gamma_6^- >$ auftrennen
- Orientierte Sequenz:

  $S' = \{\gamma_5^- \; \gamma_1^+ \; \gamma_2^- \; \gamma_4^- \; \gamma_4^+ \; \gamma_2^- \; \gamma_2^+ \; \gamma_4^+ \; \gamma_3^- \; \gamma_4^+ \; \gamma_3^- \; \gamma_5^- \; \gamma_6^+ \; \gamma_6^-\}$
- Orientierte Reihenfolge stimmt nicht! Bsp.:γ_5^-
- Verdoppeln: $\gamma_i^- \; (\gamma_i^+) \Rightarrow \gamma_{i,1}^-, \; \gamma_{i,2}^-(\gamma_{i,1}^+, \; \gamma_{i,2}^+)$
- Nun 4n Segmente: Sequenz

  $S'' = \{\gamma_{5,1}^- \; \gamma_1^+ \; \gamma_2^- \; \gamma_4^- \; \gamma_4^+ \; \gamma_2^- \; \gamma_2^+ \; \gamma_4^+ \; \gamma_3^- \; \gamma_4^+ \; \gamma_3^- \; \gamma_{5,2}^- \; \gamma_6^+ \; \gamma_6^-\}$



S'' DSS mit $(4n, s+2)$: Lem. 2.20

- Definition DSS
- Zwischen zwei $\gamma^-_{i,j}(\gamma^+_{i,j})$ kommt stets anderes Segment vor
- Zz.: Zwei Buchstaben ζ und η wechseln nicht mehr als $s+2$ mal
- Widerspruchsbeweis: Annahme $s+3$ Wechsel!
- Situation:
  $S'' = (\cdots \zeta_1 \cdots \eta_1 \cdots \zeta_2 \cdots \eta_2 \cdots\cdots \zeta_j \cdots \eta_j \cdots \zeta_k \cdots (\eta_k \cdots))$

Lem. 2.20

$S'' = (\cdots \zeta_1 \cdots \eta_1 \cdots \zeta_2 \cdots \eta_2 \cdots\cdots \zeta_j \cdots \eta_j \cdots \zeta_k \cdots (\eta_k \cdots))$

Fasse je vier zusammen:

$$\begin{array}{llll} (\zeta_1 & ,\eta_1 & ,\zeta_2 & ,\eta_2) \\ & (\eta_1 & ,\zeta_2 & ,\eta_2, \quad \zeta_3) \\ & & (\zeta_2 & ,\eta_2, \quad \zeta_3, \quad \eta_3) \\ & & \vdots & \end{array}$$

Jeweils Schnitt zw. $((\zeta_1, \zeta_2), (\eta_1, \eta_2))$, $((\eta_1, \eta_2), (\zeta_2, \zeta_3))$,...

$s+3$ ungerade: η_k ex. und $k = \frac{s+2}{2} + 1$ Induktion!!

($(s+3)$ gerade Übung!)

Lem. 2.20

$s+3$ ungerade: η_k ex. und $k = \frac{s+2}{2} + 1$ Induktion!!

$$
\begin{array}{llll}
(\zeta_1 & ,\eta_1 & ,\zeta_2 & ,\eta_2) \\
 & (\eta_1 & ,\zeta_2 & ,\eta_2, \quad \zeta_3) \\
 & & \vdots & \\
 & & (\zeta_{\frac{s+2}{2}} & ,\eta_{\frac{s+2}{2}} \quad ,\zeta_{\frac{s+2}{2}+1} \quad ,\eta_{\frac{s+2}{2}+1})
\end{array}
$$

ζ führt $\frac{s+2}{2}$ Quadrupel an! η führt $\frac{s+2}{2} - 1$ Quadrupel an!

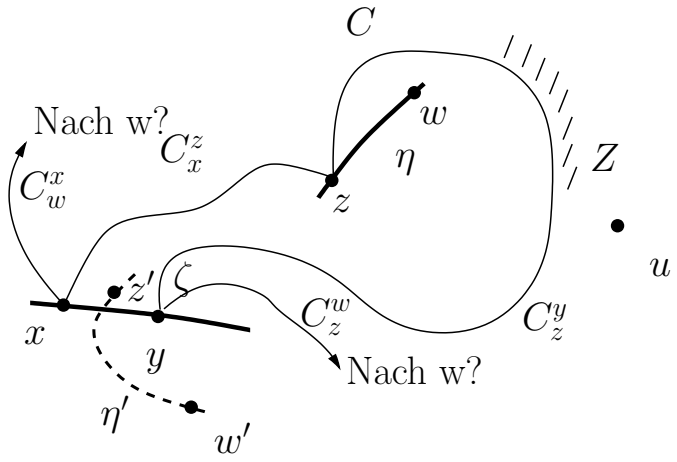
Insgesamt: $2 \cdot (\frac{s+2}{2}) - 1 = s + 1$ Quadrupel

Noch zu zeigen: Jedes Quadrupel erzeugt Schnitt!

Lem. 2.20

Zu zeigen: O.B.d.A: $(\zeta_i, \eta_i, \zeta_{i+1}, \eta_{i+1})$ erzeugt einzelnen Schnitt zw. (ζ_i, ζ_{i+1}) und (η_i, η_{i+1})

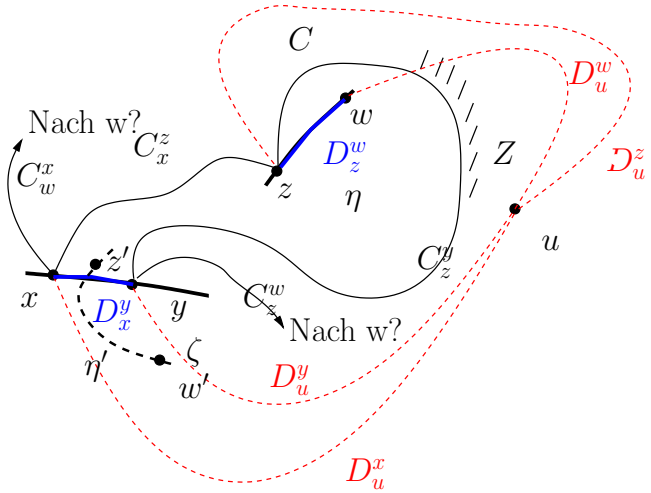
Situation wie folgt:



Zeige, dass Schnitt existieren muss!

Situation: u im Innern!

- u *sieht* x, z, y, w
- Verb. D_u^x, D_u^z, D_u^y, D_u^w schnittfrei mit C_x^z, C_z^y, C_y^w, C_w^x
- Annahme: D_x^y und D_z^w schnittfrei $\Rightarrow$ alle schnittfrei!!
- Entspricht: $K5$ in der Ebene schnittfrei realisiert! Widerspruch!!



Konklusion: Lem. 2.20

- n Bögen, je s Schnitte: Zeige: DSS mit $(4n, s+2)$
- Annahme: DSS mit mind. $(4n, s+3)$
- $(s+3)$ Wechsel auf ζ und η, sukzessive
- Bei jedem Quadrupel $(\zeta_i, \eta_i, \zeta_{i+1}, \eta_{i+1})$ (oder $(\eta_i, \zeta_{i+1}, \eta_{i+1}, \zeta_{i+2})$: Schnitt zwischen (ζ_i, ζ_{i+1}) und (η_i, η_{i+1})
- $s+1$ Quadrupel $\Rightarrow$ $s+1$ Schnitte, Widerspruch!!
- DSS mit $(4n, s+2)$

Zurück zur Aufgabe: Konfigurationsraum

- n Bögen
- Je zwei schneiden sich s mal
- X-monoton, eventuell erzeugen
- Startpunkt x
- Komplexität der Zelle Z_x: $\lambda_{s+2}(4n)$
- Divide and Conquer Ansatz sinnvoll

Alg. 2.4: Zelle Z_x!

Gegeben: Punkt x, n X-monotone Bögen.

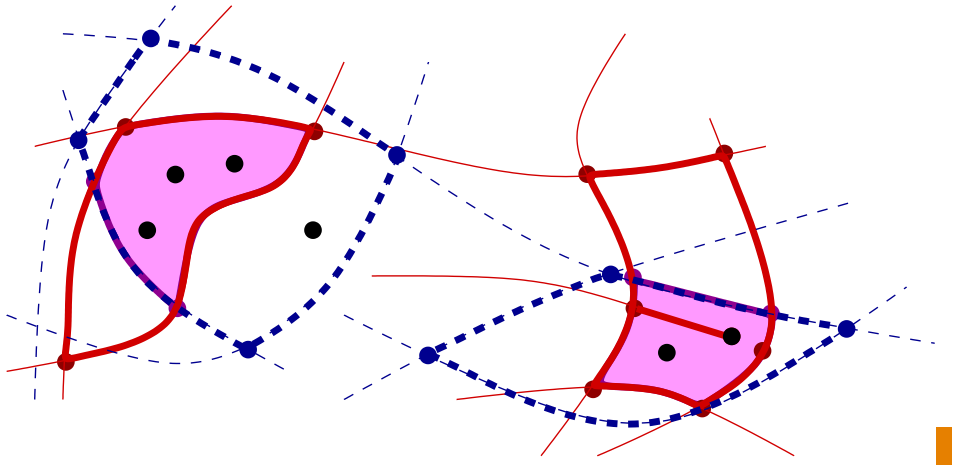
Gesucht: Zelle Z_x im Arrangement der Bögen, die x enthält.

- Zerlege Menge der Bögen in gleichgroße Teilmengen R und B.
- Berechne rek. im Arrangement $A(R)$ Zelle $Z(R)_x$, die x enthält.
- Berechne rek. im Arrangement $A(B)$ Zelle $Z(B)_x$, die x enthält.
- Berechne Zusammenhangskomponente Z_x von $Z(R)_x \cap Z(B)_x$, die x enthält und berichte diese! RED-BLUE Merge

Zuerst RED-BLUE Merge betrachten, dann zurück!

Allgemeiner RED-BLUE Merge

- Rotes Arrangement R mit Zellen $R_1, \ldots, R_{m_R}$, r Ecken.
- Blaues Arrangement B mit Zellen $B_1, \ldots, B_{m_B}$, b Ecken.
- Punktmenge p_i $i = 1, \ldots, k$
- Schnittzellen $Z_j = R_{\mu_j} \cap B_{\nu_j}$, $j = 1, \ldots, l$, die mind. ein p_i enthalten



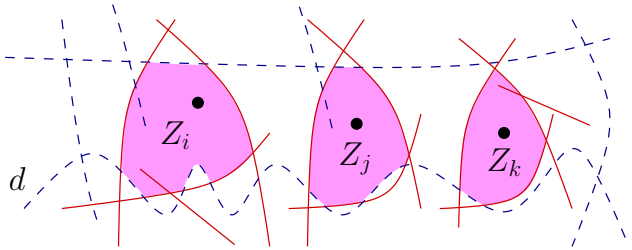
Komplexität der Schnittzellen: Lem. 2.21

Kombinationslemma: Guibas,Sharir,Sifrony 1989 (DSS *Bibel*)

Komplexität der Zellen $Z_1, \ldots, Z_\ell$, $\ell \leq k$:

$$|Z_1| + |Z_2| + \cdots + |Z_\ell| \in O(r + b + k)$$

Nicht trivial:



Zur Analyse der Berechnung verwenden!!!