Karlsruhe Institute of Technology

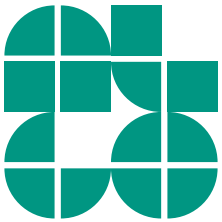
Vorlesung Algorithmische Geometrie

Sichtbarkeitsgraphen

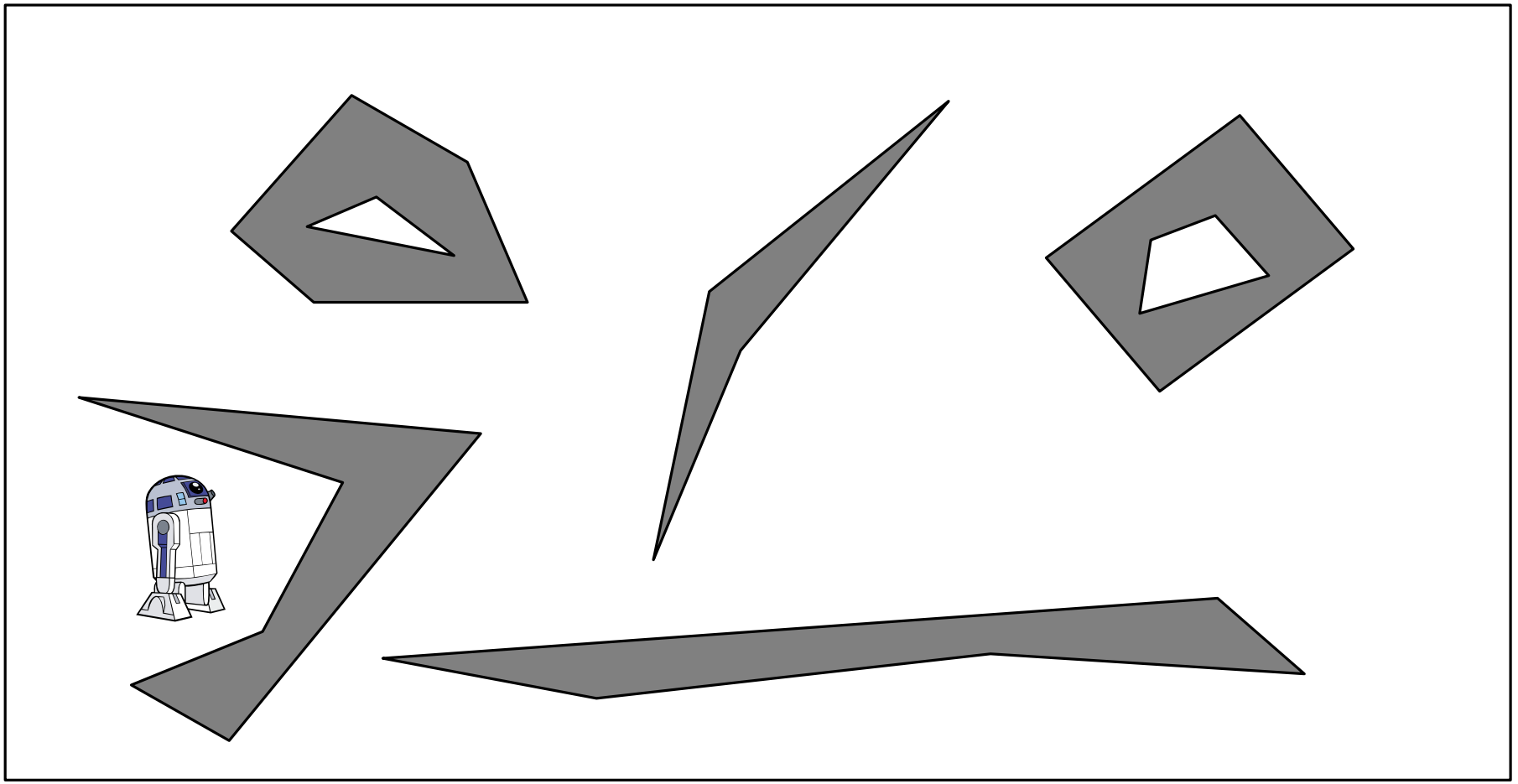
LEHRSTUHL FÜR ALGORITHMIK I · INSTITUT FÜR THEORETISCHE INFORMATIK · FAKULTÄT FÜR INFORMATIK

Martin Nöllenburg

12.07.2011

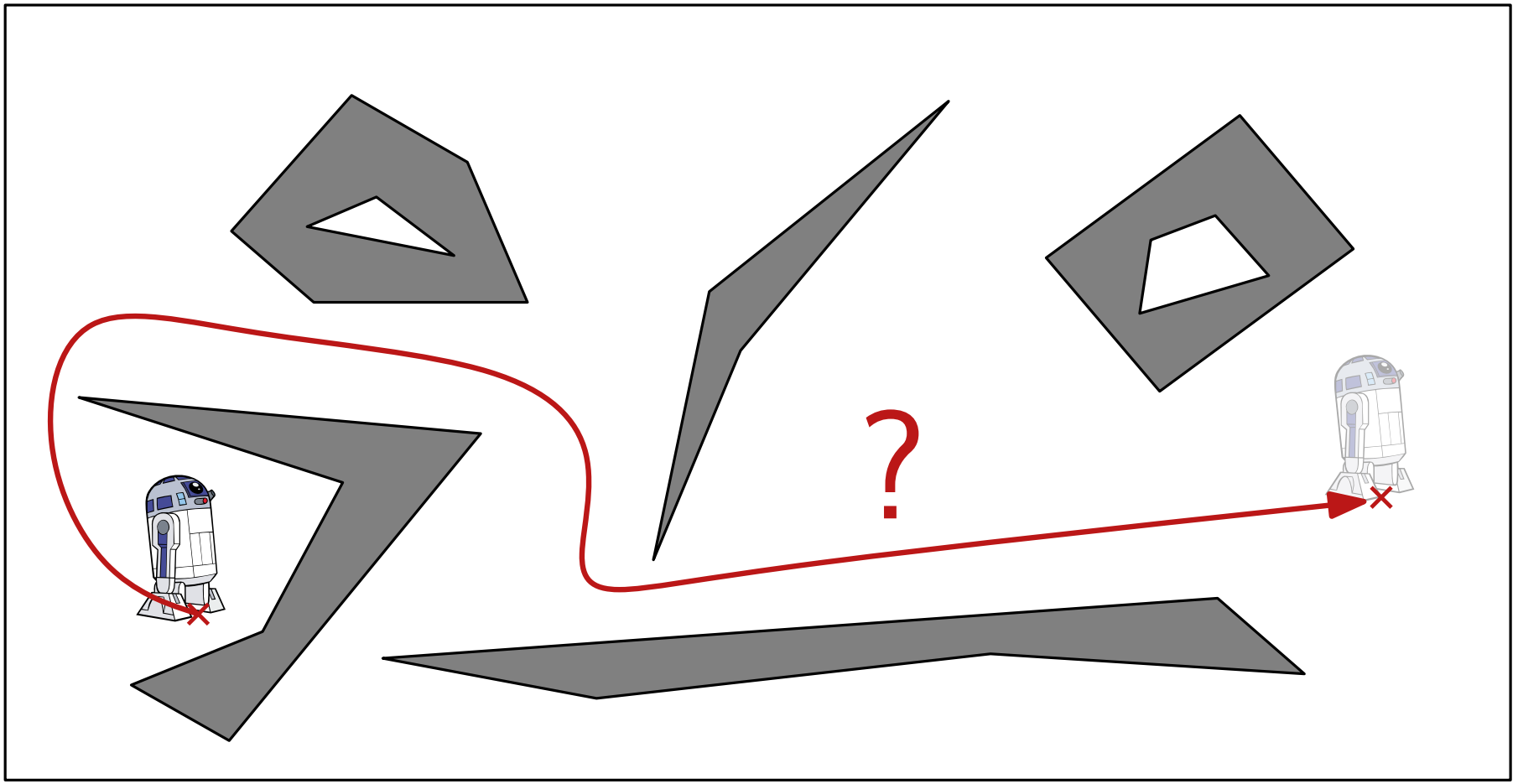


Bewegungsplanung für Roboter



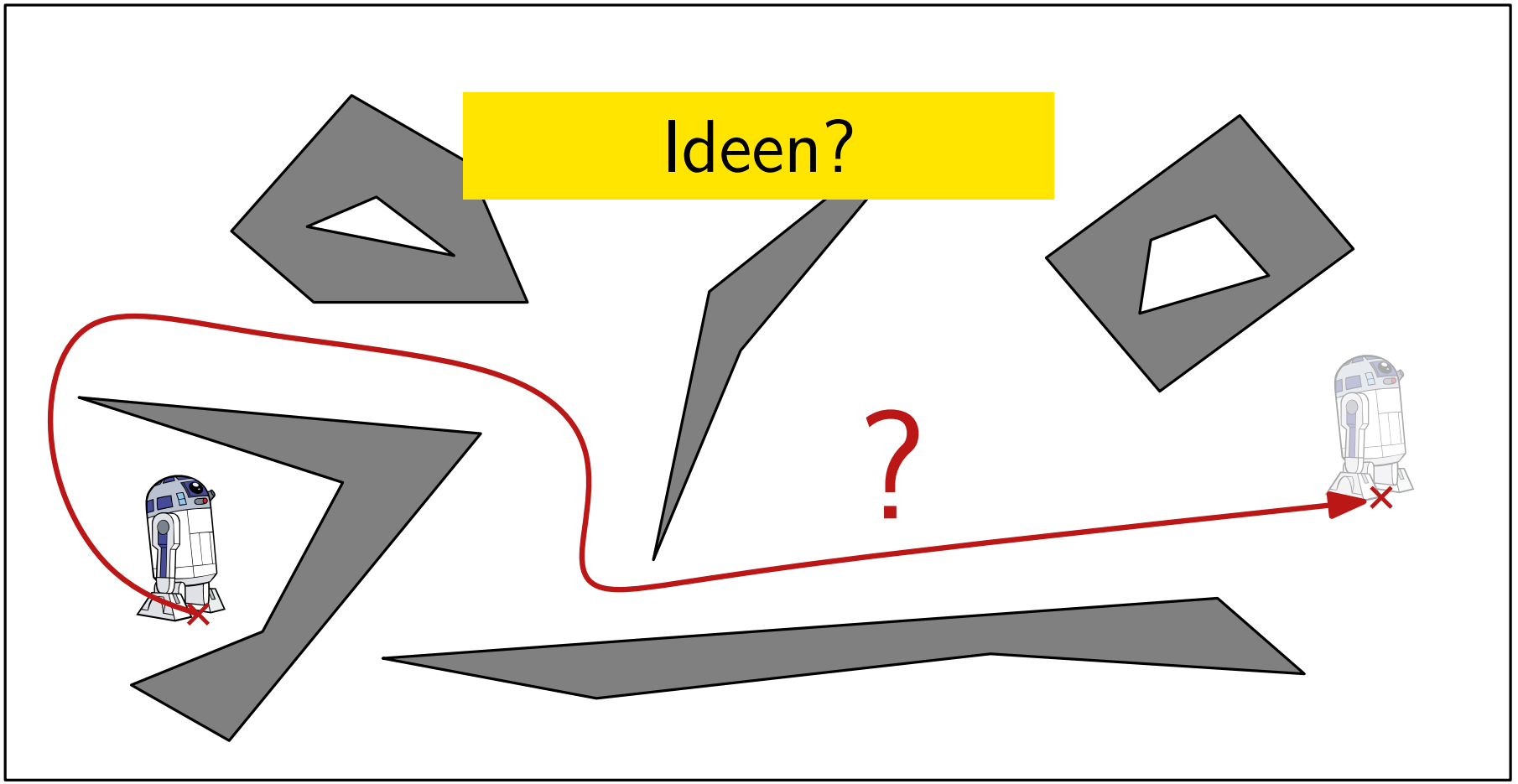
Problem: Gegeben ein (punktförmiger) Roboter an Position p_{start} in einem Gebiet mit polygonalen Hindernissen

Bewegungsplanung für Roboter



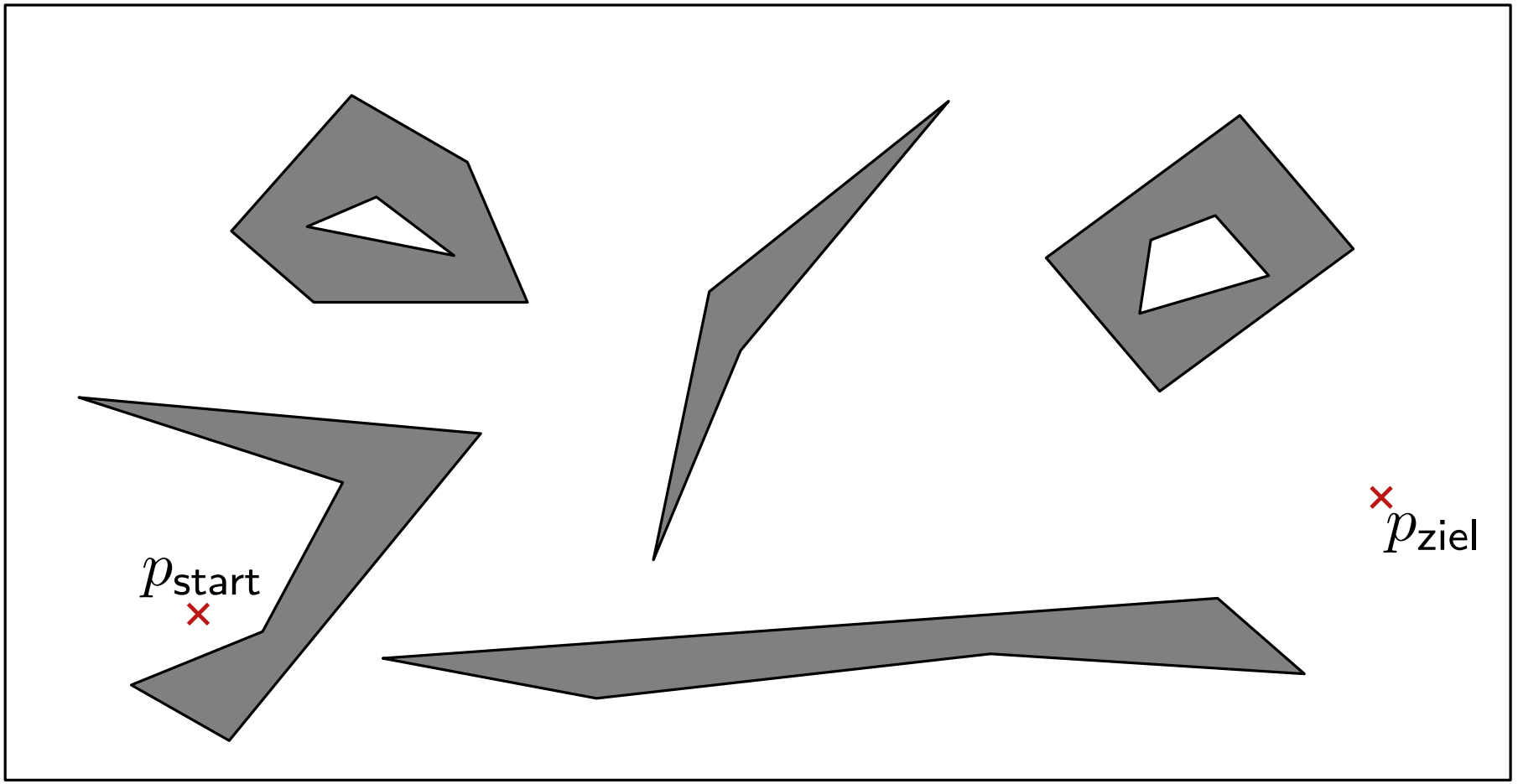
Problem: Gegeben ein (punktförmiger) Roboter an Position p_{start} in einem Gebiet mit polygonalen Hindernissen finde einen möglichst kurzen Weg zum Ziel p_{ziel} um die Hindernisse herum.

Bewegungsplanung für Roboter

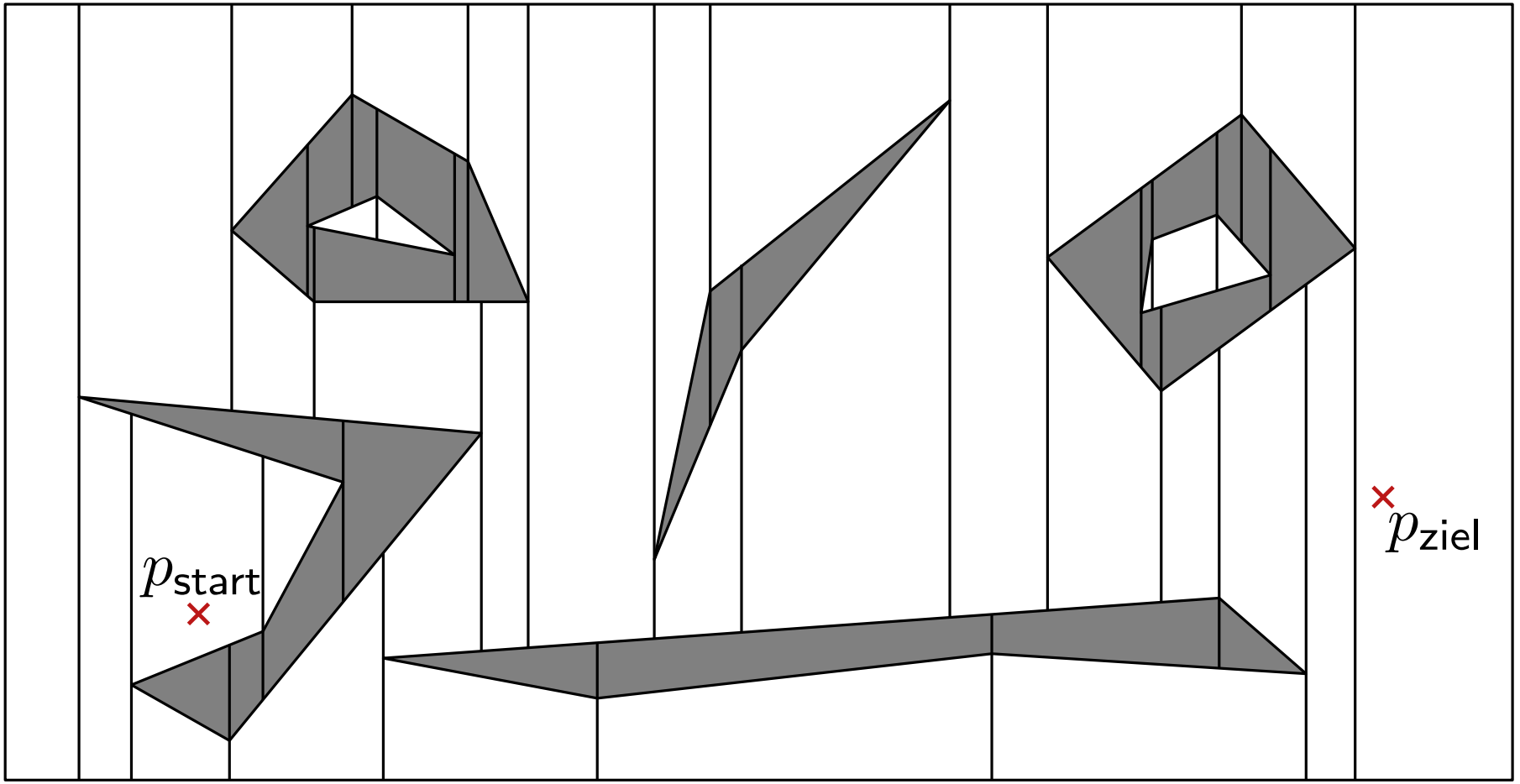


Problem: Gegeben ein (punktförmiger) Roboter an Position p_{start} in einem Gebiet mit polygonalen Hindernissen finde einen möglichst kurzen Weg zum Ziel p_{ziel} um die Hindernisse herum.

Erste Idee: Kürzeste Wege in Graphen

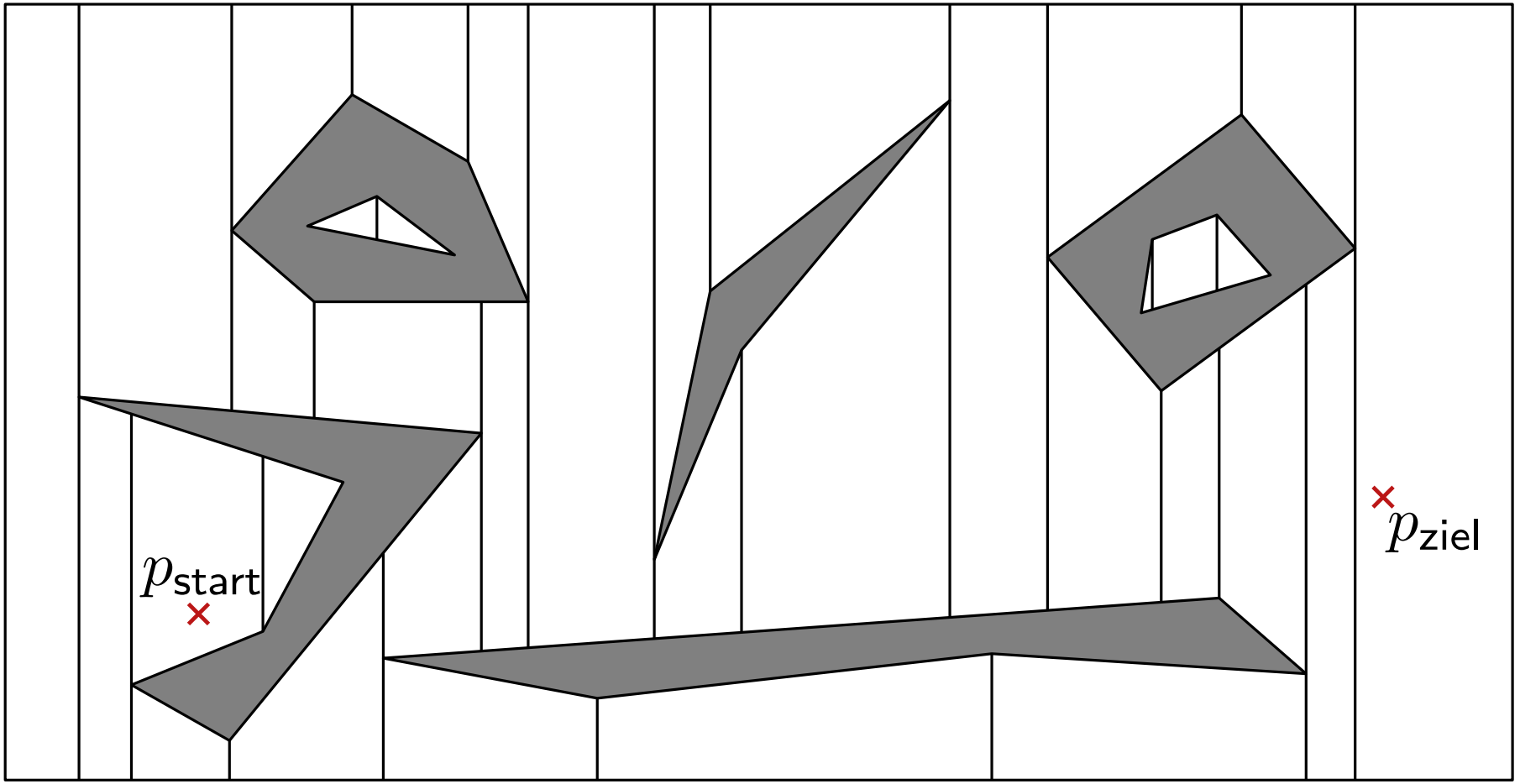


Erste Idee: Kürzeste Wege in Graphen



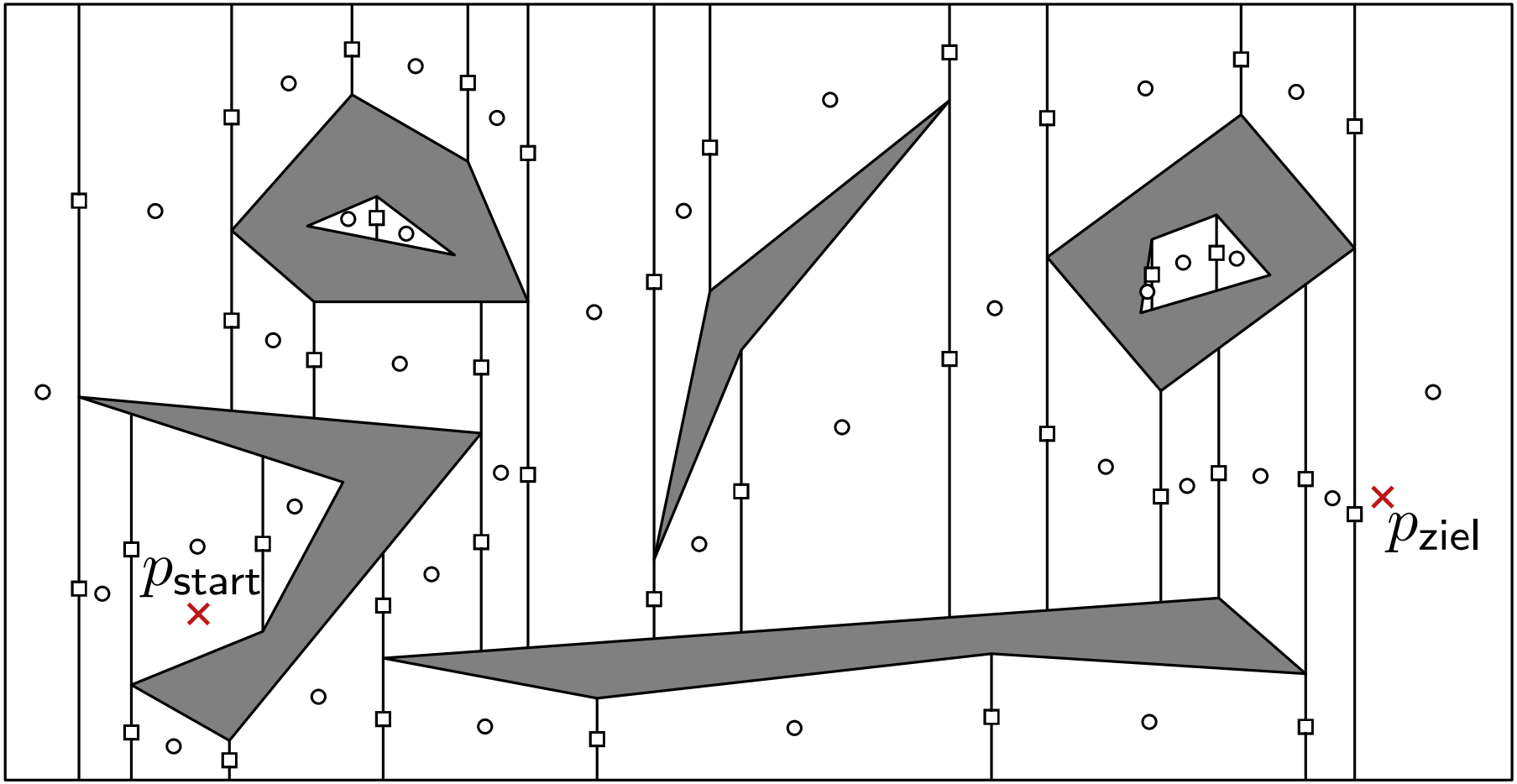
- erstelle Trapezzerlegung

Erste Idee: Kürzeste Wege in Graphen



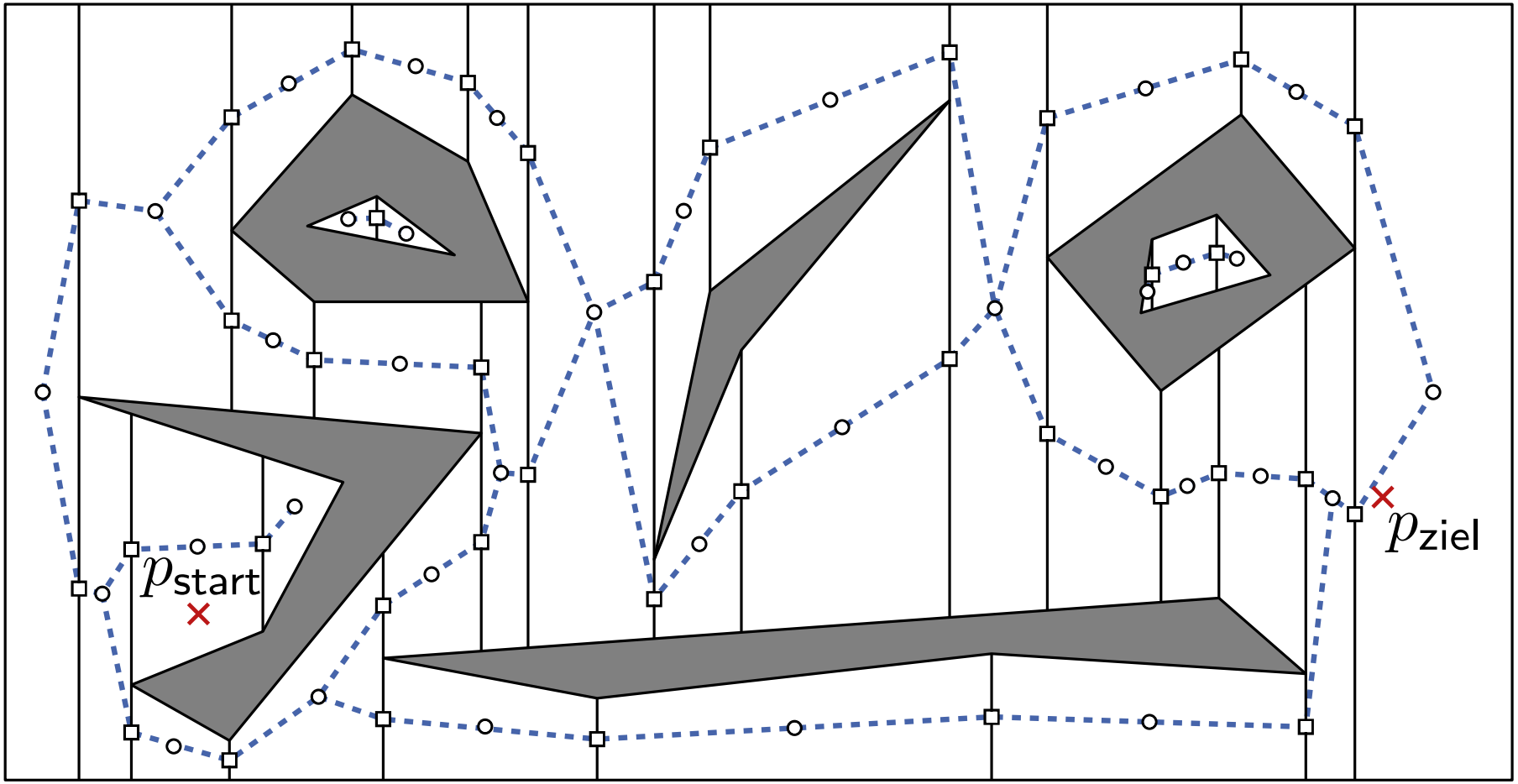
- erstelle Trapezzerlegung
- entferne Segmente in Hindernissen

Erste Idee: Kürzeste Wege in Graphen



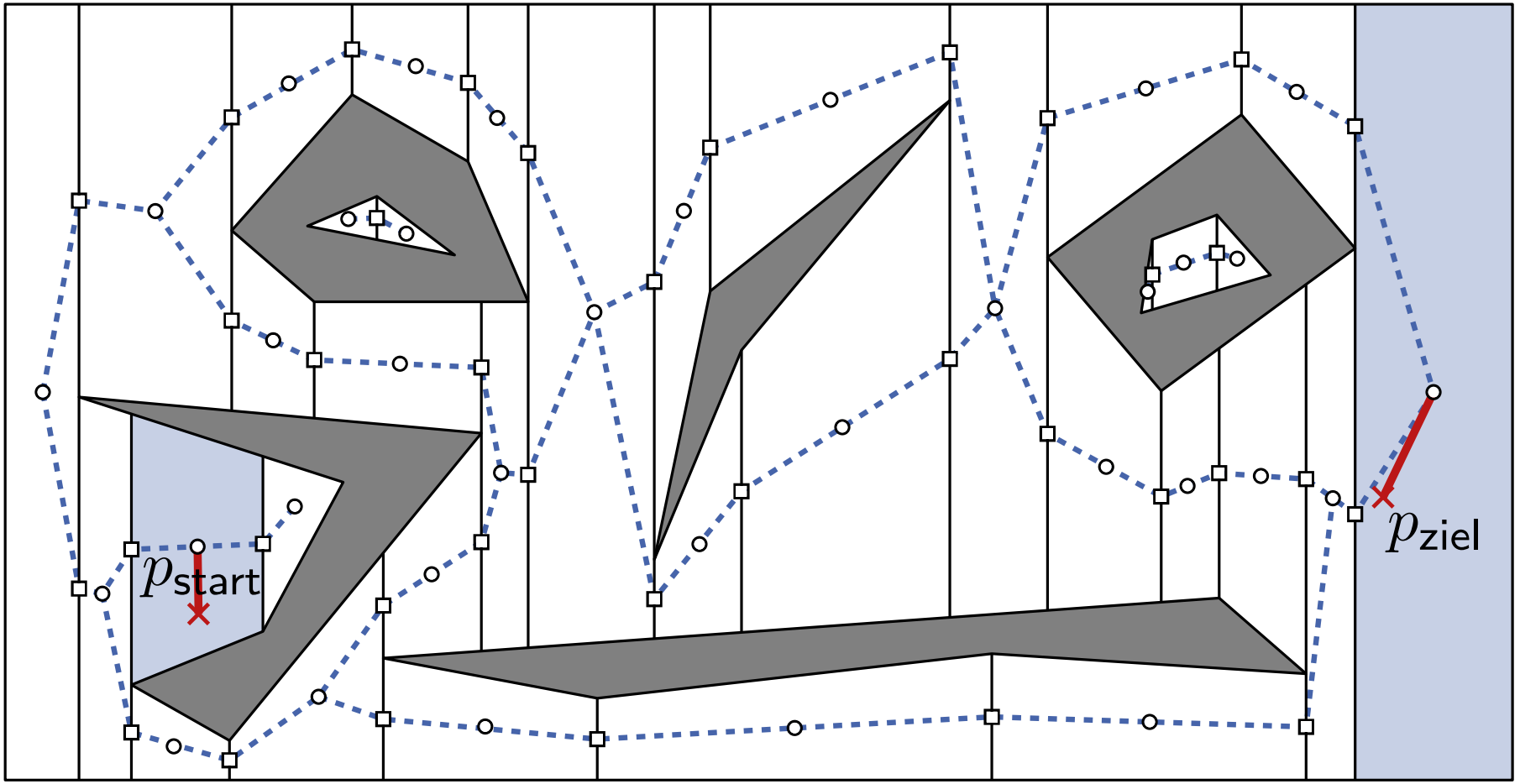
- erstelle Trapezzerlegung
- entferne Segmente in Hindernissen
- Knoten in Trapezen und Vertikalen

Erste Idee: Kürzeste Wege in Graphen



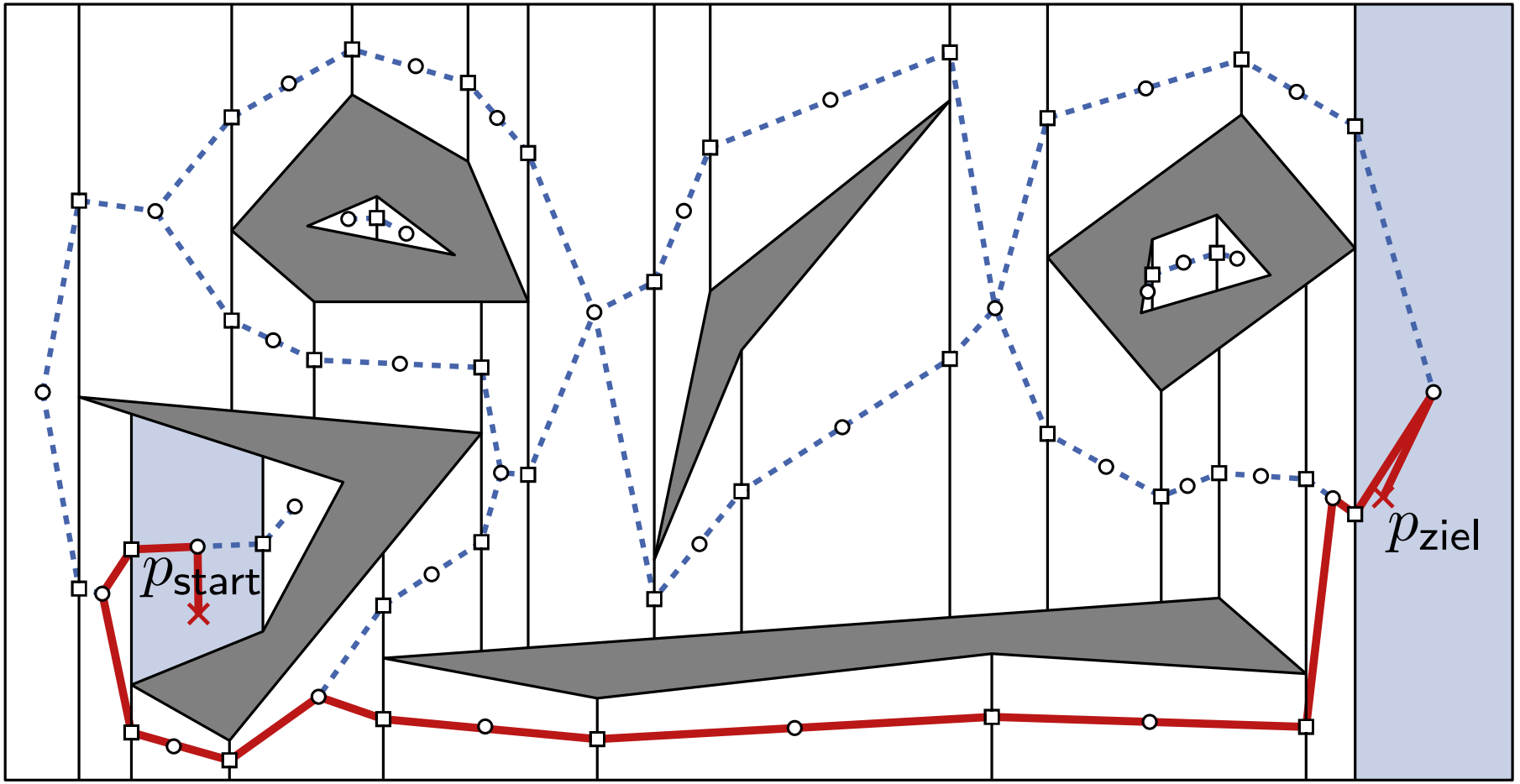
- erstelle Trapezzerlegung
- entferne Segmente in Hindernissen
- Knoten in Trapezen und Vertikalen
- euklidisch gewichteter „Dualgraph“ G mit Viaknoten auf Vertikalen

Erste Idee: Kürzeste Wege in Graphen



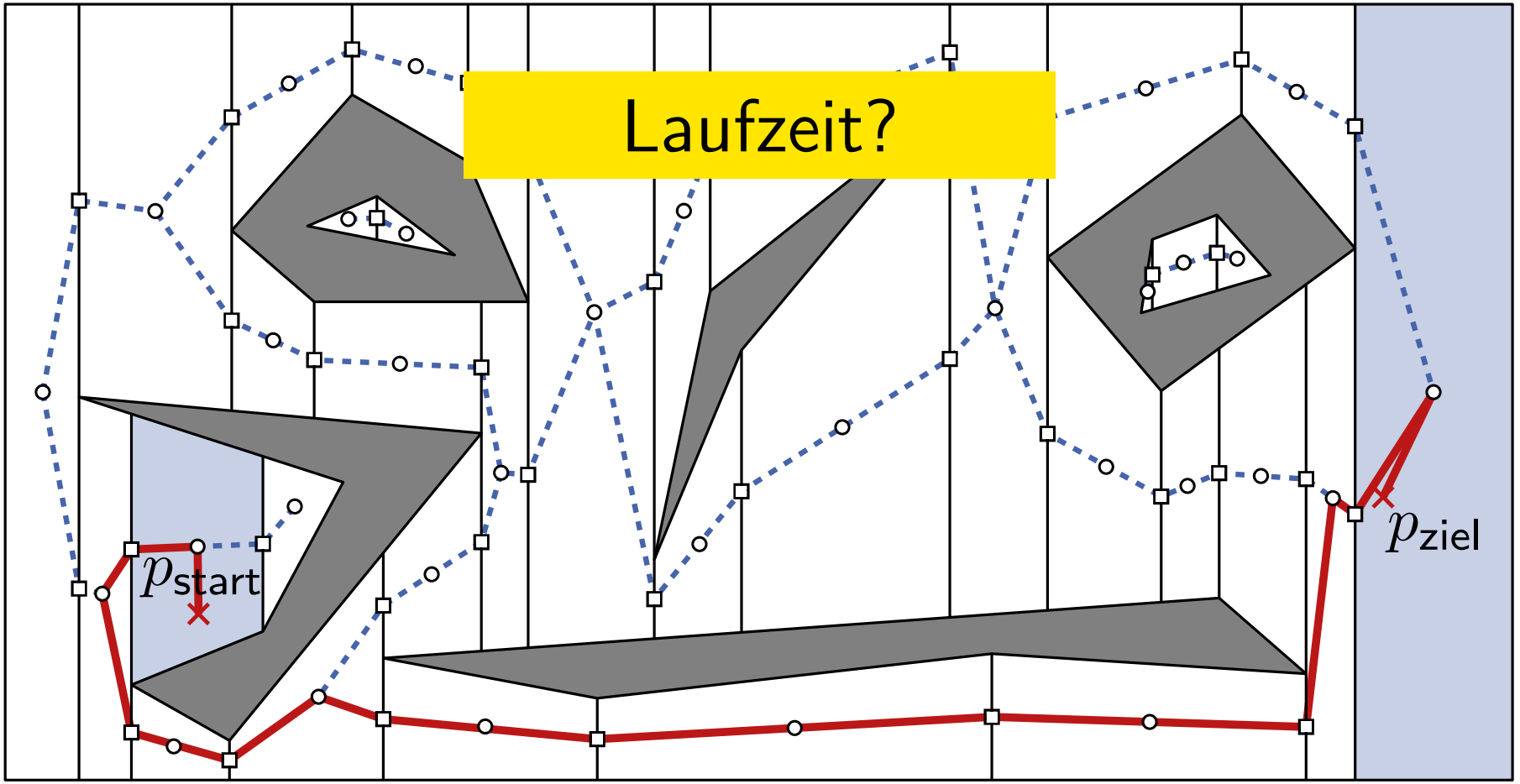
- erstelle Trapezzerlegung
- entferne Segmente in Hindernissen
- Knoten in Trapezen und Vertikalen
- euklidisch gewichteter „Dualgraph“ G mit Viaknoten auf Vertikalen
- Lokalisiere Start und Ziel

Erste Idee: Kürzeste Wege in Graphen



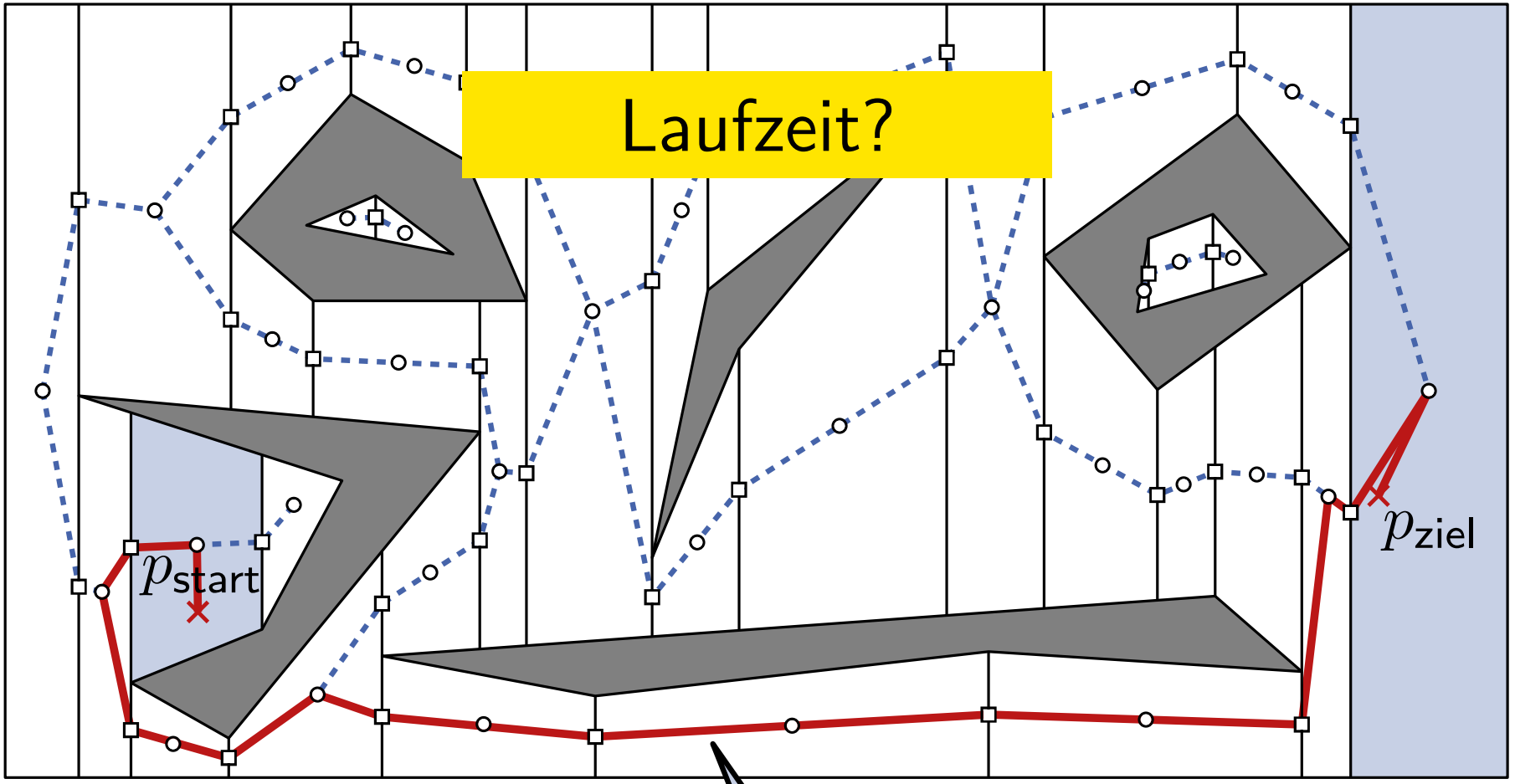
- erstelle Trapezzerlegung
- entferne Segmente in Hindernissen
- Knoten in Trapezen und Vertikalen
- euklidisch gewichteter „Dualgraph“ G mit Viaknoten auf Vertikalen
- Lokalisiere Start und Ziel
- kürzester Weg mit Dijkstra in G

Erste Idee: Kürzeste Wege in Graphen



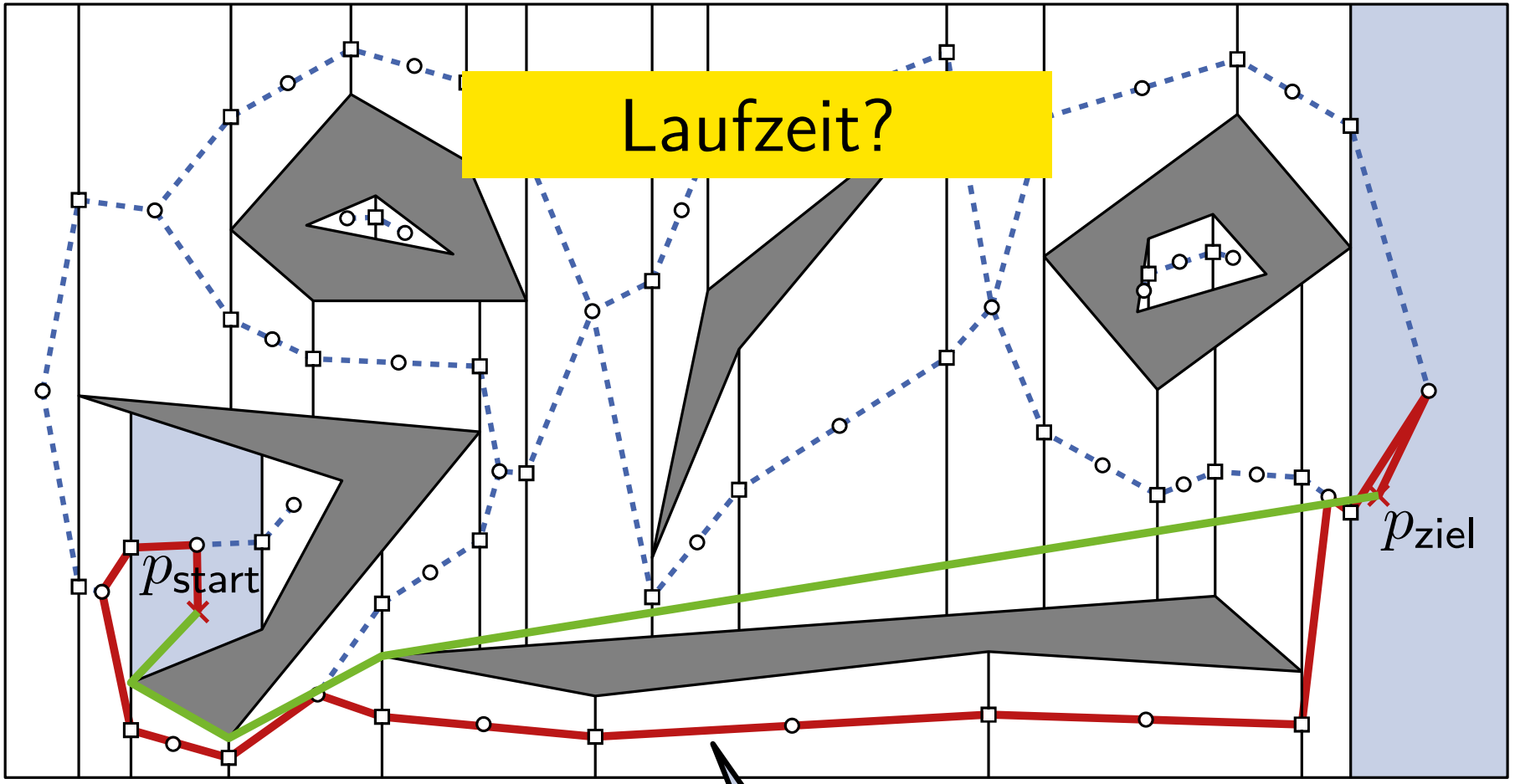
- erstelle Trapezzerlegung
- entferne Segmente in Hindernissen
- Knoten in Trapezen und Vertikalen
- Lokalisiere Start und Ziel
- kürzester Weg mit Dijkstra in G
- euklidisch gewichteter „Dualgraph“ G mit Viaknoten auf Vertikalen

Erste Idee: Kürzeste Wege in Graphen



- erstelle Trapezzerlegung
- entferne Segmente in Hindernissen
- Knoten in Trapezen und Vertikalen
- euklidisch gewichteter „Dualgraph“ G mit Viaknoten auf Vertikalen
- Lokalisiere Start und Ziel
- kürzester Weg mit Dijkstra in G

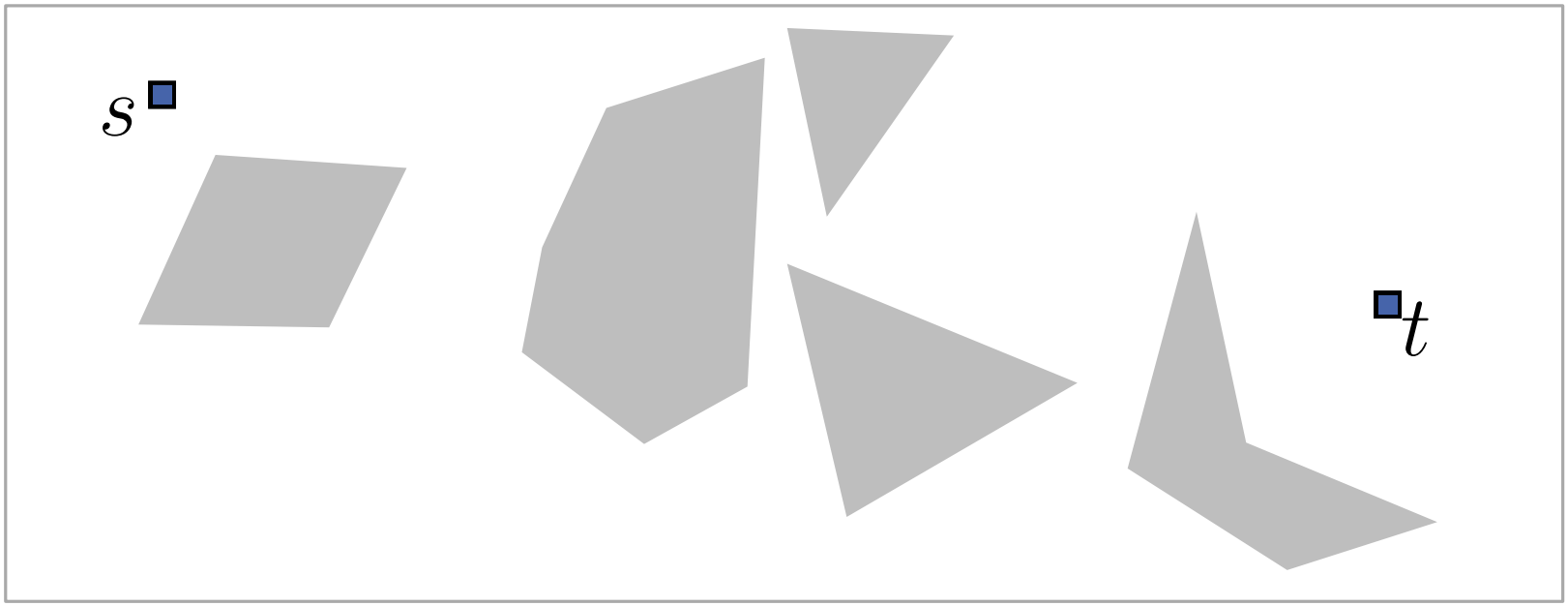
Erste Idee: Kürzeste Wege in Graphen



- erstelle Trapezzerlegung
- entferne Segmente in Hindernissen
- Knoten in Trapezen und Vertikalen
- euklidisch gewichteter „Dualgraph“ G mit Viaknoten auf Vertikalen
- Lokalisiere Start und Ziel
- kürzester Weg mit Dijkstra in G

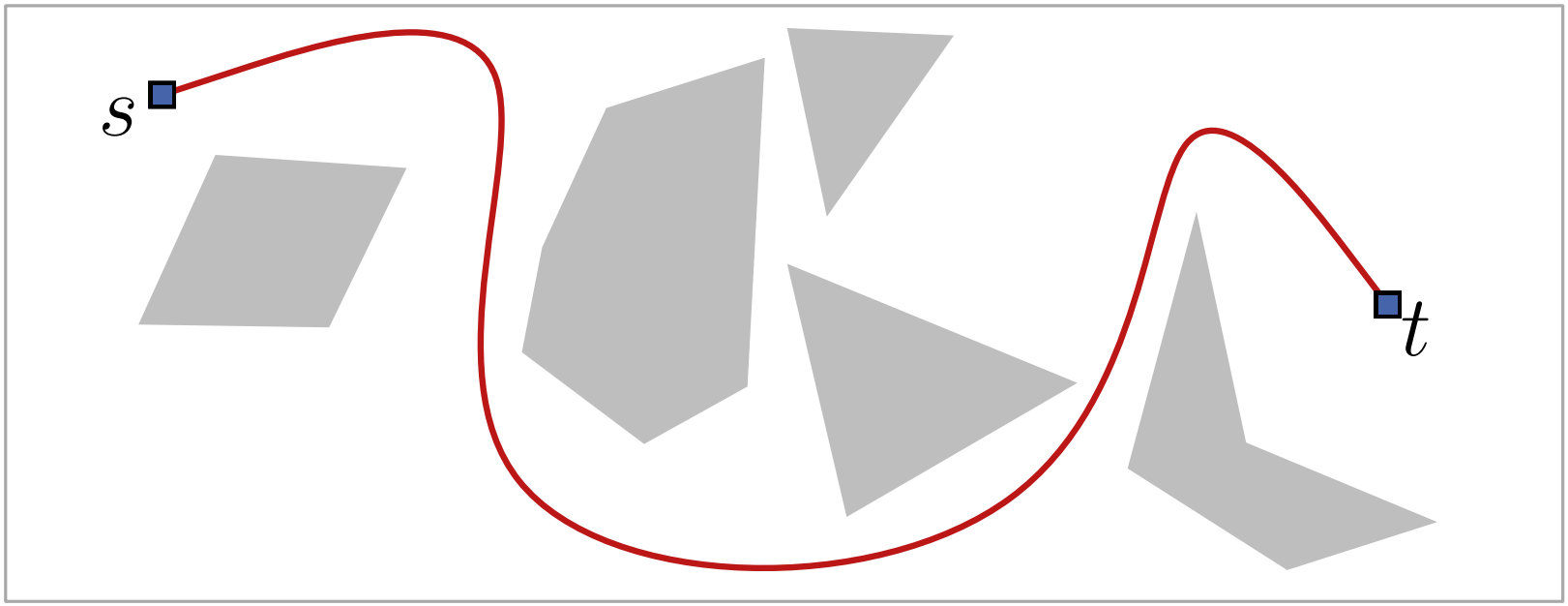
Kürzeste Wege in Polygongebieten

Lemma 1: Für eine Menge S von disjunkten Polygonen in $\mathbb{R}^2$ und zwei Punkte s und t außerhalb S



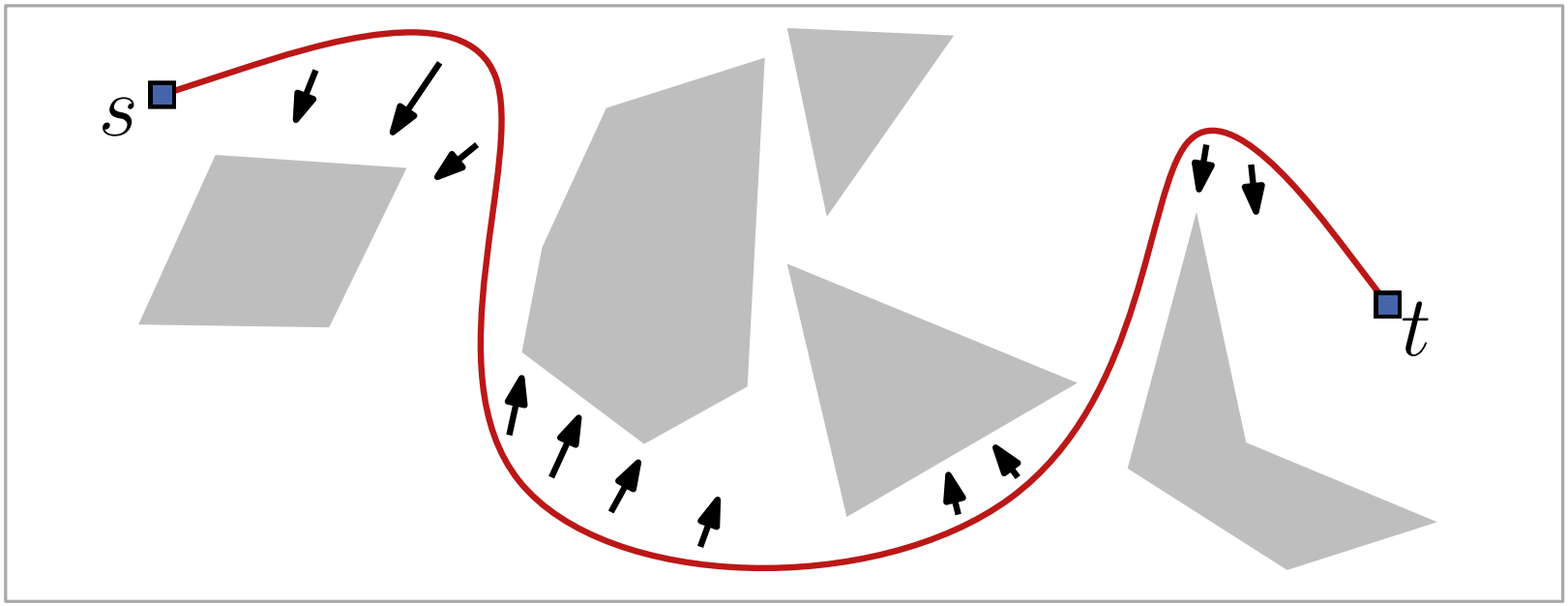
Kürzeste Wege in Polygongebieten

Lemma 1: Für eine Menge S von disjunkten Polygonen in $\mathbb{R}^2$ und zwei Punkte s und t außerhalb S ist jeder kürzeste st-Weg in $\mathbb{R}^2 \setminus \bigcup S$ ein Polygonzug dessen innere Knoten Knoten von S sind.



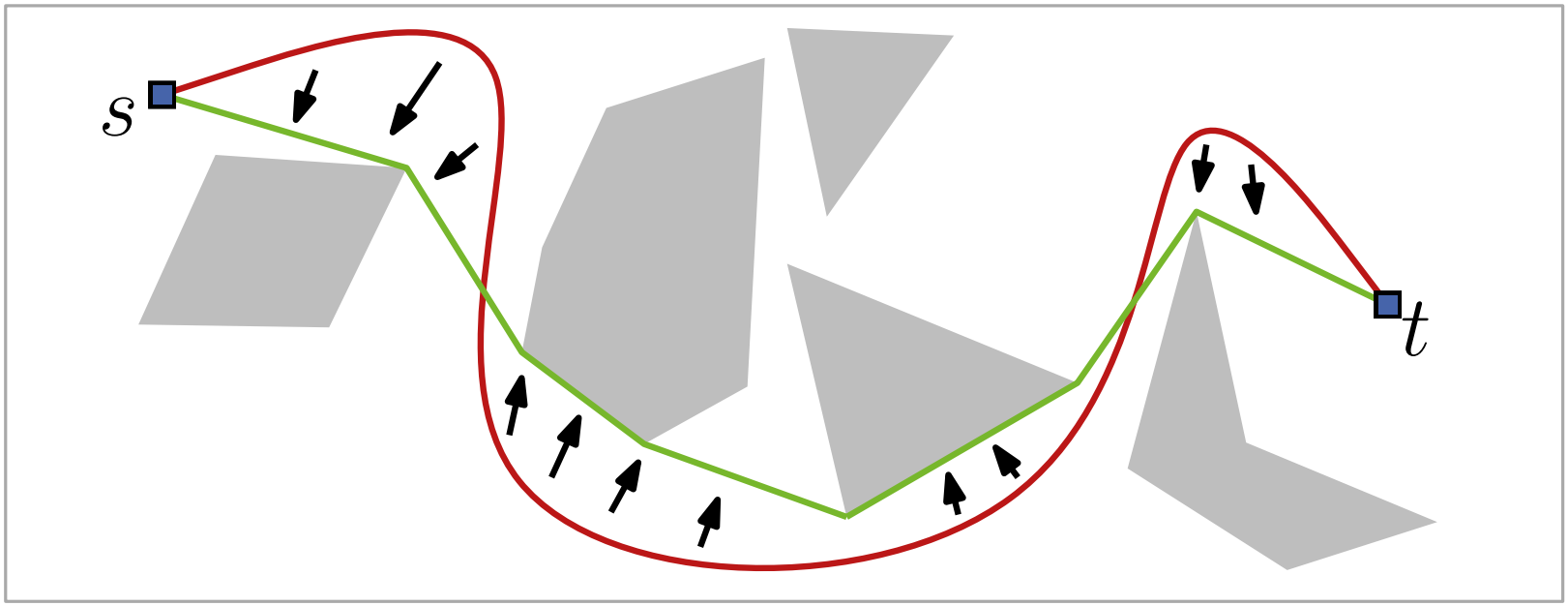
Kürzeste Wege in Polygongebieten

Lemma 1: Für eine Menge S von disjunkten Polygonen in $\mathbb{R}^2$ und zwei Punkte s und t außerhalb S ist jeder kürzeste st-Weg in $\mathbb{R}^2 \setminus \bigcup S$ ein Polygonzug dessen innere Knoten Knoten von S sind.



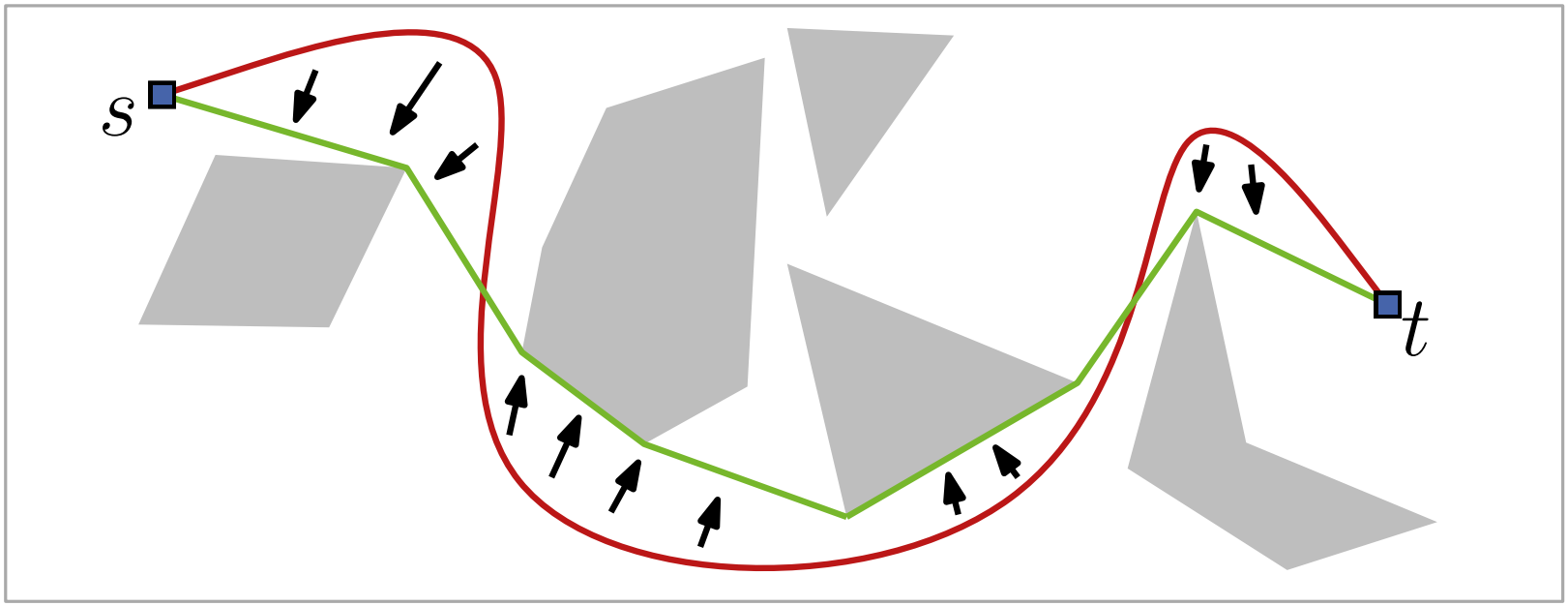
Kürzeste Wege in Polygongebieten

Lemma 1: Für eine Menge S von disjunkten Polygonen in $\mathbb{R}^2$ und zwei Punkte s und t außerhalb S ist jeder kürzeste st-Weg in $\mathbb{R}^2 \setminus \bigcup S$ ein Polygonzug dessen innere Knoten Knoten von S sind.



Kürzeste Wege in Polygongebieten

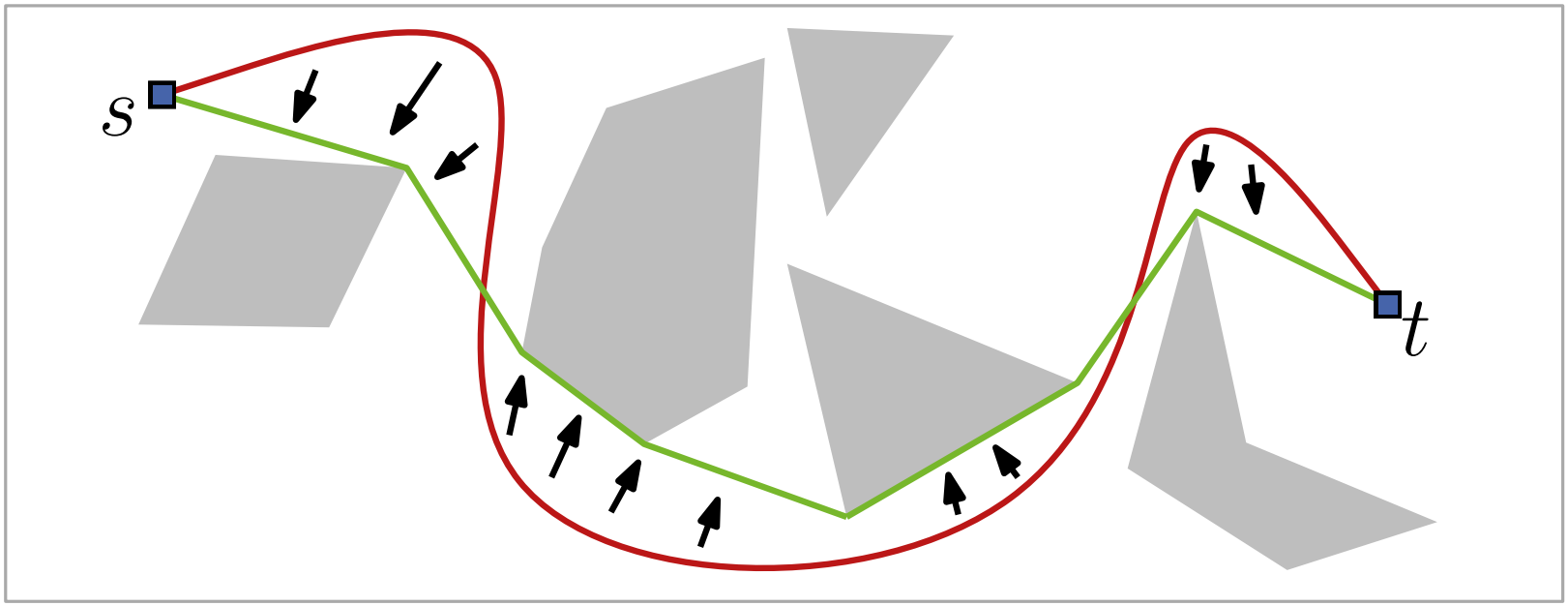
Lemma 1: Für eine Menge S von disjunkten Polygonen in $\mathbb{R}^2$ und zwei Punkte s und t außerhalb S ist jeder kürzeste st-Weg in $\mathbb{R}^2 \setminus \bigcup S$ ein Polygonzug dessen innere Knoten Knoten von S sind.



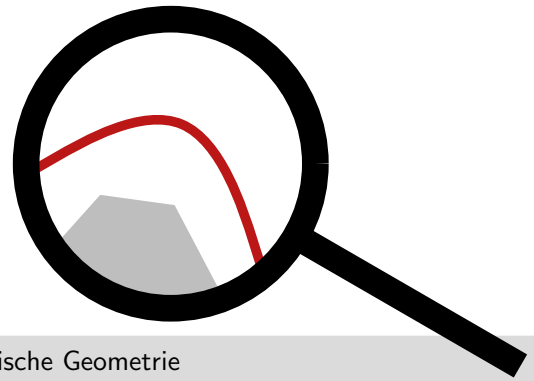
Beweisskizze:

Kürzeste Wege in Polygongebieten

Lemma 1: Für eine Menge S von disjunkten Polygonen in $\mathbb{R}^2$ und zwei Punkte s und t außerhalb S ist jeder kürzeste st-Weg in $\mathbb{R}^2 \setminus \bigcup S$ ein Polygonzug dessen innere Knoten Knoten von S sind.

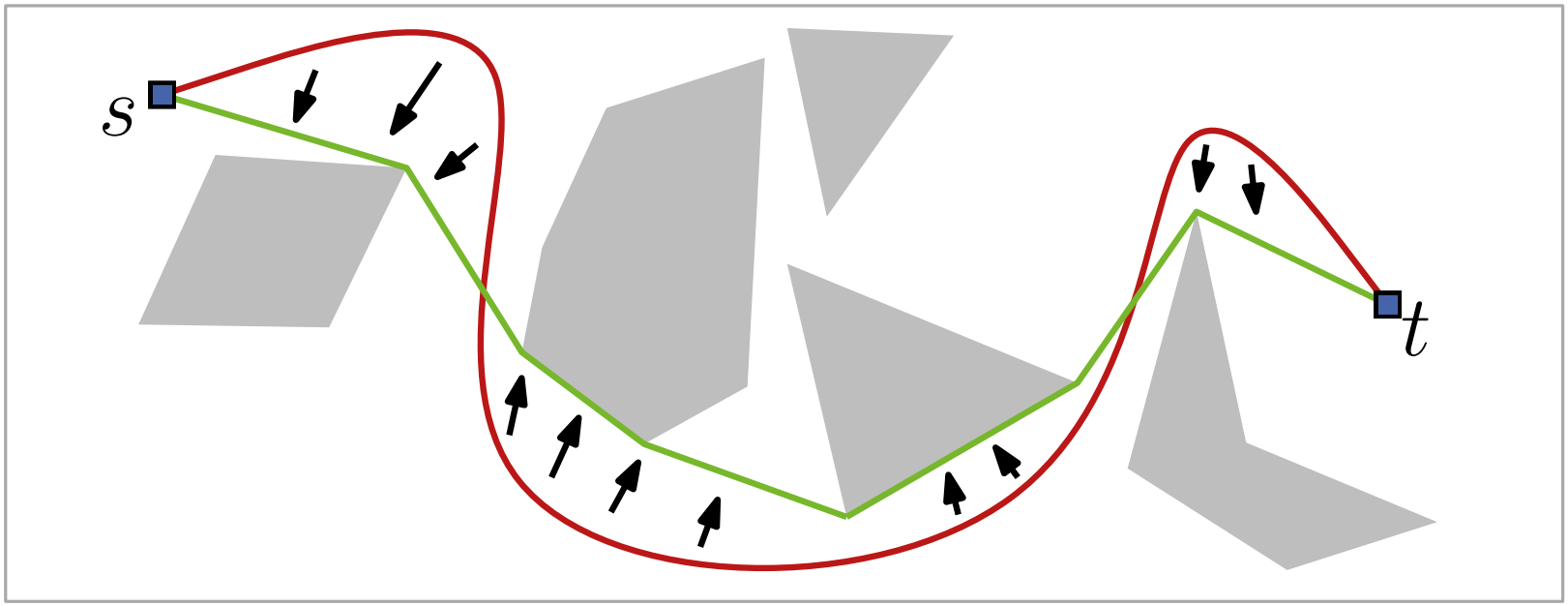


Beweisskizze:

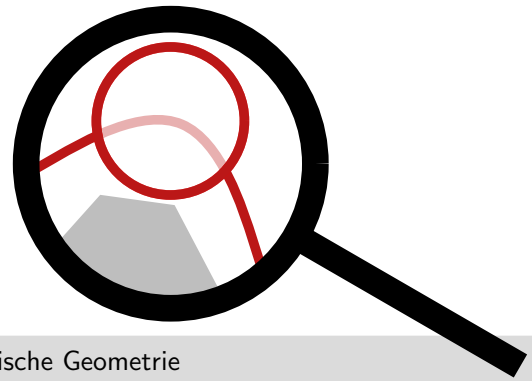


Kürzeste Wege in Polygongebieten

Lemma 1: Für eine Menge S von disjunkten Polygonen in $\mathbb{R}^2$ und zwei Punkte s und t außerhalb S ist jeder kürzeste st-Weg in $\mathbb{R}^2 \setminus \bigcup S$ ein Polygonzug dessen innere Knoten Knoten von S sind.

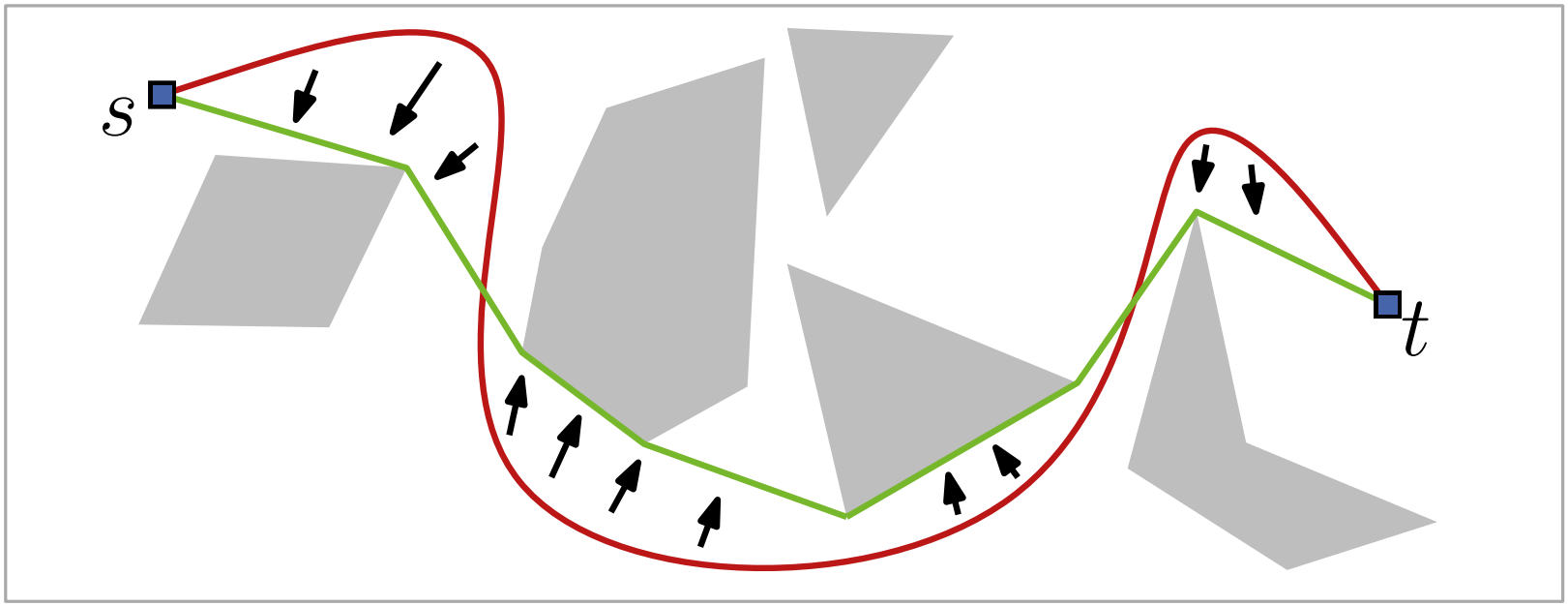


Beweisskizze:

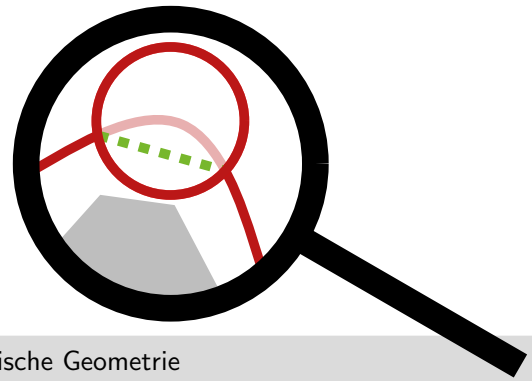


Kürzeste Wege in Polygongebieten

Lemma 1: Für eine Menge S von disjunkten Polygonen in $\mathbb{R}^2$ und zwei Punkte s und t außerhalb S ist jeder kürzeste st-Weg in $\mathbb{R}^2 \setminus \bigcup S$ ein Polygonzug dessen innere Knoten Knoten von S sind.

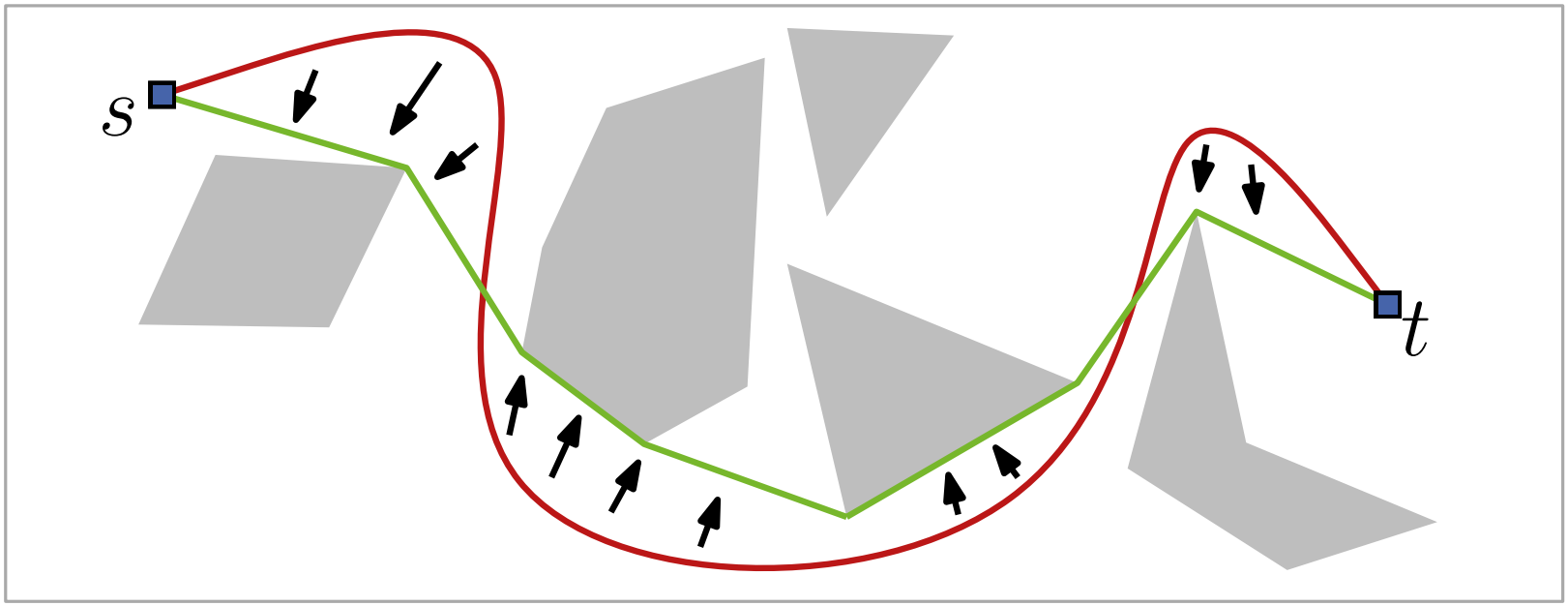


Beweisskizze:

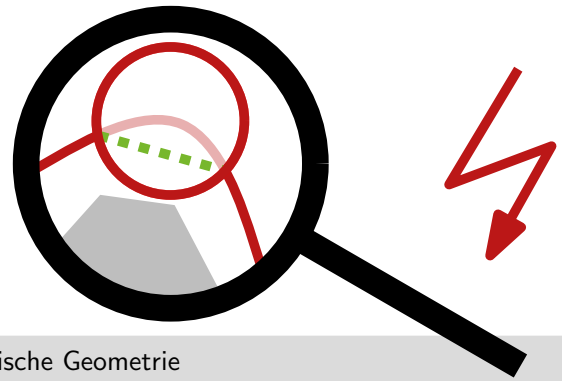


Kürzeste Wege in Polygongebieten

Lemma 1: Für eine Menge S von disjunkten Polygonen in $\mathbb{R}^2$ und zwei Punkte s und t außerhalb S ist jeder kürzeste st-Weg in $\mathbb{R}^2 \setminus \bigcup S$ ein Polygonzug dessen innere Knoten Knoten von S sind.

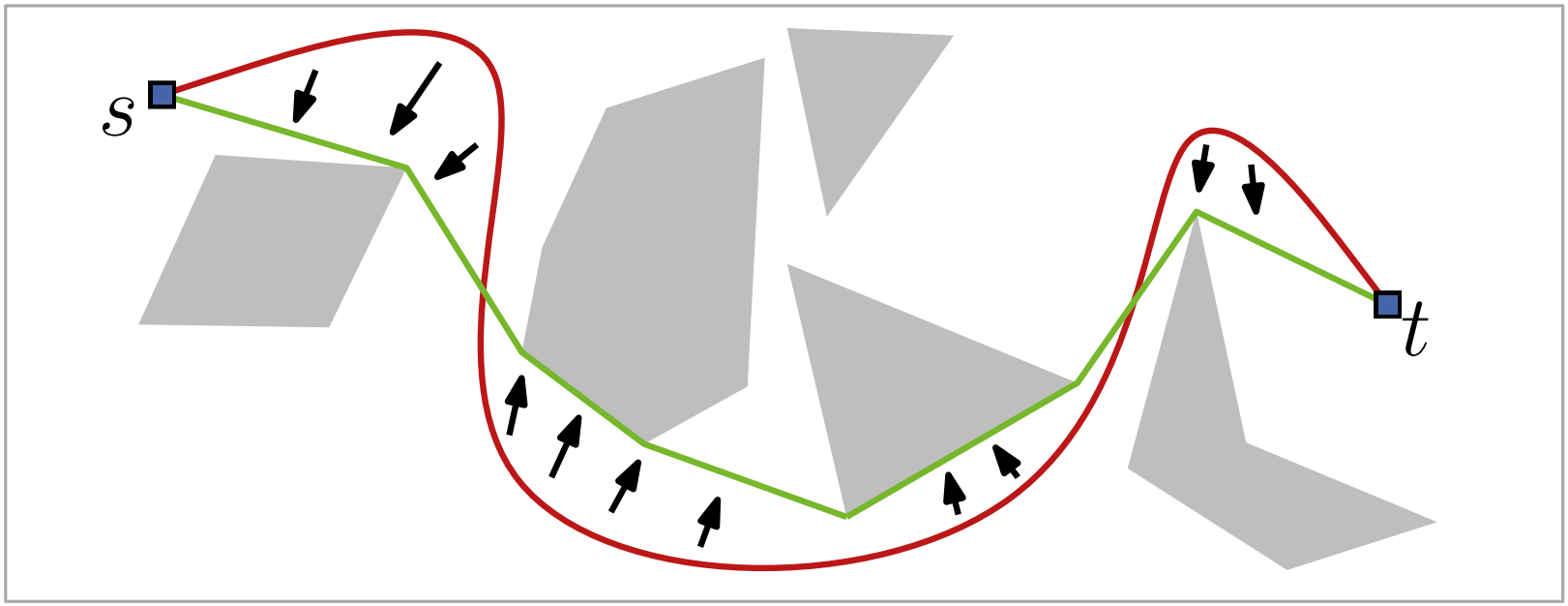


Beweisskizze:

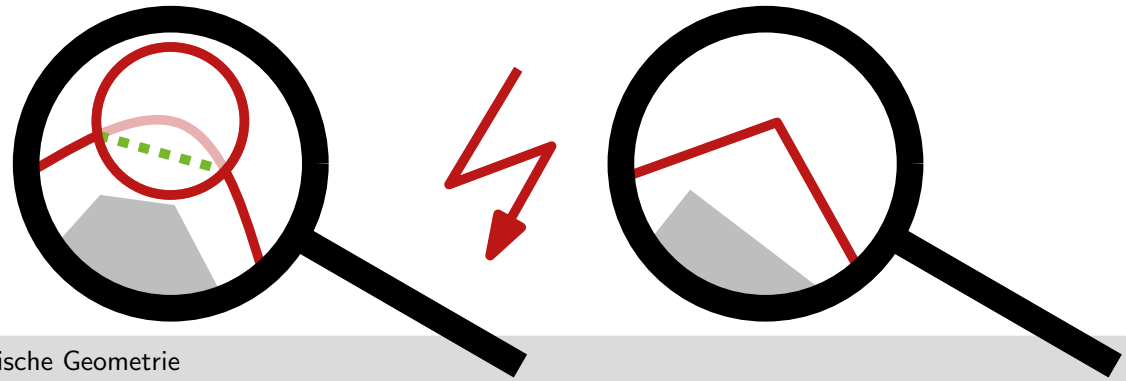


Kürzeste Wege in Polygongebieten

Lemma 1: Für eine Menge S von disjunkten Polygonen in $\mathbb{R}^2$ und zwei Punkte s und t außerhalb S ist jeder kürzeste st-Weg in $\mathbb{R}^2 \setminus \bigcup S$ ein Polygonzug dessen innere Knoten Knoten von S sind.

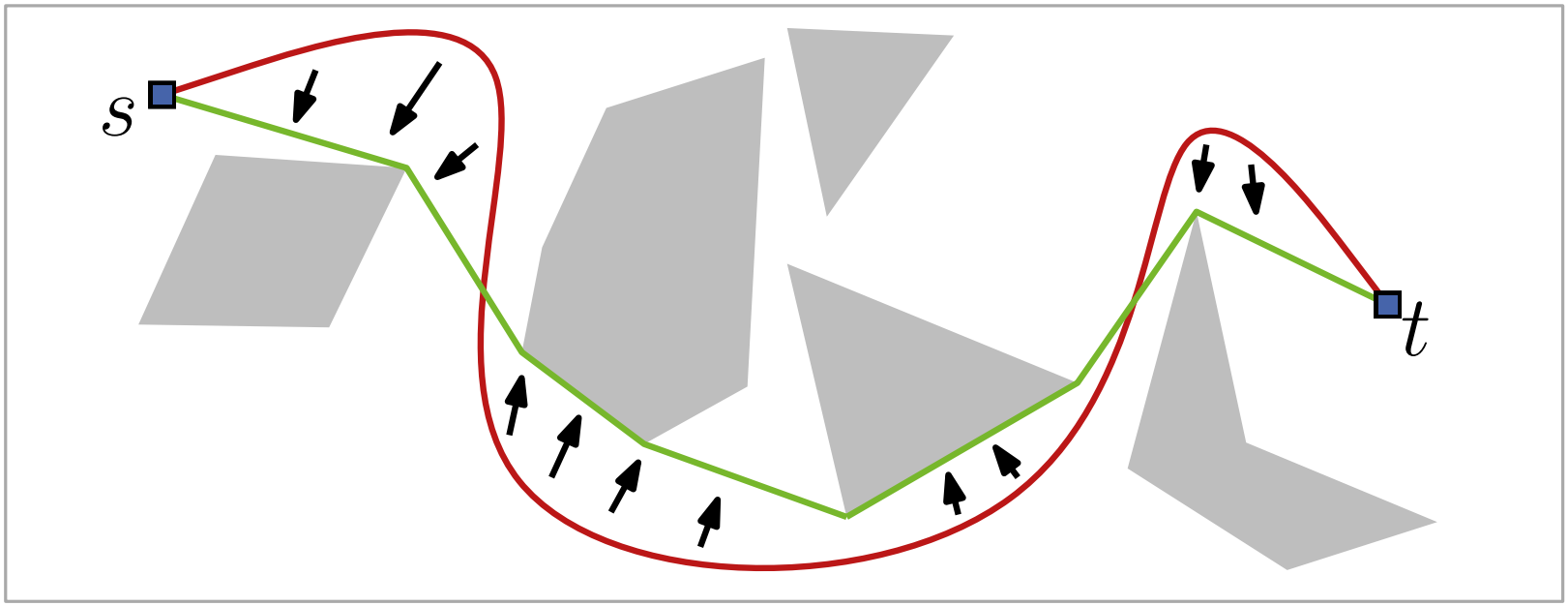


Beweisskizze:

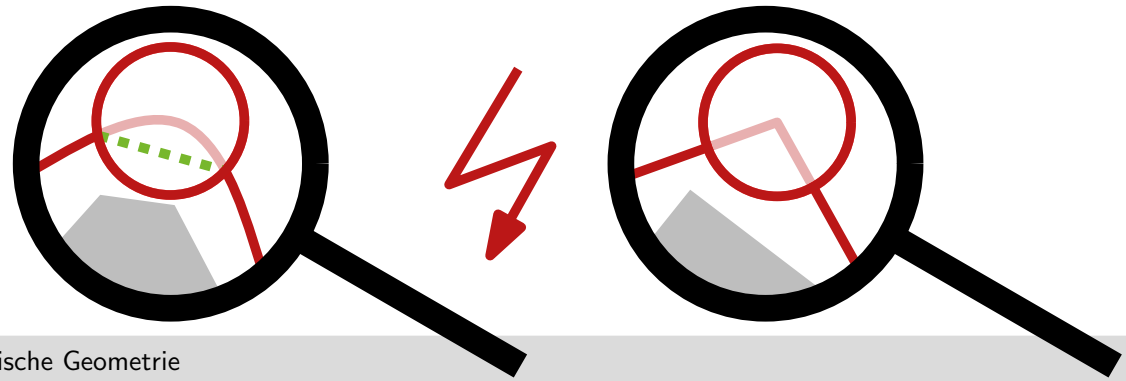


Kürzeste Wege in Polygongebieten

Lemma 1: Für eine Menge S von disjunkten Polygonen in $\mathbb{R}^2$ und zwei Punkte s und t außerhalb S ist jeder kürzeste st-Weg in $\mathbb{R}^2 \setminus \bigcup S$ ein Polygonzug dessen innere Knoten Knoten von S sind.

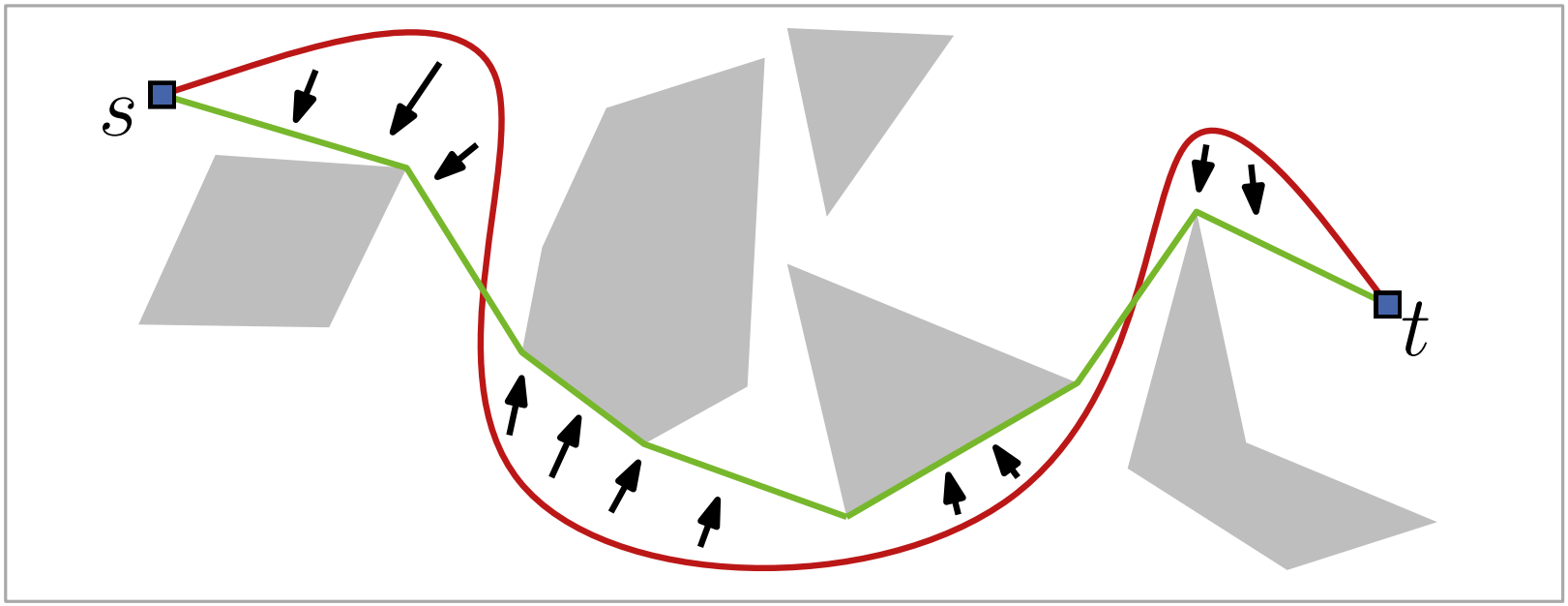


Beweisskizze:

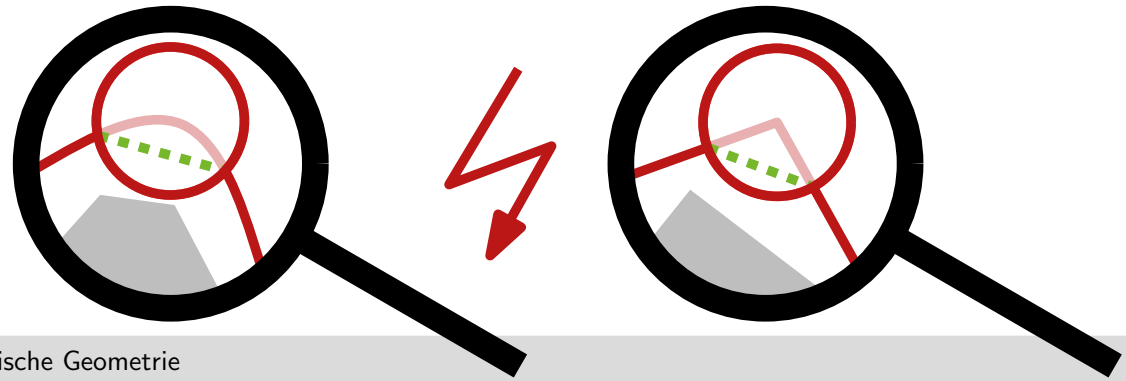


Kürzeste Wege in Polygongebieten

Lemma 1: Für eine Menge S von disjunkten Polygonen in $\mathbb{R}^2$ und zwei Punkte s und t außerhalb S ist jeder kürzeste st-Weg in $\mathbb{R}^2 \setminus \bigcup S$ ein Polygonzug dessen innere Knoten Knoten von S sind.

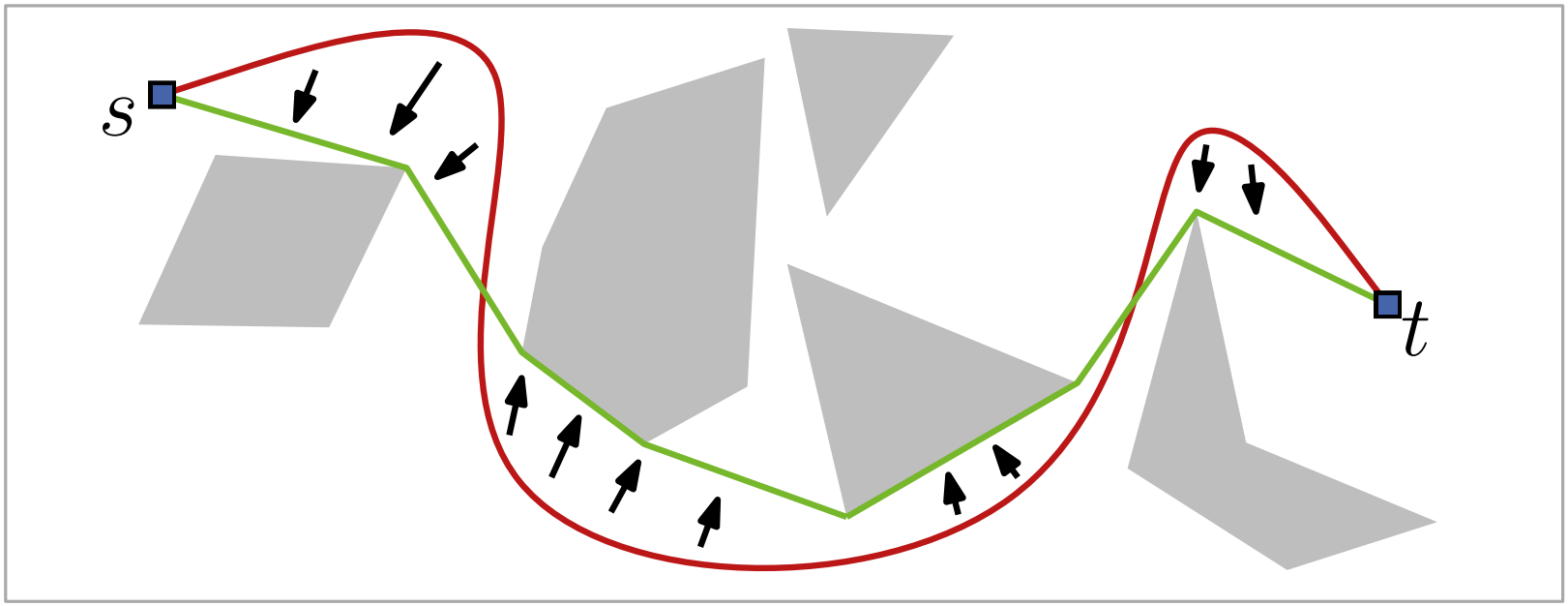


Beweisskizze:

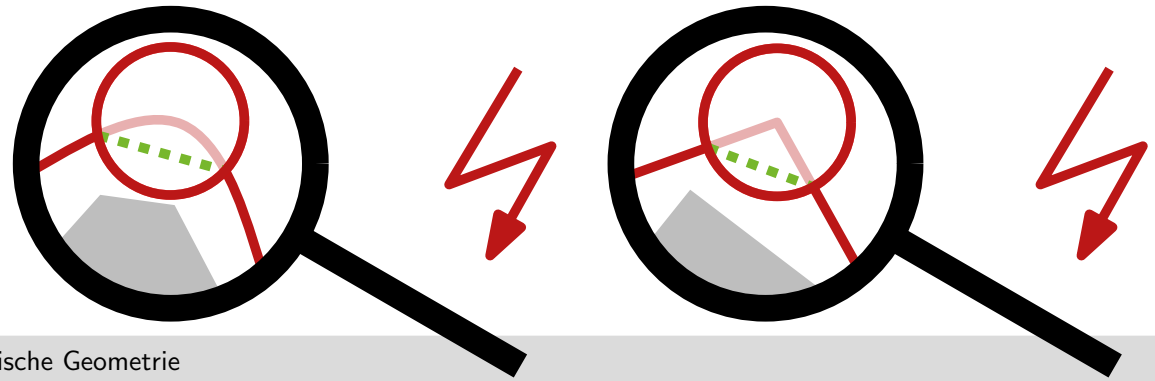


Kürzeste Wege in Polygongebieten

Lemma 1: Für eine Menge S von disjunkten Polygonen in $\mathbb{R}^2$ und zwei Punkte s und t außerhalb S ist jeder kürzeste st-Weg in $\mathbb{R}^2 \setminus \bigcup S$ ein Polygonzug dessen innere Knoten Knoten von S sind.



Beweisskizze:

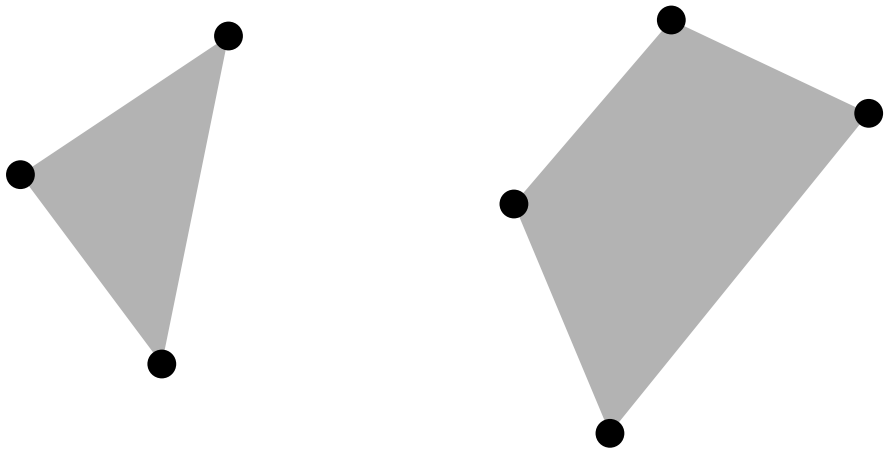


Sichtbarkeitsgraph

Gegeben sei eine Menge S disjunkter offener Polygone...

Sichtbarkeitsgraph

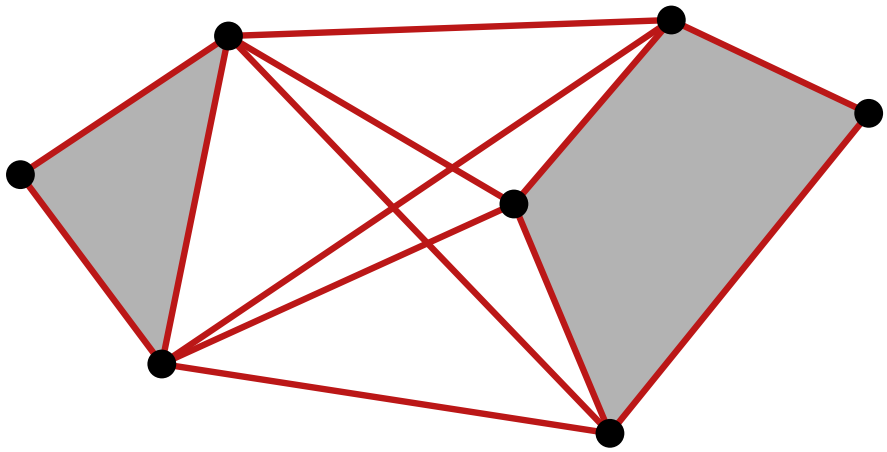
Gegeben sei eine Menge S disjunkter offener Polygone...



...mit Knotenmenge $V(S)$.

Sichtbarkeitsgraph

Gegeben sei eine Menge S disjunkter offener Polygone...

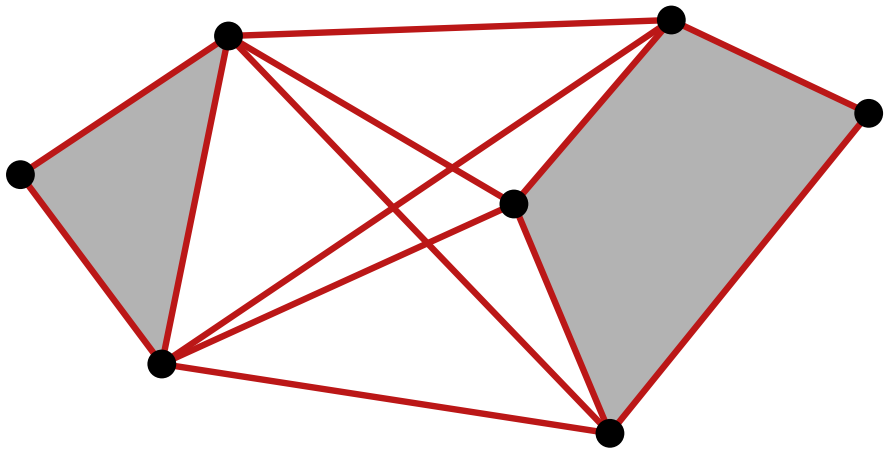


...mit Knotenmenge $V(S)$.

Def.: Dann ist $G_{\text{vis}}(S) = (V(S), E_{\text{vis}}(S))$ der **Sichtbarkeitsgraph** von S mit $E_{\text{vis}}(S) = \{uv \mid u, v \in V(S) \text{ und } u \textit{ sieht } v\}$ und $w(uv) = |uv|$.

Sichtbarkeitsgraph

Gegeben sei eine Menge S disjunkter offener Polygone...

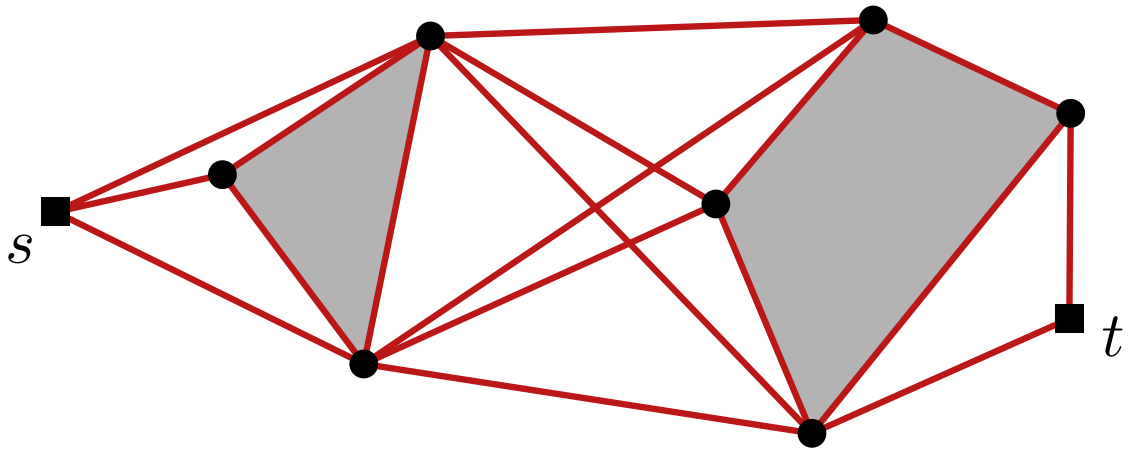


...mit Knotenmenge $V(S)$.

Def.: Dann ist $G_{\text{vis}}(S) = (V(S), E_{\text{vis}}(S))$ der **Sichtbarkeitsgraph** von S mit $E_{\text{vis}}(S) = \{uv \mid u, v \in V(S) \text{ und } u \textit{ sieht } v\}$ und $w(uv) = |uv|$.
Dabei gilt u **sieht** $v :\Leftrightarrow \overline{uv} \subset \mathcal{C}_{\text{free}} = \mathbb{R}^2 \setminus \bigcup S$

Sichtbarkeitsgraph

Gegeben sei eine Menge S disjunkter offener Polygone...



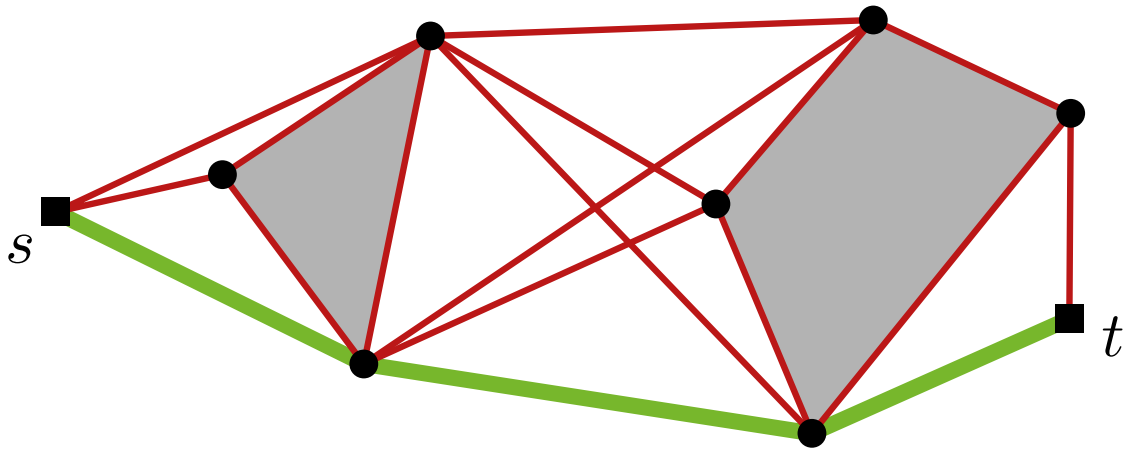
...mit Knotenmenge $V(S)$.

Def.: Dann ist $G_{\text{vis}}(S) = (V(S), E_{\text{vis}}(S))$ der **Sichtbarkeitsgraph** von S mit $E_{\text{vis}}(S) = \{uv \mid u, v \in V(S) \text{ und } u \textit{ sieht } v\}$ und $w(uv) = |uv|$.
Dabei gilt u **sieht** $v :\Leftrightarrow \overline{uv} \subset \mathcal{C}_{\text{free}} = \mathbb{R}^2 \setminus \bigcup S$

Definiere $S^\star = S \cup \{s, t\}$ und $G_{\text{vis}}(S^\star)$ analog.

Sichtbarkeitsgraph

Gegeben sei eine Menge S disjunkter offener Polygone...



...mit Knotenmenge $V(S)$.

Def.: Dann ist $G_{\text{vis}}(S) = (V(S), E_{\text{vis}}(S))$ der **Sichtbarkeitsgraph** von S mit $E_{\text{vis}}(S) = \{uv \mid u, v \in V(S) \text{ und } u \textit{ sieht } v\}$ und $w(uv) = |uv|$.
Dabei gilt u **sieht** $v :\Leftrightarrow \overline{uv} \subset \mathcal{C}_{\text{free}} = \mathbb{R}^2 \setminus \bigcup S$

Definiere $S^\star = S \cup \{s, t\}$ und $G_{\text{vis}}(S^\star)$ analog.

Lemma 1 $\Rightarrow$ Der kürzeste st-Weg, der die Hindernisse in S vermeidet, entspricht einem kürzesten Weg in $G_{\text{vis}}(S^\star)$.

Algorithmus

$\textsc{ShortestPath}(S, s, t)$

Input: Hindernismenge S, Punkte $s, t \in \mathbb{R}^2 \setminus \bigcup S$
Output: kürzester kollisionsfreier st-Weg in S

1 $G_{\mathsf{vis}} \leftarrow \textsc{VisibilityGraph}(S \cup \{s, t\})$
2 **foreach** $uv \in E_{\mathsf{vis}}(S)$ **do** $w(uv) \leftarrow |uv|$
3 **return** $\textsc{Dijkstra}(G_{\mathsf{vis}}, w, s, t)$

Algorithmus

Karlsruhe Institute of Technology

$\textsc{ShortestPath}(S, s, t)$

$n = |V(S)|, m = |E_{\text{vis}}(S)|$

Input: Hindernismenge S, Punkte $s, t \in \mathbb{R}^2 \setminus \bigcup S$

Output: kürzester kollisionsfreier st-Weg in S

1	$G_{\text{vis}} \leftarrow \textsc{VisibilityGraph}(S \cup \{s, t\})$	?
2	**foreach** $uv \in E_{\text{vis}}(S)$ **do** $w(uv) \leftarrow \lvert uv \rvert$	$O(m)$
3	**return** $\textsc{Dijkstra}(G_{\text{vis}}, w, s, t)$	$O(n \log n + m)$

Algorithmus

$\textsc{ShortestPath}(S, s, t)$

$n = |V(S)|, m = |E_{\text{vis}}(S)|$

Input: Hindernismenge S, Punkte $s, t \in \mathbb{R}^2 \setminus \bigcup S$

Output: kürzester kollisionsfreier st-Weg in S

1 $G_{\text{vis}} \leftarrow \textsc{VisibilityGraph}(S \cup \{s, t\})$ $\quad O(n^2 \log n)$

2 **foreach** $uv \in E_{\text{vis}}(S)$ **do** $w(uv) \leftarrow |uv|$ $\quad O(m)$

3 **return** $\textsc{Dijkstra}(G_{\text{vis}}, w, s, t)$ $\quad O(n \log n + m)$

$O(n^2 \log n)$

Satz 1: Ein kürzester st-Weg in einem Gebiet mit Polygon-Hindernissen mit n Kanten kann in $O(n^2 \log n)$ Zeit berechnet werden.

Sichtbarkeitsgraph berechnen

Karlsruhe Institute of Technology

$\textsc{VisibilityGraph}(S)$

Input: Menge disjunkter Polygone S

Output: Sichtbarkeitsgraph $G_{\mathsf{vis}}(S)$

1 $E \leftarrow \emptyset$

2 **foreach** $v \in V(S)$ **do**

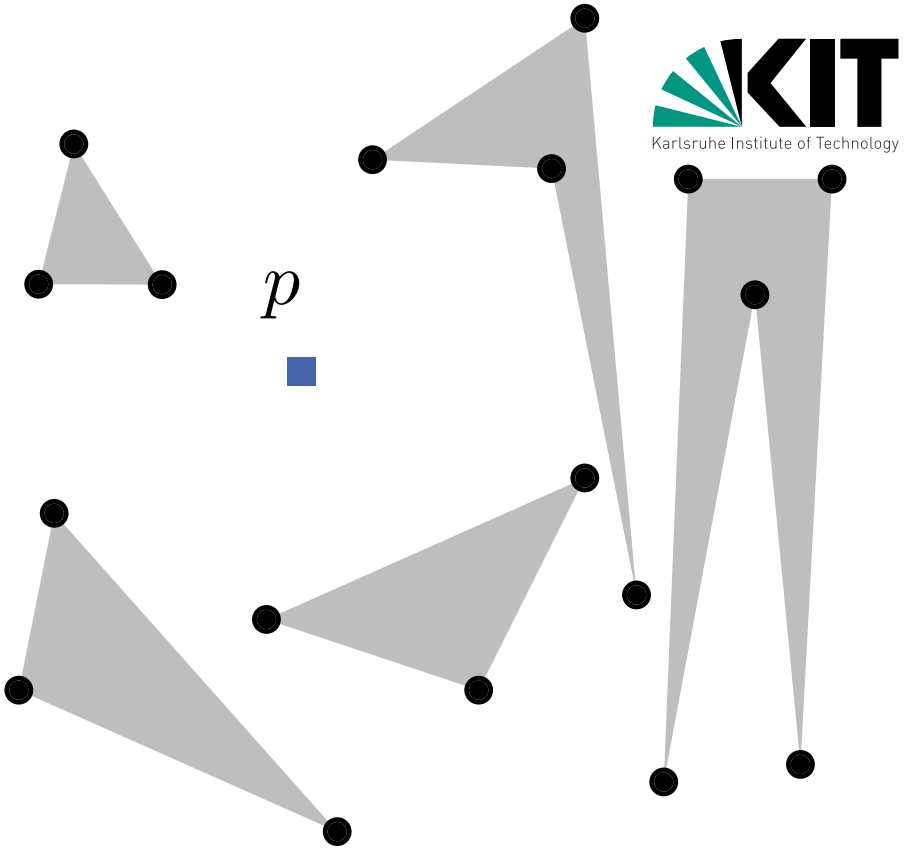
3 $\quad W \leftarrow \textsc{VisibleVertices}(v, S)$

4 $\quad E \leftarrow E \cup \{vw \mid w \in W\}$

5 **return** E

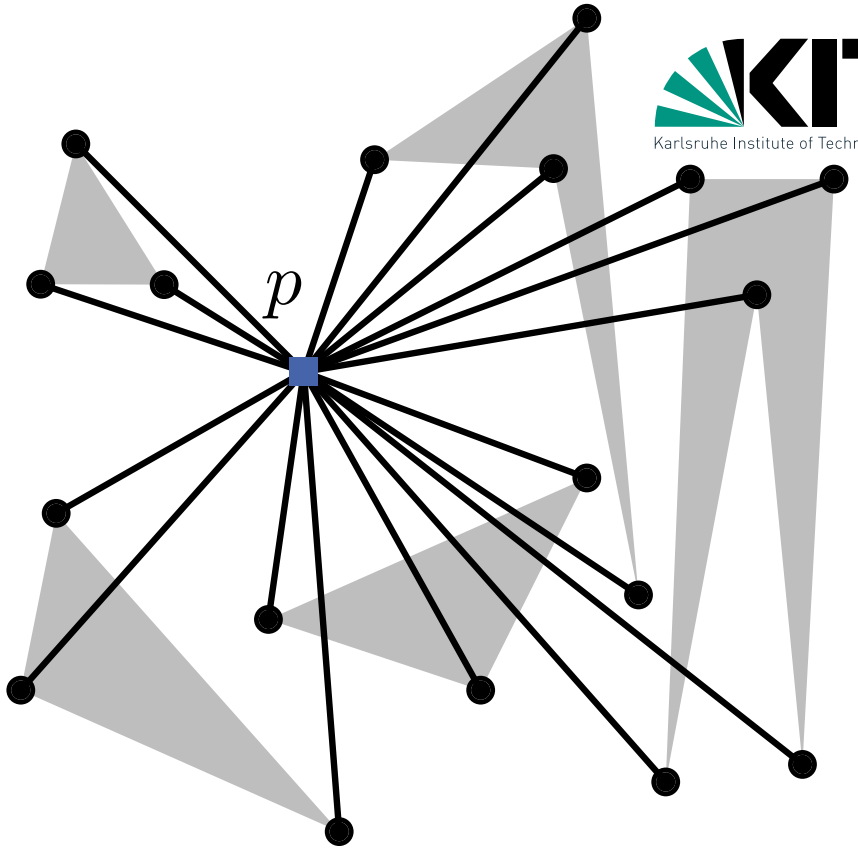
Sichtbare Knoten berechnen

$\textsc{VisibleVertices}(p, S)$



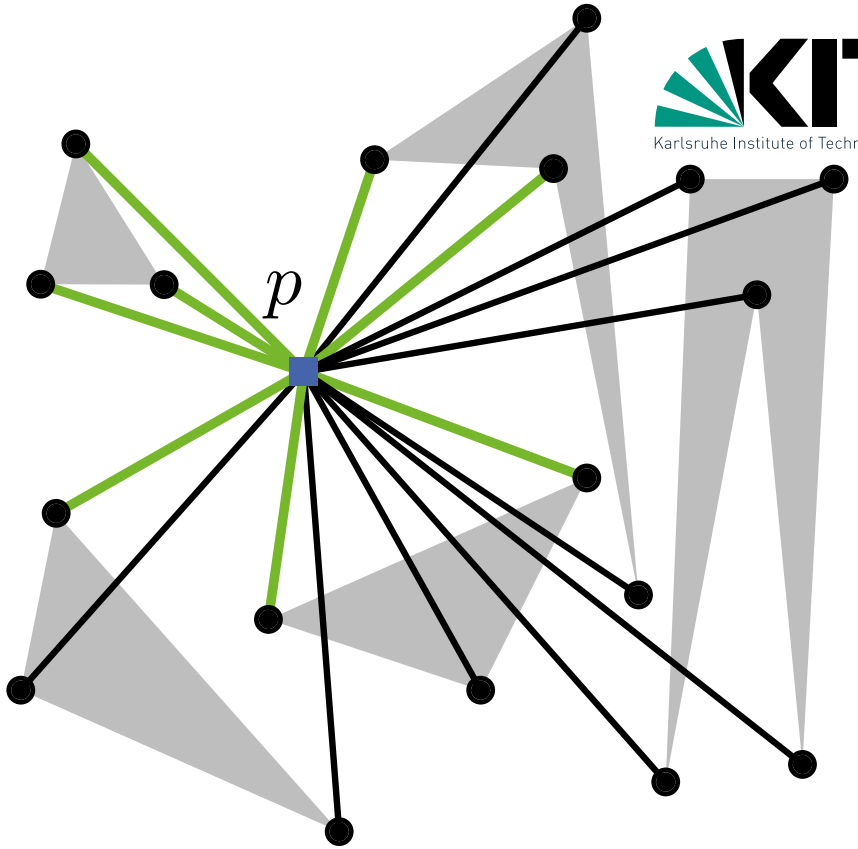
Sichtbare Knoten berechnen

VisibleVertices(p, S)



Sichtbare Knoten berechnen

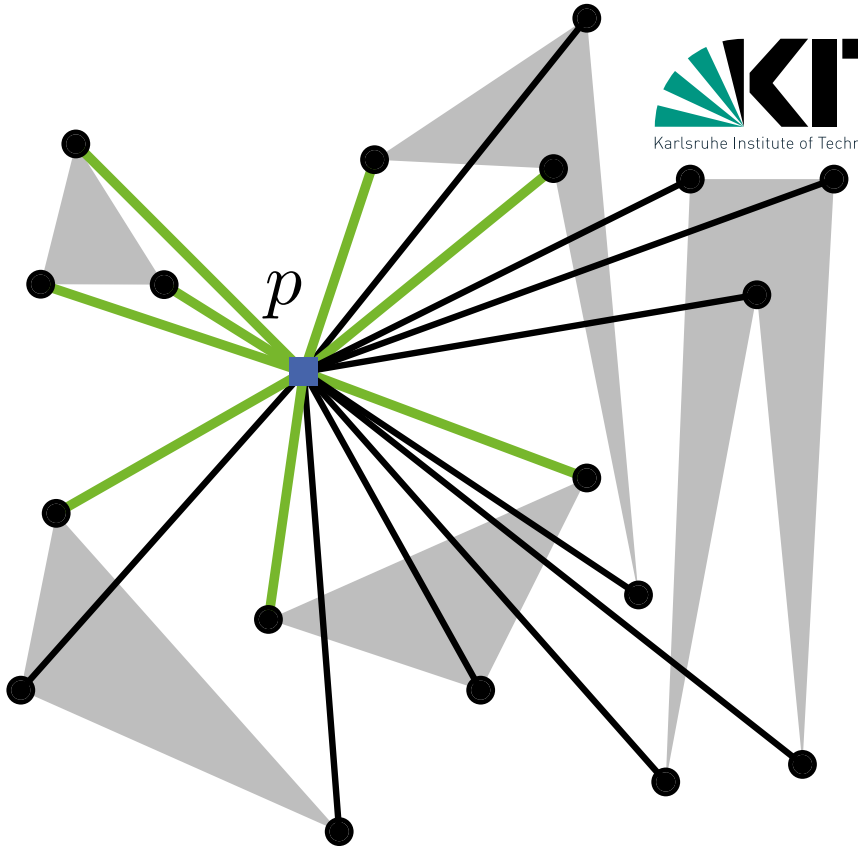
$\textsc{VisibleVertices}(p, S)$



Sichtbare Knoten berechnen

$\textsc{VisibleVertices}(p, S)$

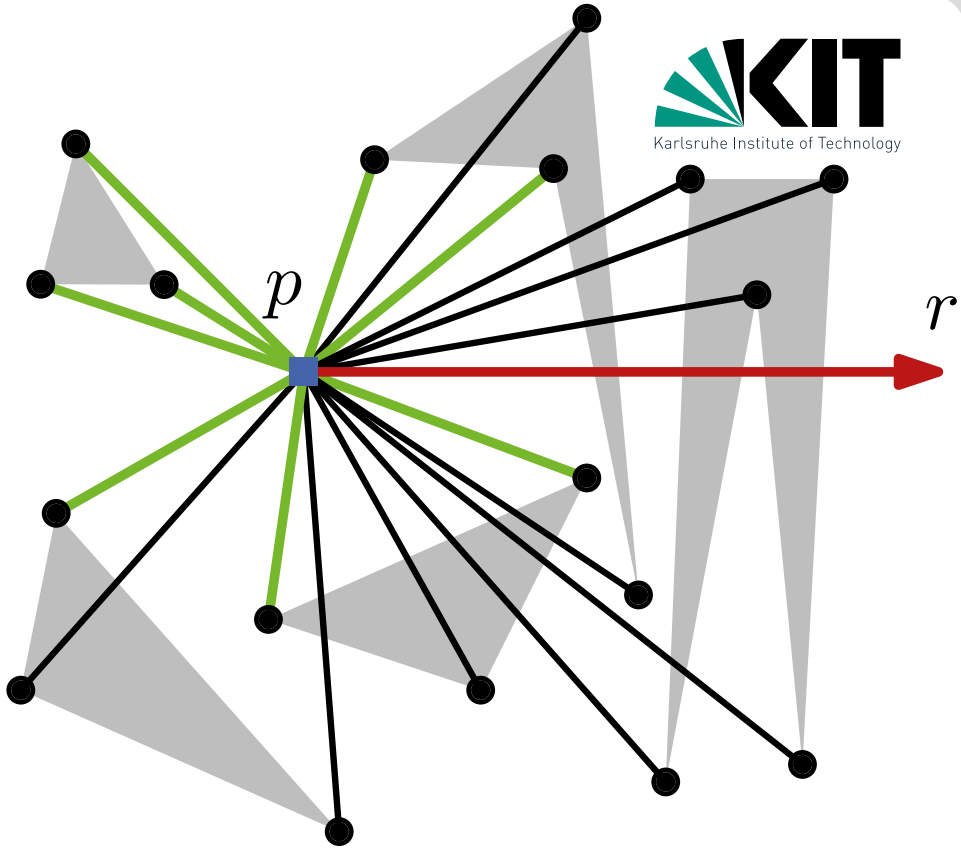
Aufgabe: Gegeben p und S finde in $O(n \log n)$ Zeit alle von p aus sichtbaren Knoten in $V(S)$!



Sichtbare Knoten berechnen

$\textsc{VisibleVertices}(p, S)$

$r \leftarrow \{p + k\binom{1}{0} \mid k \in \mathbb{R}_0^+\}$

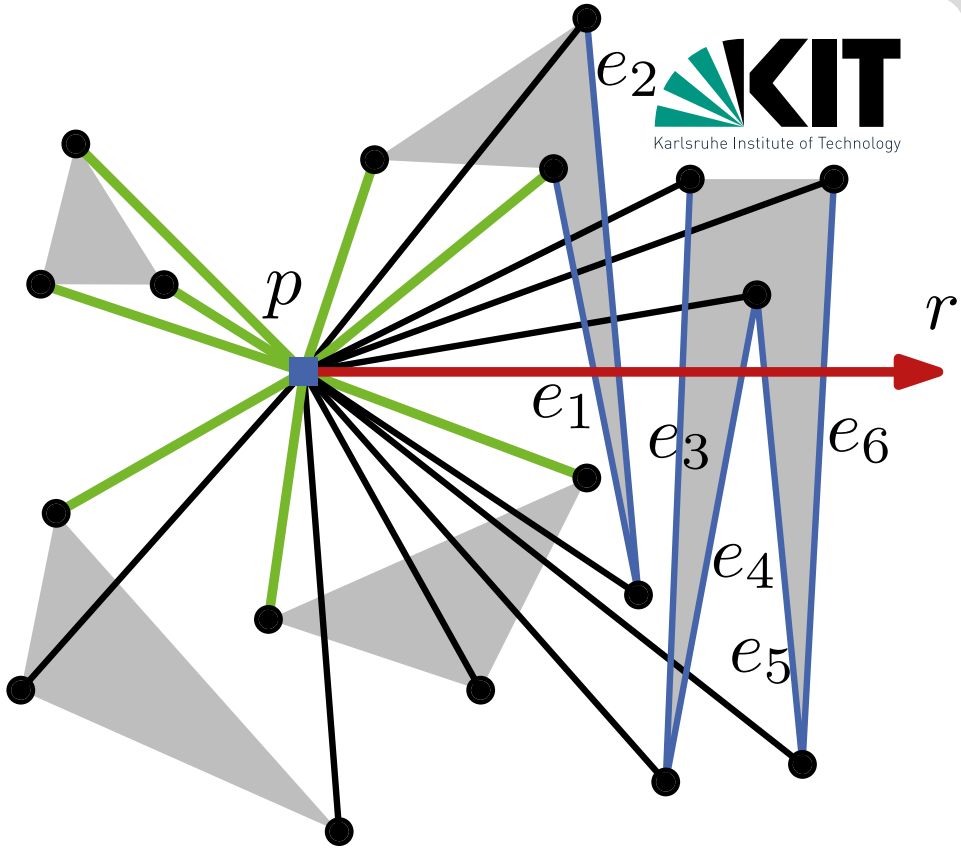


Sichtbare Knoten berechnen

$\textsc{VisibleVertices}(p, S)$

$r \leftarrow \{p + k\binom{1}{0} \mid k \in \mathbb{R}_0^+\}$

$I \leftarrow \{e \in E(S) \mid e \cap r \neq \emptyset\}$



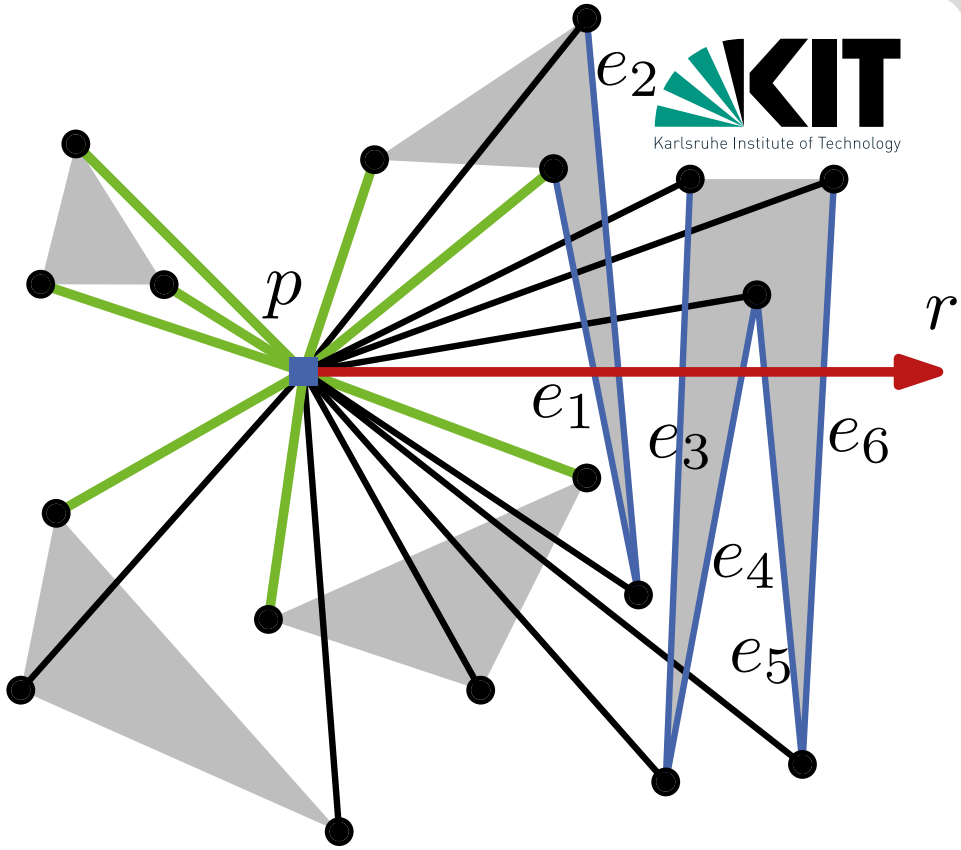
Sichtbare Knoten berechnen

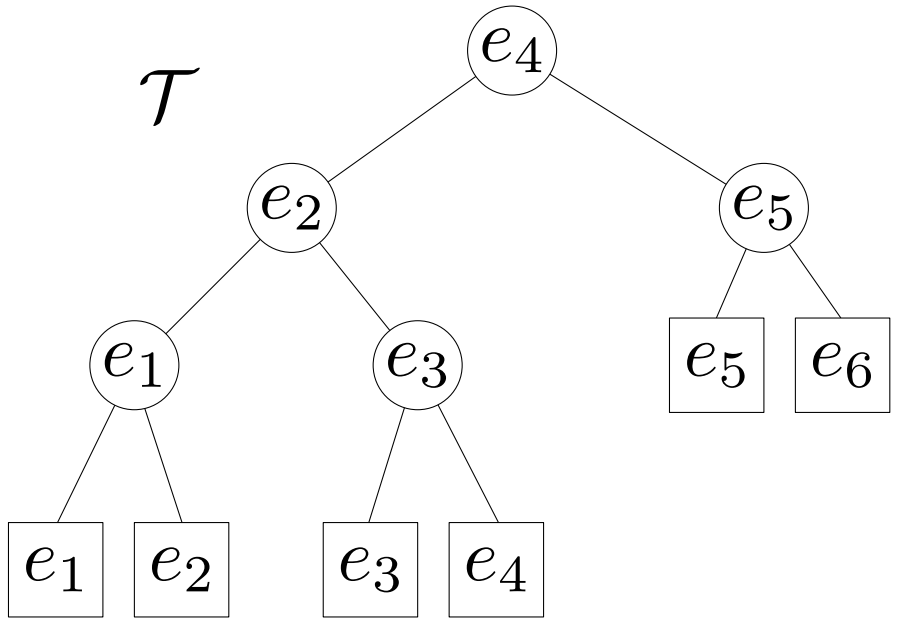
$\textsc{VisibleVertices}(p, S)$

$r \leftarrow \{p + k\binom{1}{0} \mid k \in \mathbb{R}_0^+\}$

$I \leftarrow \{e \in E(S) \mid e \cap r \neq \emptyset\}$

$\mathcal{T} \leftarrow \text{balancedBinaryTree}(I)$





Sichtbare Knoten berechnen

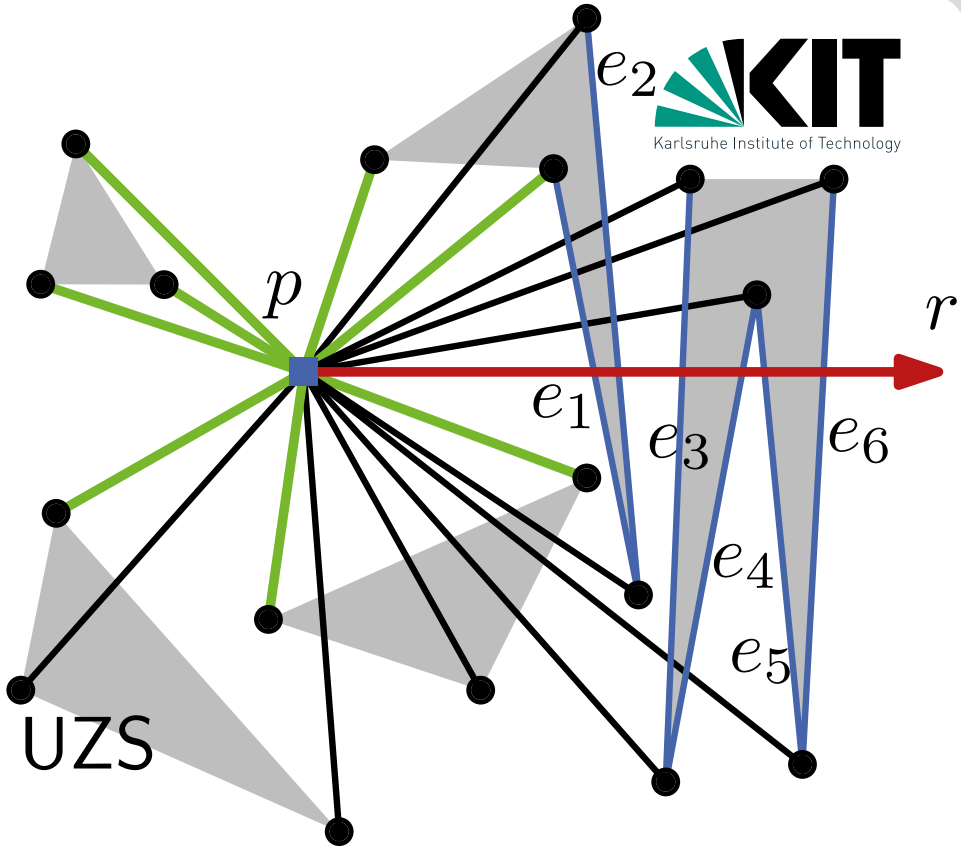
$\textsc{VisibleVertices}(p, S)$

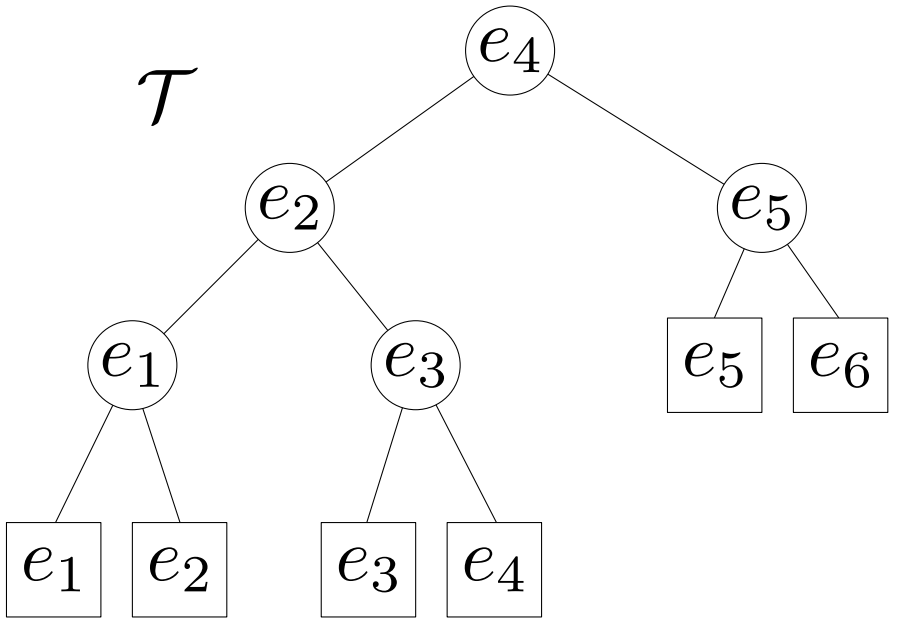
$r \leftarrow \{p + k\binom{1}{0} \mid k \in \mathbb{R}_0^+\}$

$I \leftarrow \{e \in E(S) \mid e \cap r \neq \emptyset\}$

$\mathcal{T} \leftarrow \text{balancedBinaryTree}(I)$

$w_1, \ldots, w_n \leftarrow$ sortiere $V(S)$ im UZS





Sichtbare Knoten berechnen

$\textsc{VisibleVertices}(p, S)$

$r \leftarrow \{p + k\binom{1}{0} \mid k \in \mathbb{R}_0^+\}$

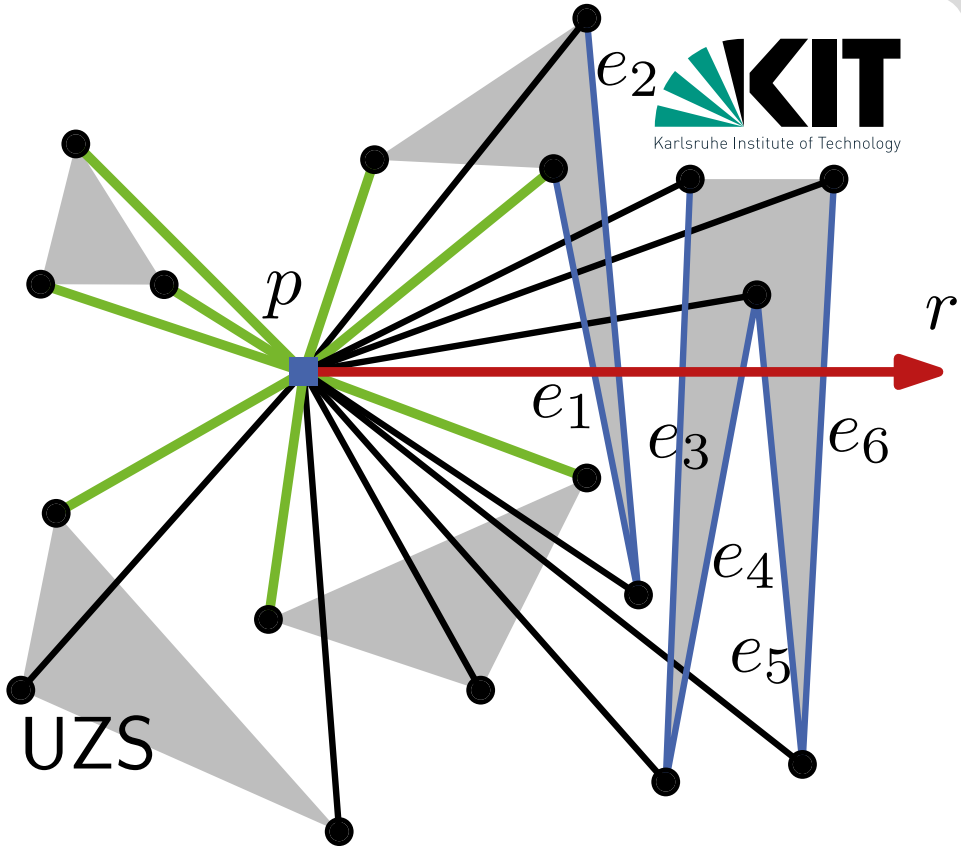
$I \leftarrow \{e \in E(S) \mid e \cap r \neq \emptyset\}$

$\mathcal{T} \leftarrow$ balancedBinaryTree(I)

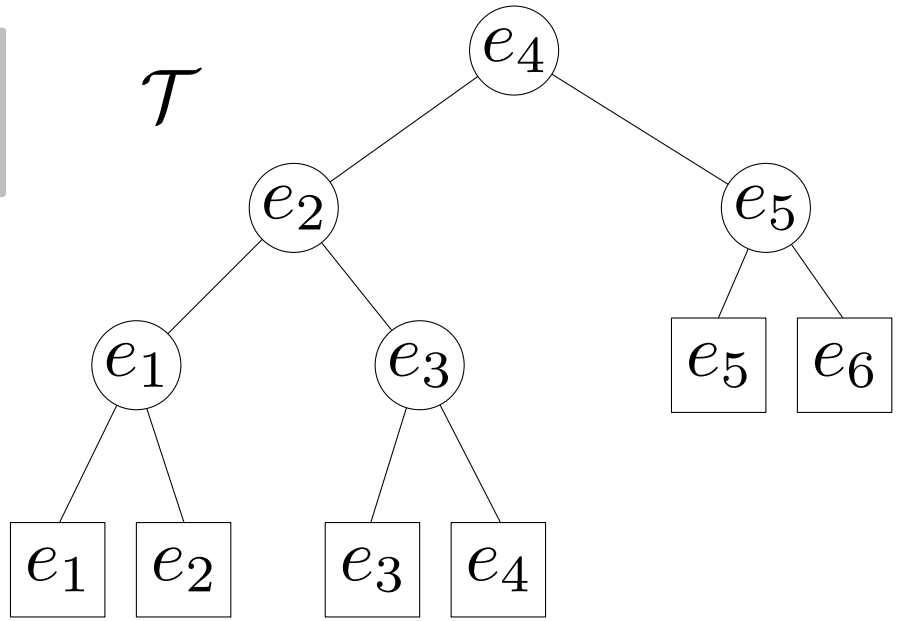
$w_1, \ldots, w_n \leftarrow$ sortiere $V(S)$ im UZS

$v \prec v' :\Leftrightarrow$

$\angle v < \angle v'$ or
$(\angle v = \angle v'$ and $|pv| < |pv'|)$



$\mathcal{T}$



Sichtbare Knoten berechnen

$\textsc{VisibleVertices}(p, S)$

$r \leftarrow \{p + k\binom{1}{0} \mid k \in \mathbb{R}_0^+\}$

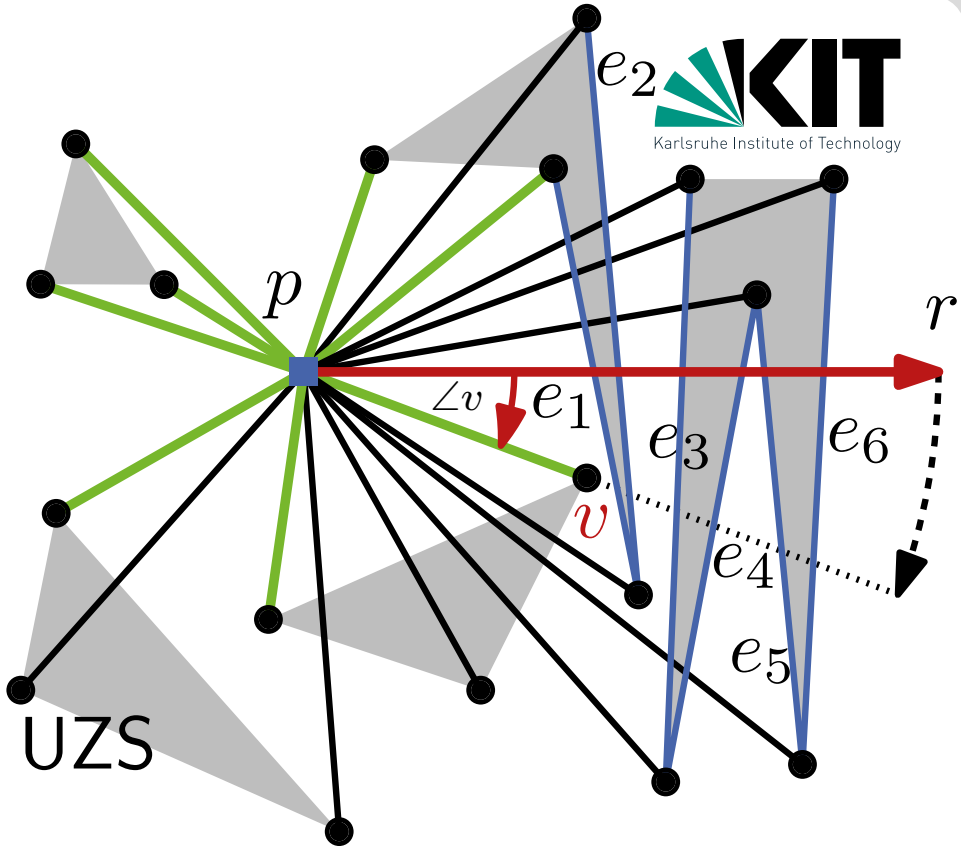
$I \leftarrow \{e \in E(S) \mid e \cap r \neq \emptyset\}$

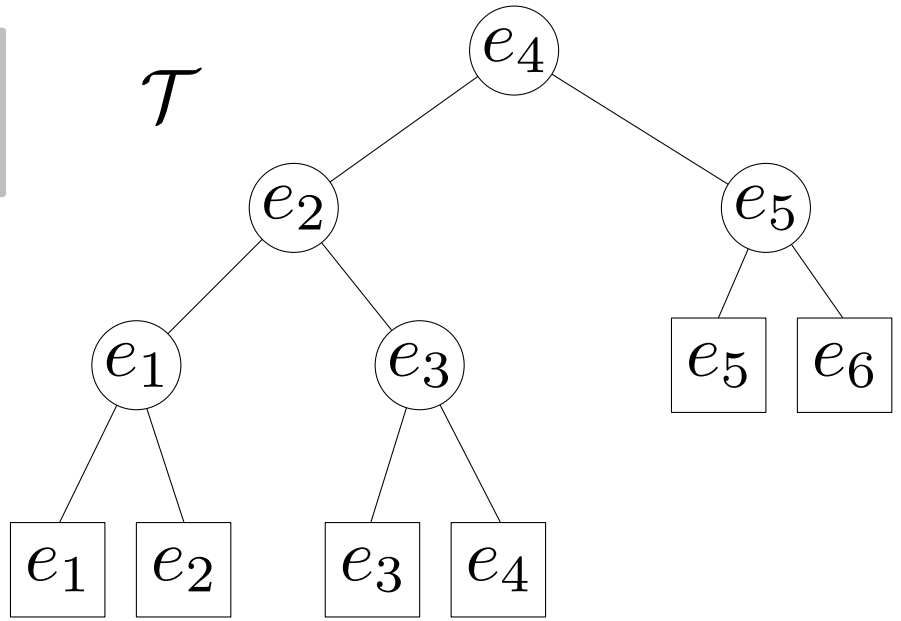
$\mathcal{T} \leftarrow$ balancedBinaryTree(I)

$w_1, \ldots, w_n \leftarrow$ sortiere $V(S)$ im UZS

$v \prec v' :\Leftrightarrow$

$\angle v < \angle v'$ or
$(\angle v = \angle v'$ and $|pv| < |pv'|)$





Sichtbare Knoten berechnen

$\textsc{VisibleVertices}(p, S)$

$r \leftarrow \{p + k\binom{1}{0} \mid k \in \mathbb{R}_0^+\}$

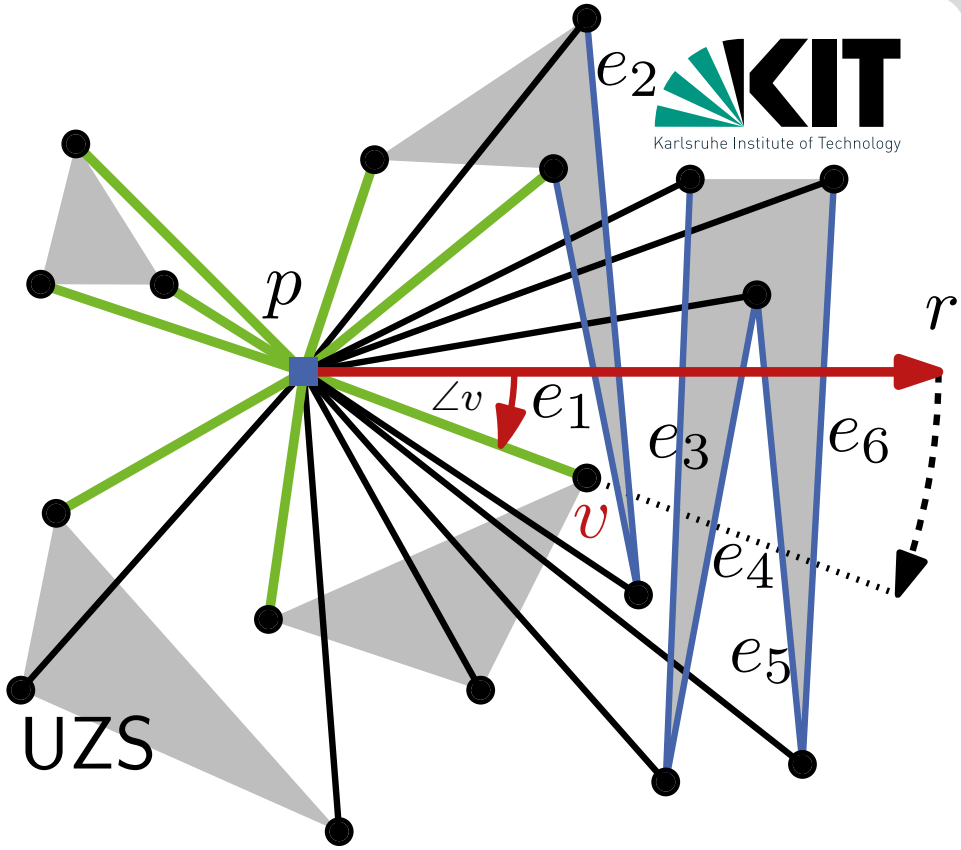
$I \leftarrow \{e \in E(S) \mid e \cap r \neq \emptyset\}$

$\mathcal{T} \leftarrow$ balancedBinaryTree(I)

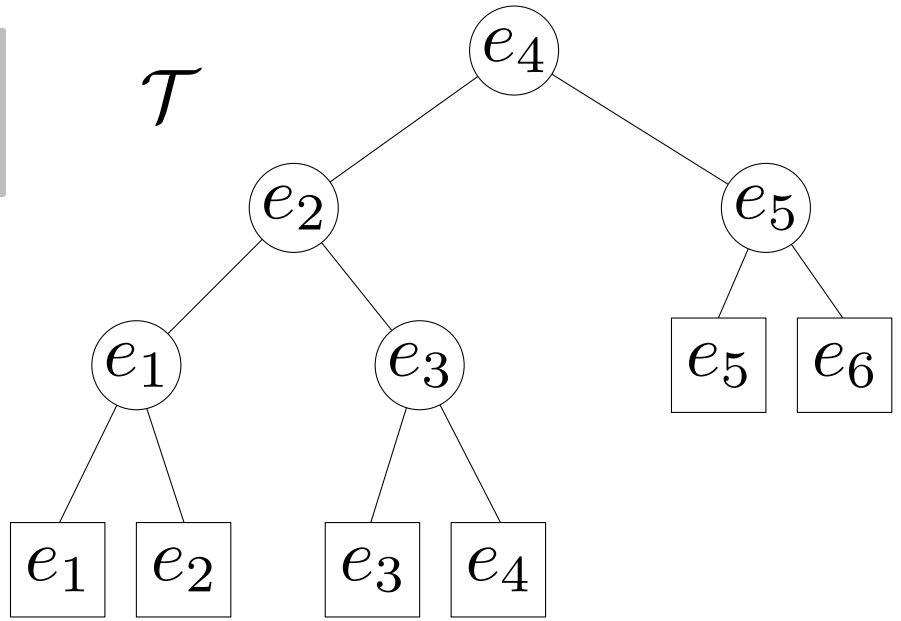
$w_1, \ldots, w_n \leftarrow$ sortiere $V(S)$ im UZS

$v \prec v' :\Leftrightarrow$

$\angle v < \angle v'$ or
$(\angle v = \angle v'$ and $|pv| < |pv'|)$



$\mathcal{T}$



Sweep-Verfahren mit Rotation

Sichtbare Knoten berechnen

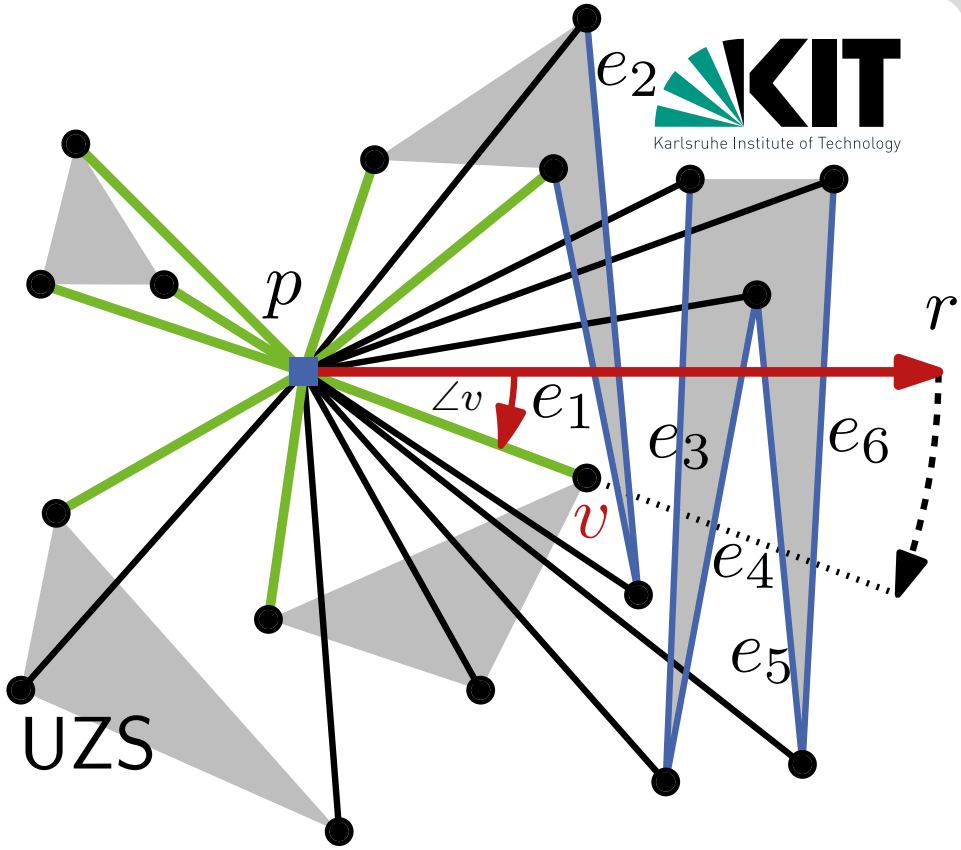
$\textsc{VisibleVertices}(p, S)$

$r \leftarrow \{p + k\binom{1}{0} \mid k \in \mathbb{R}_0^+\}$

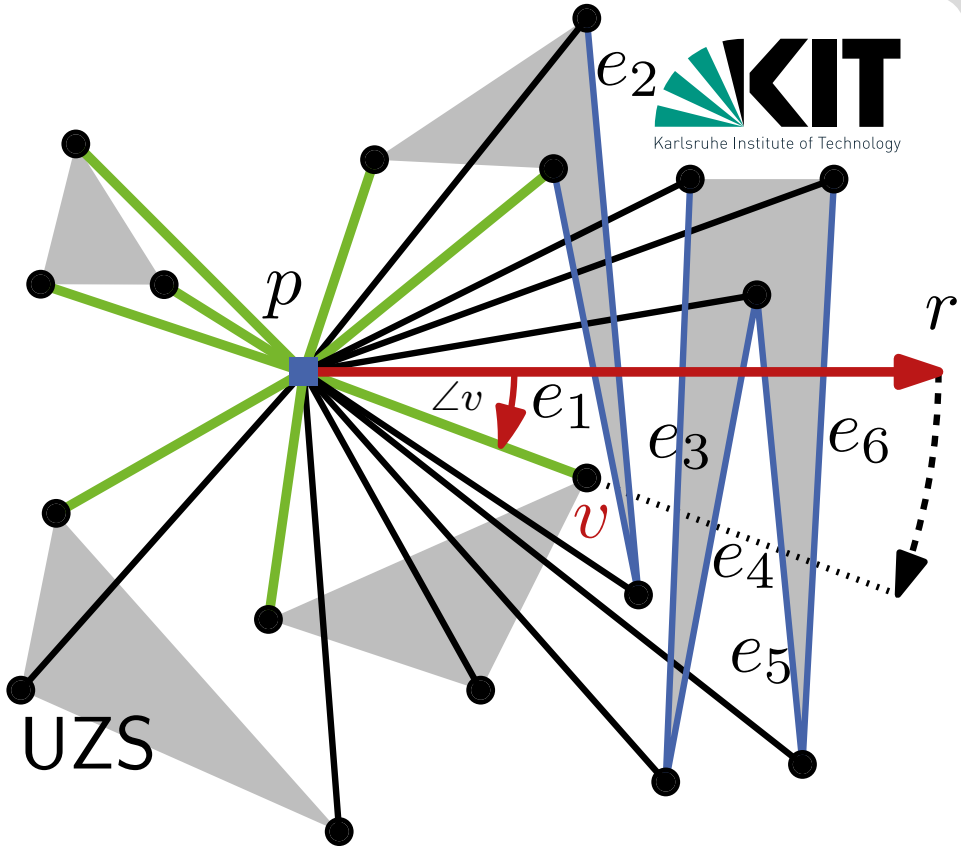
$I \leftarrow \{e \in E(S) \mid e \cap r \neq \emptyset\}$

$\mathcal{T} \leftarrow \text{balancedBinaryTree}(I)$

$w_1, \ldots, w_n \leftarrow$ sortiere $V(S)$ im UZS



Sichtbare Knoten berechnen



$\textsc{VisibleVertices}(p, S)$

$r \leftarrow \{p + k\binom{1}{0} \mid k \in \mathbb{R}_0^+\}$

$I \leftarrow \{e \in E(S) \mid e \cap r \neq \emptyset\}$

$\mathcal{T} \leftarrow$ balancedBinaryTree(I)

$w_1, \ldots, w_n \leftarrow$ sortiere $V(S)$ im UZS

$W \leftarrow \emptyset$

for $i = 1$ **to** n **do**

  if $\textsc{Visible}(p, w_i)$ **then**

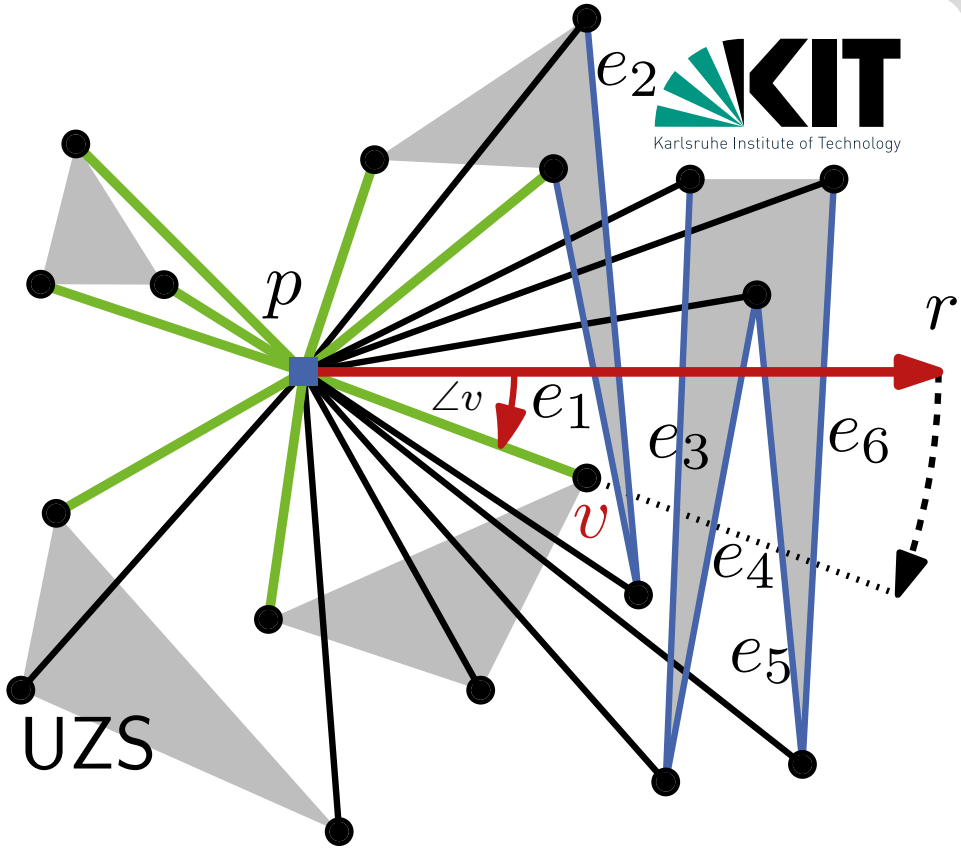
    $W \leftarrow W \cup \{w_i\}$

  füge in $\mathcal{T}$ zu w_i inzidente Kante aus $\overrightarrow{pw_i}^+$ ein

  lösche aus $\mathcal{T}$ zu w_i inzidente Kanten aus $\overrightarrow{pw_i}^-$

return W

Sichtbare Knoten berechnen



$\textsc{VisibleVertices}(p, S)$

$r \leftarrow \{p + k\binom{1}{0} \mid k \in \mathbb{R}_0^+\}$

$I \leftarrow \{e \in E(S) \mid e \cap r \neq \emptyset\}$

$\mathcal{T} \leftarrow$ balancedBinaryTree(I)

$w_1, \ldots, w_n \leftarrow$ sortiere $V(S)$ im UZS

$W \leftarrow \emptyset$

for $i = 1$ **to** n **do**

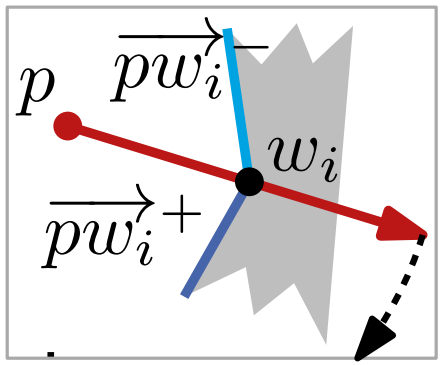
  if $\textsc{Visible}(p, w_i)$ **then**

    $W \leftarrow W \cup \{w_i\}$

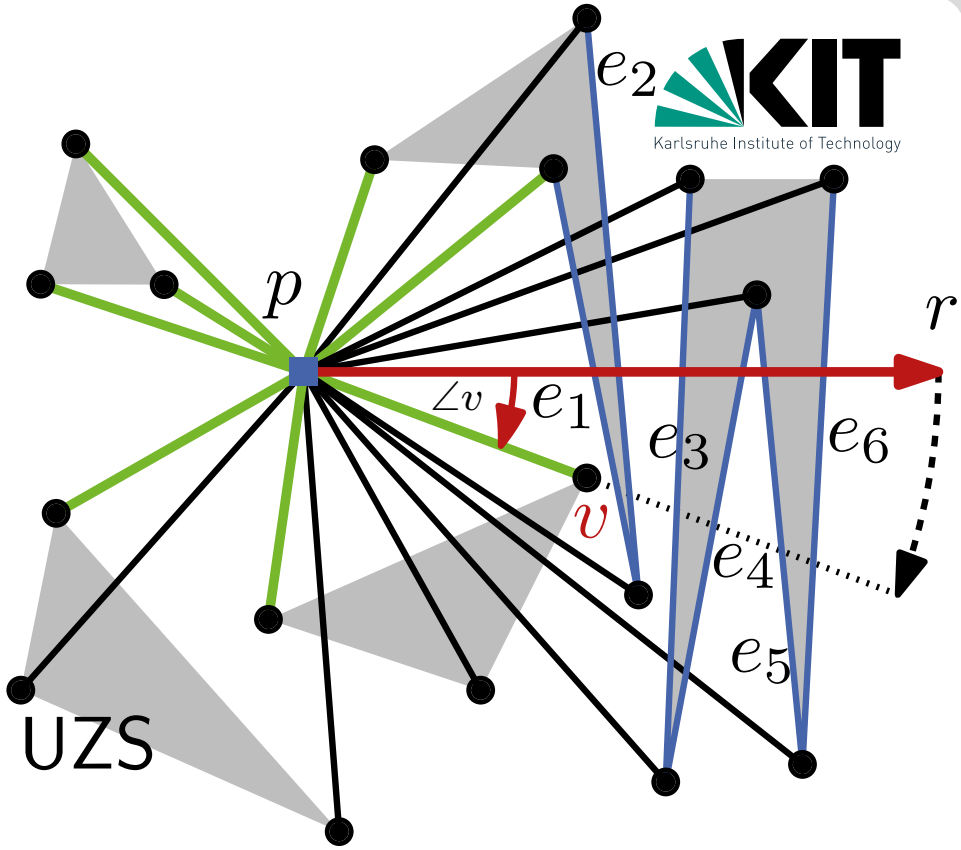
  füge in $\mathcal{T}$ zu w_i inzidente Kante aus $\overrightarrow{pw_i}^+$ ein

  lösche aus $\mathcal{T}$ zu w_i inzidente Kanten aus $\overrightarrow{pw_i}^-$

return W



Sichtbare Knoten berechnen



$\textsc{VisibleVertices}(p, S)$

$r \leftarrow \{p + k\binom{1}{0} \mid k \in \mathbb{R}_0^+\}$

$I \leftarrow \{e \in E(S) \mid e \cap r \neq \emptyset\}$

$\mathcal{T} \leftarrow$ balancedBinaryTree(I)

$w_1, \ldots, w_n \leftarrow$ sortiere $V(S)$ im UZS

$W \leftarrow \emptyset$

for $i = 1$ **to** n **do**

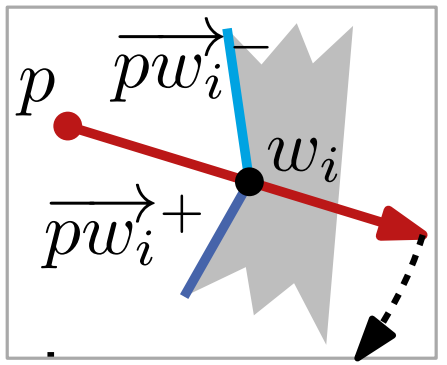
  if $\textsc{Visible}(p, w_i)$ **then**

    $W \leftarrow W \cup \{w_i\}$

  füge in $\mathcal{T}$ zu w_i inzidente Kante aus $\overrightarrow{pw_i}^+$ ein

  lösche aus $\mathcal{T}$ zu w_i inzidente Kanten aus $\overrightarrow{pw_i}^-$

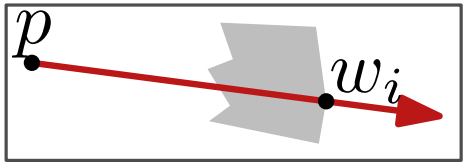
return W



Fallunterscheidung Sichtbarkeit

$\textsc{Visible}(p, w_i)$

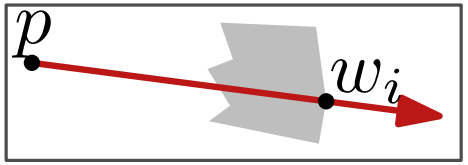
if $\overline{pw_i}$ schneidet Polygon von w_i **then**
  return false

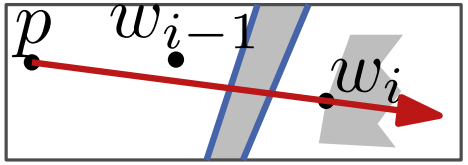


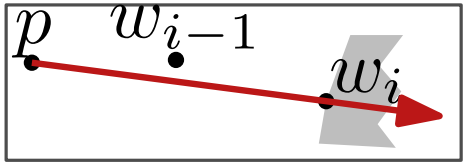
Fallunterscheidung Sichtbarkeit

VISIBLE(p, w_i)

if $\overline{pw_i}$ schneidet Polygon von w_i **then**
  return false
if $i = 1$ oder $w_{i-1} \notin \overline{pw_i}$ **then**
  $e \leftarrow$ Kante im linkesten Blatt von $\mathcal{T}$
  if $e \neq$ nil und $\overline{pw_i} \cap e \neq \emptyset$ **then**
    return false
  else return true



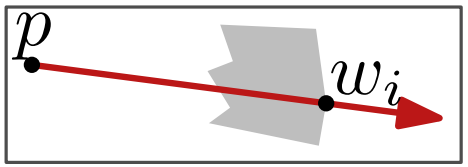


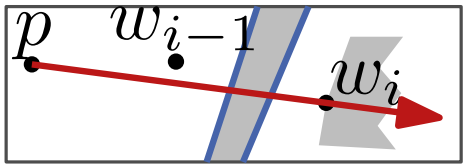


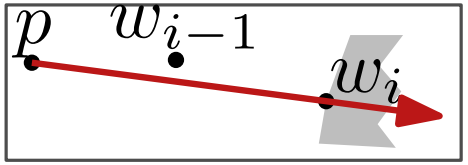
Fallunterscheidung Sichtbarkeit

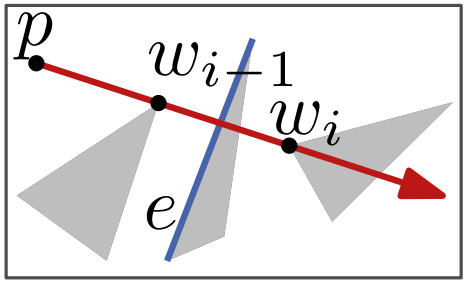
$\textsc{Visible}(p, w_i)$

if $\overline{pw_i}$ schneidet Polygon von w_i **then**
  return false
if $i = 1$ oder $w_{i-1} \notin \overline{pw_i}$ **then**
  $e \leftarrow$ Kante im linkesten Blatt von $\mathcal{T}$
  if $e \neq$ nil und $\overline{pw_i} \cap e \neq \emptyset$ **then**
    return false
  else return true
else
  if w_{i-1} nicht sichtbar **then**
    return false
  else
    $e \leftarrow$ suche Kante in $\mathcal{T}$, die $\overline{w_{i-1}w_i}$ schneidet
    if $e \neq$ nil **then return** false
    else return true









Zusammenfassung

Satz 1: Ein kürzester st-Weg in einem Gebiet mit Polygon-Hindernissen mit n Kanten kann in $O(n^2 \log n)$ Zeit berechnet werden.

Zusammenfassung

Satz 1: Ein kürzester st-Weg in einem Gebiet mit Polygon-Hindernissen mit n Kanten kann in $O(n^2 \log n)$ Zeit berechnet werden.

Beweis:

- Korrektheit folgt direkt aus Lemma 1

Zusammenfassung

Satz 1: Ein kürzester st-Weg in einem Gebiet mit Polygon-Hindernissen mit n Kanten kann in $O(n^2 \log n)$ Zeit berechnet werden.

Beweis:

- Korrektheit folgt direkt aus Lemma 1
- Laufzeit:
  - VisibleVertices benötigt $O(n \log n)$ Zeit pro Knoten
  - n Aufrufe von VisibleVertices

Zusammenfassung

Satz 1: Ein kürzester st-Weg in einem Gebiet mit Polygon-Hindernissen mit n Kanten kann in $O(n^2 \log n)$ Zeit berechnet werden.

Beweis:

- Korrektheit folgt direkt aus Lemma 1
- Laufzeit:
  - VisibleVertices benötigt $O(n \log n)$ Zeit pro Knoten
  - n Aufrufe von VisibleVertices

□

Diskussion

Roboter sind meistens nicht punktförmig...

Diskussion

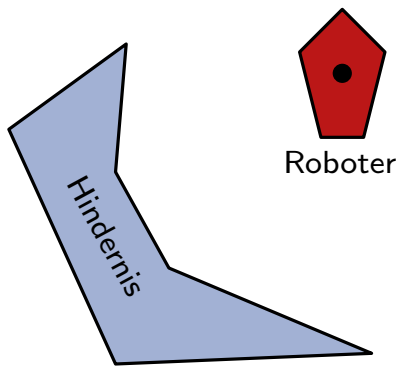
Roboter sind meistens nicht punktförmig...

Für den Fall von Robotern, deren Grundfläche ein konvexes Polygon ist und die nicht rotieren können, geht es trotzdem durch geeignete Vergrößerung der Hindernisse
($\rightarrow$ Minkowski-Summe, Kap. 13 in [BCKO08]).

Diskussion

Roboter sind meistens nicht punktförmig...

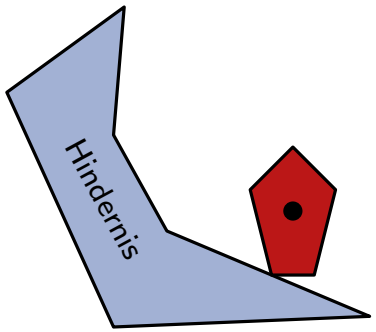
Für den Fall von Robotern, deren Grundfläche ein konvexes Polygon ist und die nicht rotieren können, geht es trotzdem durch geeignete Vergrößerung der Hindernisse
($\rightarrow$ Minkowski-Summe, Kap. 13 in [BCKO08]).



Diskussion

Roboter sind meistens nicht punktförmig...

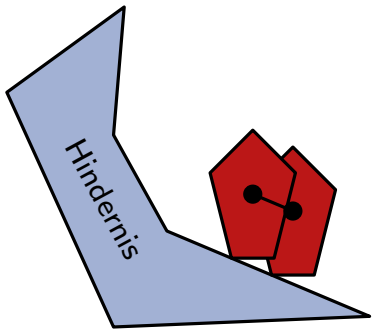
Für den Fall von Robotern, deren Grundfläche ein konvexes Polygon ist und die nicht rotieren können, geht es trotzdem durch geeignete Vergrößerung der Hindernisse
($\rightarrow$ Minkowski-Summe, Kap. 13 in [BCKO08]).



Diskussion

Roboter sind meistens nicht punktförmig...

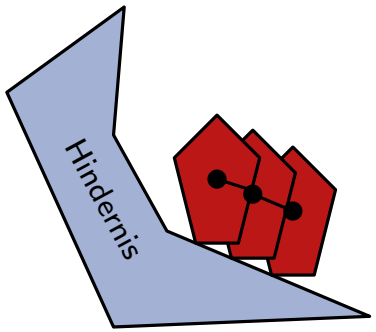
Für den Fall von Robotern, deren Grundfläche ein konvexes Polygon ist und die nicht rotieren können, geht es trotzdem durch geeignete Vergrößerung der Hindernisse
($\rightarrow$ Minkowski-Summe, Kap. 13 in [BCKO08]).



Diskussion

Roboter sind meistens nicht punktförmig...

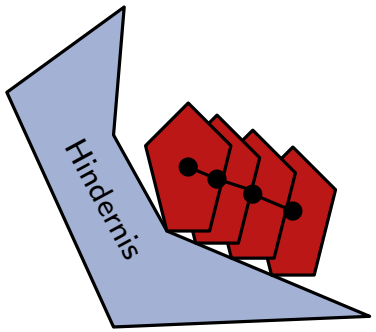
Für den Fall von Robotern, deren Grundfläche ein konvexes Polygon ist und die nicht rotieren können, geht es trotzdem durch geeignete Vergrößerung der Hindernisse
($\rightarrow$ Minkowski-Summe, Kap. 13 in [BCKO08]).



Diskussion

Roboter sind meistens nicht punktförmig...

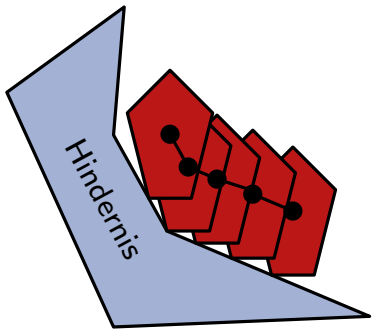
Für den Fall von Robotern, deren Grundfläche ein konvexes Polygon ist und die nicht rotieren können, geht es trotzdem durch geeignete Vergrößerung der Hindernisse
($\rightarrow$ Minkowski-Summe, Kap. 13 in [BCKO08]).



Diskussion

Roboter sind meistens nicht punktförmig...

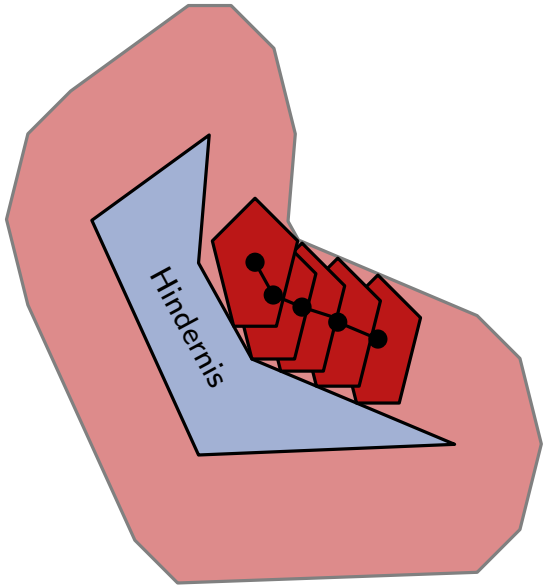
Für den Fall von Robotern, deren Grundfläche ein konvexes Polygon ist und die nicht rotieren können, geht es trotzdem durch geeignete Vergrößerung der Hindernisse
($\rightarrow$ Minkowski-Summe, Kap. 13 in [BCKO08]).



Diskussion

Roboter sind meistens nicht punktförmig...

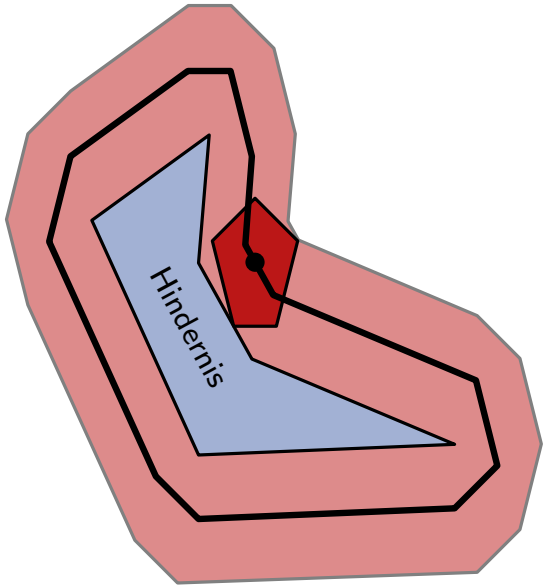
Für den Fall von Robotern, deren Grundfläche ein konvexes Polygon ist und die nicht rotieren können, geht es trotzdem durch geeignete Vergrößerung der Hindernisse
($\rightarrow$ Minkowski-Summe, Kap. 13 in [BCKO08]).



Diskussion

Roboter sind meistens nicht punktförmig...

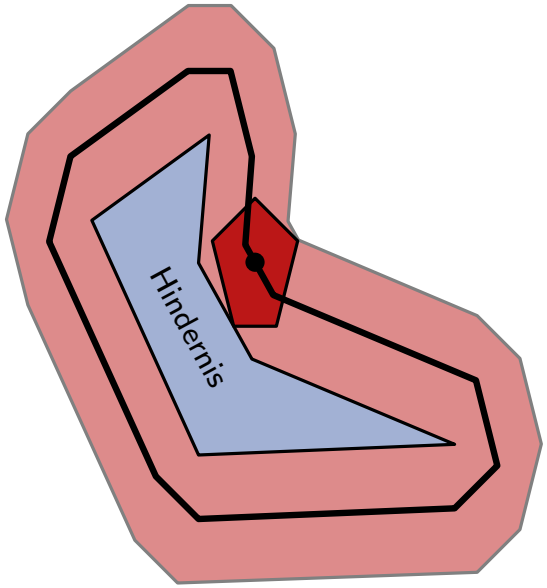
Für den Fall von Robotern, deren Grundfläche ein konvexes Polygon ist und die nicht rotieren können, geht es trotzdem durch geeignete Vergrößerung der Hindernisse
($\rightarrow$ Minkowski-Summe, Kap. 13 in [BCKO08]).



Diskussion

Roboter sind meistens nicht punktförmig...

Für den Fall von Robotern, deren Grundfläche ein konvexes Polygon ist und die nicht rotieren können, geht es trotzdem durch geeignete Vergrößerung der Hindernisse
($\rightarrow$ Minkowski-Summe, Kap. 13 in [BCKO08]).

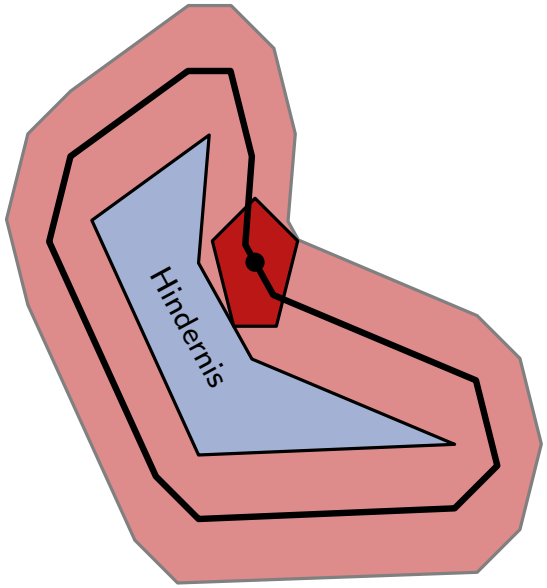


Geht es schneller als $O(n^2 \log n)$?

Diskussion

Roboter sind meistens nicht punktförmig...

Für den Fall von Robotern, deren Grundfläche ein konvexes Polygon ist und die nicht rotieren können, geht es trotzdem durch geeignete Vergrößerung der Hindernisse
($\rightarrow$ Minkowski-Summe, Kap. 13 in [BCKO08]).



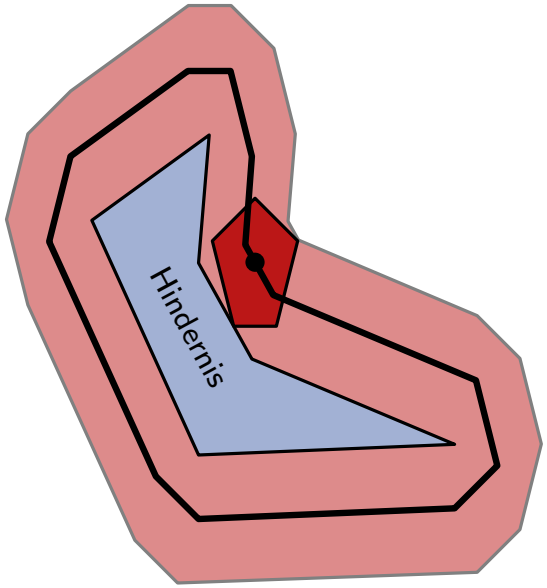
Geht es schneller als $O(n^2 \log n)$?

Ja, durch Ausnutzung der Dualität und einen simultanen Rotations-Sweep für alle Punkte im dualen Geradenarrangement geht es auch in $O(n^2)$. Da G_{vis} $\Omega(n^2)$ Kanten haben kann, lässt sich der Sichtbarkeitsgraph im Allgemeinen auch nicht schneller konstruieren.
Es gibt jedoch einen ausgabesensitiven $O(n \log n + m)$-Algorithmus.
[Ghosh, Mount 1987]

Diskussion

Roboter sind meistens nicht punktförmig...

Für den Fall von Robotern, deren Grundfläche ein konvexes Polygon ist und die nicht rotieren können, geht es trotzdem durch geeignete Vergrößerung der Hindernisse
($\rightarrow$ Minkowski-Summe, Kap. 13 in [BCKO08]).



Geht es schneller als $O(n^2 \log n)$?

Ja, durch Ausnutzung der Dualität und einen simultanen Rotations-Sweep für alle Punkte im dualen Geradenarrangement geht es auch in $O(n^2)$. Da G_{vis} $\Omega(n^2)$ Kanten haben kann, lässt sich der Sichtbarkeitsgraph im Allgemeinen auch nicht schneller konstruieren.
Es gibt jedoch einen ausgabesensitiven $O(n \log n + m)$-Algorithmus.
[Ghosh, Mount 1987]

Sucht man jedoch nur einen kürzesten Euklidischen st-Weg, gibt es einen Algorithmus mit optimaler Laufzeit $O(n \log n)$. [Hershberger, Suri 1999]