

Motion Planning

Elmar Langetepe
University of Bonn

Voronoi Diagramm von Liniensegmenten

Voronoi Diagramm von Liniensegmenten

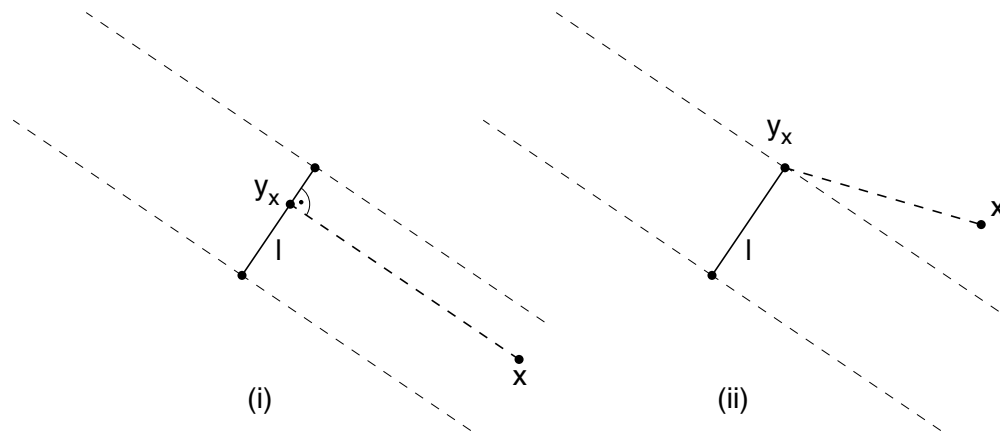
- Jetzt alle Objekte Punkte oder Liniensegmente

Voronoi Diagramm von Liniensegmenten

- Jetzt alle Objekte Punkte oder Liniensegmente
- Bisektor zwischen Punkt und Segment

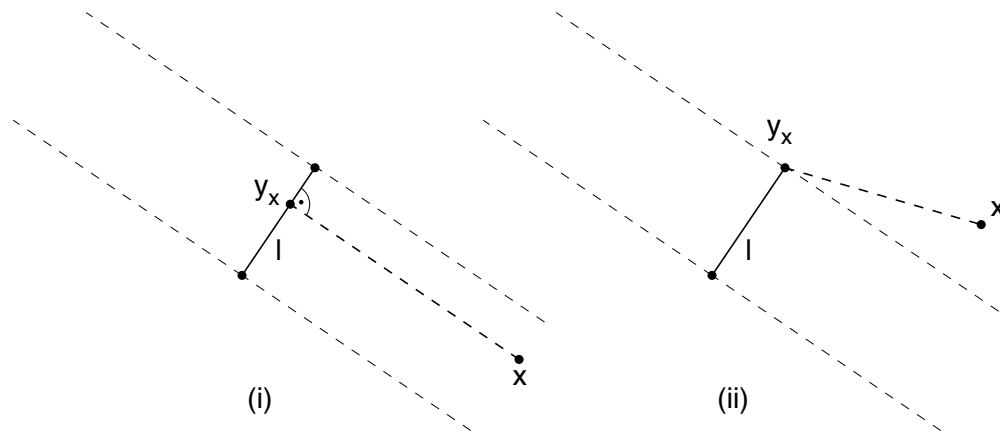
Voronoi Diagramm von Liniensegmenten

- Jetzt alle Objekte Punkte oder Liniensegmente
- Bisektor zwischen Punkt und Segment
- Abstand eines Punktes x zu Segment l , Streifen



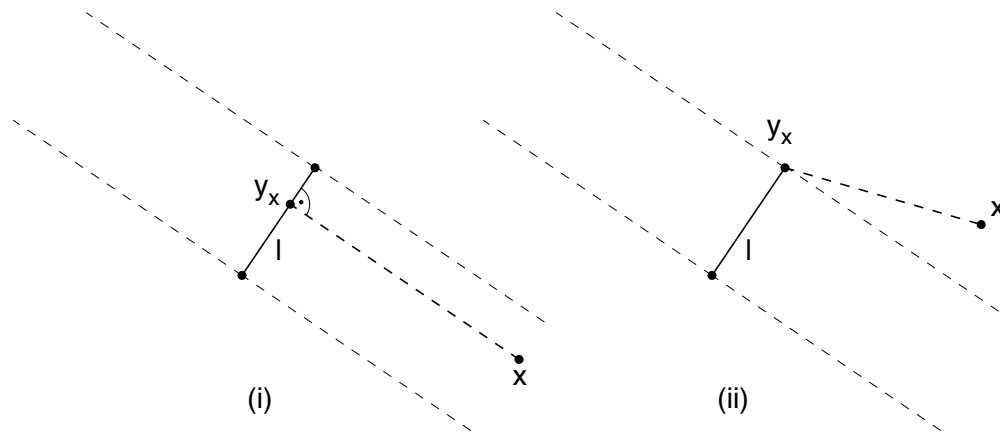
Voronoi Diagramm von Liniensegmenten

- Jetzt alle Objekte Punkte oder Liniensegmente
- Bisektor zwischen Punkt und Segment
- Abstand eines Punktes x zu Segment l , Streifen
- Bisektor zwischen zwei Segmenten l_1 und l_2



Voronoi Diagramm von Liniensegmenten

- Jetzt alle Objekte Punkte oder Liniensegmente
- Bisektor zwischen Punkt und Segment
- Abstand eines Punktes x zu Segment l , Streifen
- Bisektor zwischen zwei Segmenten l_1 und l_2
- $B(l_1, l_2) = \{x \in \mathbb{R}^2; |xl_1| = |xl_2|\}$



Bisektor von Segmenten

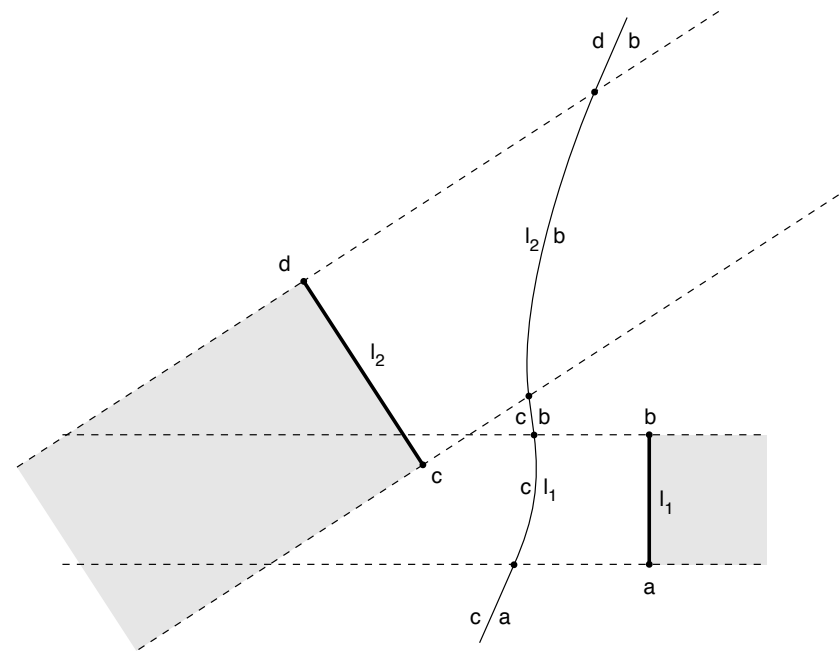
Bisektor von Segmenten

Lemma 5.24 Der Bisektor von zwei disjunkten Liniensegmenten l_1 und l_2 ist eine Kurve aus Parabelstücken, Liniensegmenten und zwei Halbgeraden.

Bisektor von Segmenten

Lemma 5.24 Der Bisektor von zwei disjunkten Liniensegmenten l_1 und l_2 ist eine Kurve aus Parabelstücken, Liniensegmenten und zwei Halbgeraden.

Verantwortungsbereiche der Streifen, Lage zueinander

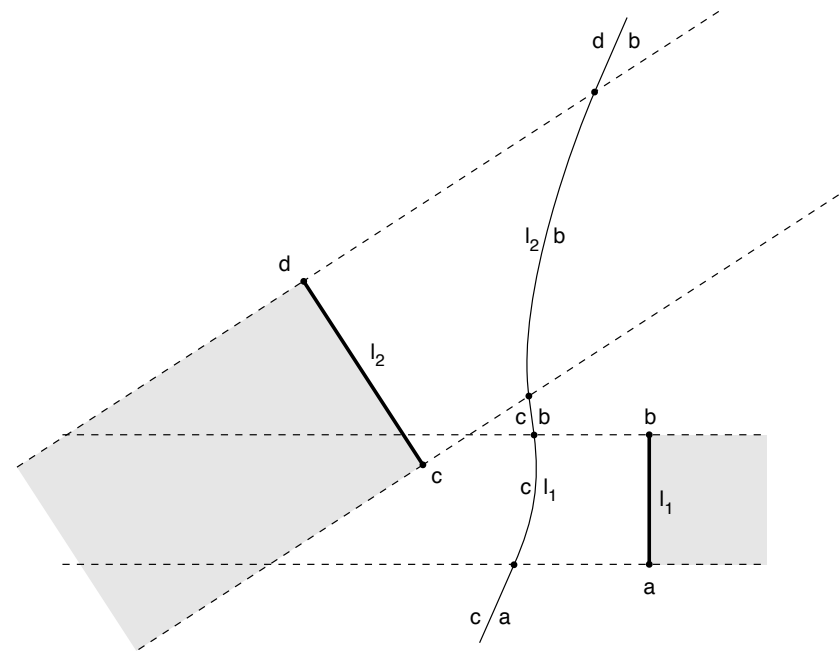


Bisektor von Segmenten

Lemma 5.24 Der Bisektor von zwei disjunkten Liniensegmenten l_1 und l_2 ist eine Kurve aus Parabelstücken, Liniensegmenten und zwei Halbgeraden.

Verantwortungsbereiche der Streifen, Lage zueinander

1. l_1 Punkt, l_2 Punkt:
Bisektorstück Gerade

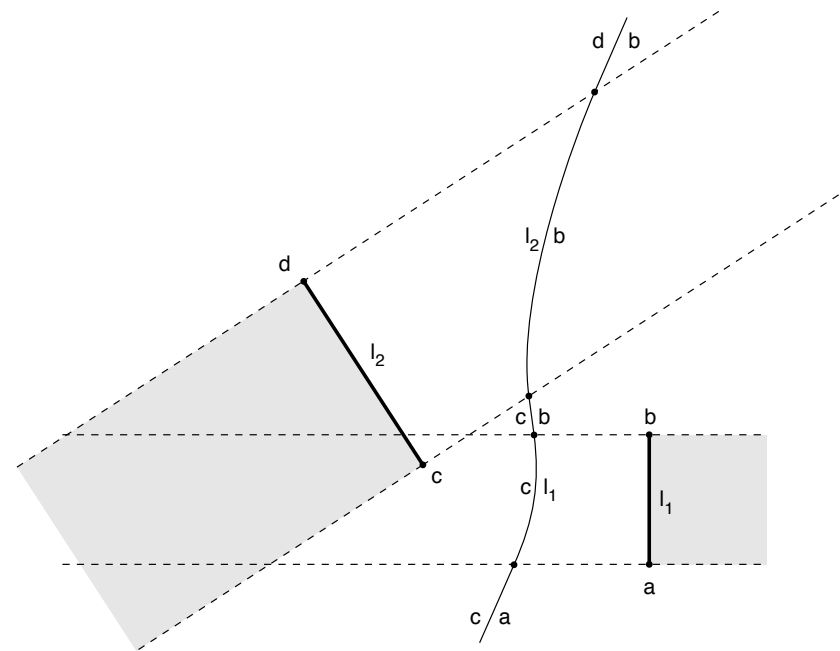


Bisektor von Segmenten

Lemma 5.24 Der Bisektor von zwei disjunkten Liniensegmenten l_1 und l_2 ist eine Kurve aus Parabelstücken, Liniensegmenten und zwei Halbgeraden.

Verantwortungsbereiche der Streifen, Lage zueinander

1. l_1 Punkt, l_2 Punkt:
Bisektorstück Gerade
2. l_1 Segment, l_2 Punkt:
Bisektorstück Parabel

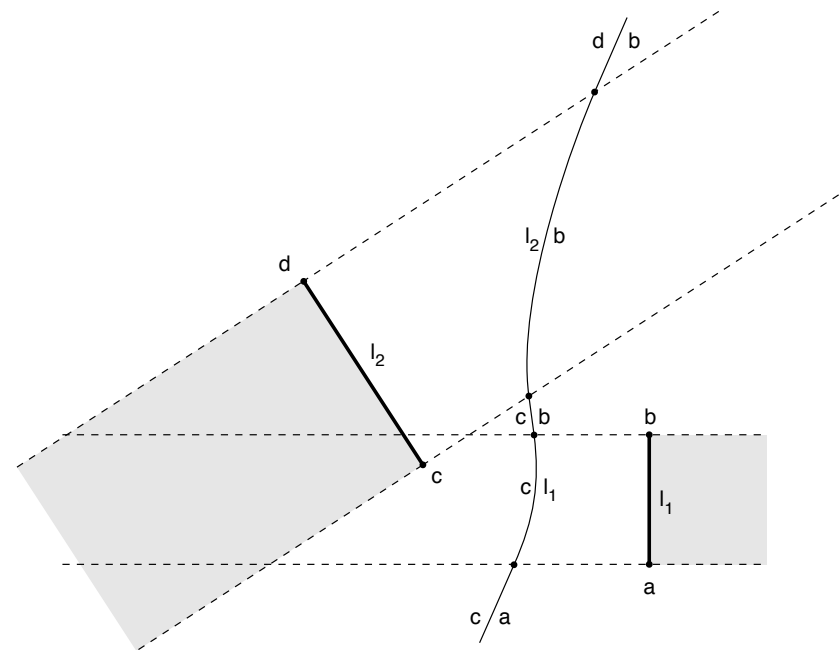


Bisektor von Segmenten

Lemma 5.24 Der Bisektor von zwei disjunkten Liniensegmenten l_1 und l_2 ist eine Kurve aus Parabelstücken, Liniensegmenten und zwei Halbgeraden.

Verantwortungsbereiche der Streifen, Lage zueinander

1. l_1 Punkt, l_2 Punkt:
Bisektorstück Gerade
2. l_1 Segment, l_2 Punkt:
Bisektorstück Parabel
3. l_1 Segment, l_2 Segment:
Bisektorstück Gerade



Regionen sind sternförmig

Regionen sind sternförmig

Menge von Liniensegmenten S , Voronoi-Region von einem Liniensegment $VR(l, S)$!

Regionen sind sternförmig

Menge von Liniensegmenten S , Voronoi-Region von einem Liniensegment $VR(l, S)$!

Lemma 5.25 Sei S eine Menge von Liniensegmenten und $l \in S$. Für jeden Punkt x in der Voronoi-Region $VR(l, S)$ gilt: Das Liniensegment xy_x zwischen x und dem Punkt $y_x \in l$ der am nächsten zu x liegt, liegt in $VR(l, S)$.

Regionen sind sternförmig

Menge von Liniensegmenten S , Voronoi-Region von einem Liniensegment $VR(l, S)$!

Lemma 5.25 Sei S eine Menge von Liniensegmenten und $l \in S$. Für jeden Punkt x in der Voronoi-Region $VR(l, S)$ gilt: Das Liniensegment xy_x zwischen x und dem Punkt $y_x \in l$ der am nächsten zu x liegt, liegt in $VR(l, S)$.

Beweis:

Regionen sind sternförmig

Menge von Liniensegmenten S , Voronoi-Region von einem Liniensegment $VR(l, S)$!

Lemma 5.25 Sei S eine Menge von Liniensegmenten und $l \in S$. Für jeden Punkt x in der Voronoi-Region $VR(l, S)$ gilt: Das Liniensegment xy_x zwischen x und dem Punkt $y_x \in l$ der am nächsten zu x liegt, liegt in $VR(l, S)$.

Beweis: Widerspruch!

Regionen zusammenhängend

Regionen zusammenhängend

Korollar 5.26 Die Voronoi-Regionen von Liniensegmenten sind zusammenhängend.

Regionen zusammenhängend

Korollar 5.26 Die Voronoi-Regionen von Liniensegmenten sind zusammenhängend.

Beweis:

Regionen zusammenhängend

Korollar 5.26 Die Voronoi-Regionen von Liniensegmenten sind zusammenhängend.

Beweis: Alle Punkte der Region haben *Blickkontakt* zu l !

Bisektor: Maximal 7 Stücke!

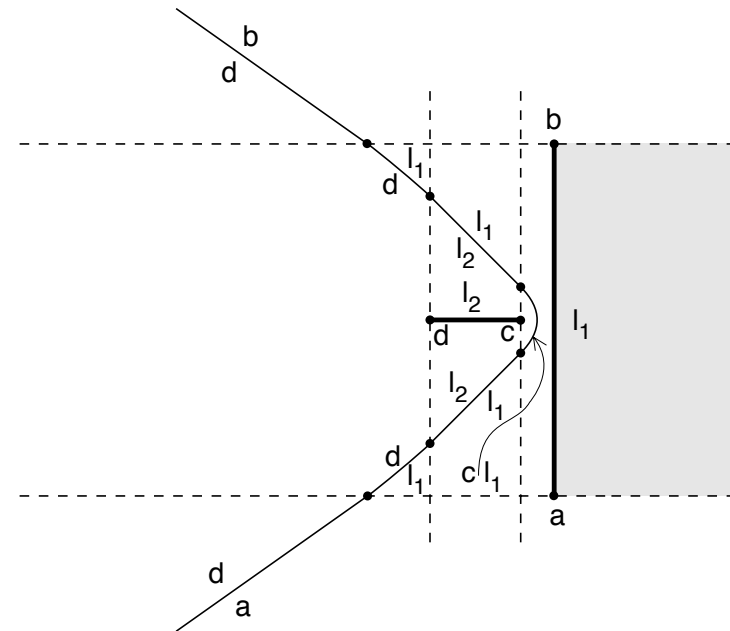
Bisektor: Maximal 7 Stücke!

Lemma 5.27 Der Bisektor von zwei disjunkten Liniensegmenten l_1 und l_2 ist eine Kurve aus Parabelstücken, Liniensegmenten und zwei Halbgeraden und besteht aus maximal 7 Stücken.

Bisektor: Maximal 7 Stücke!

Lemma 5.27 Der Bisektor von zwei disjunkten Liniensegmenten l_1 und l_2 ist eine Kurve aus Parabelstücken, Liniensegmenten und zwei Halbgeraden und besteht aus maximal 7 Stücken.

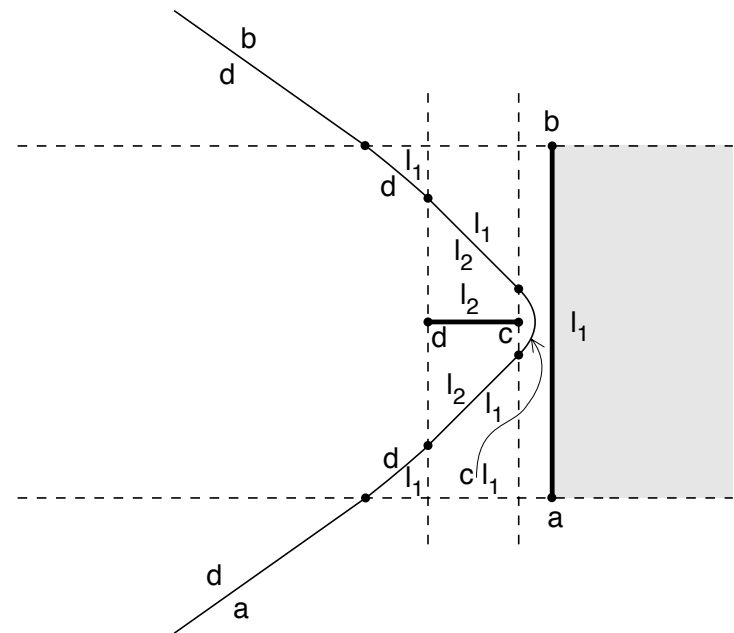
- Halbstreifen, insgesamt 8



Bisektor: Maximal 7 Stücke!

Lemma 5.27 Der Bisektor von zwei disjunkten Liniensegmenten l_1 und l_2 ist eine Kurve aus Parabelstücken, Liniensegmenten und zwei Halbgeraden und besteht aus maximal 7 Stücken.

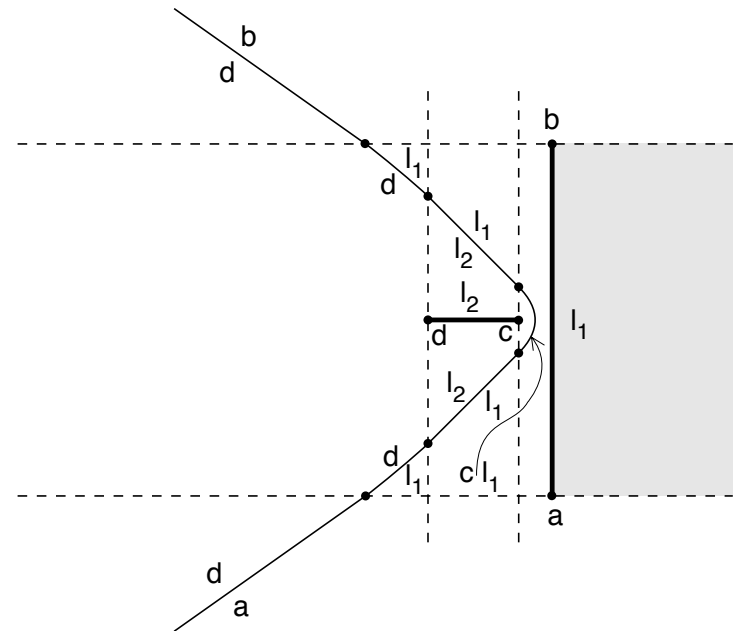
- Halbstreifen, insgesamt 8
- Einmal betreten, einmal verlassen, monoton



Bisektor: Maximal 7 Stücke!

Lemma 5.27 Der Bisektor von zwei disjunkten Liniensegmenten l_1 und l_2 ist eine Kurve aus Parabelstücken, Liniensegmenten und zwei Halbgeraden und besteht aus maximal 7 Stücken.

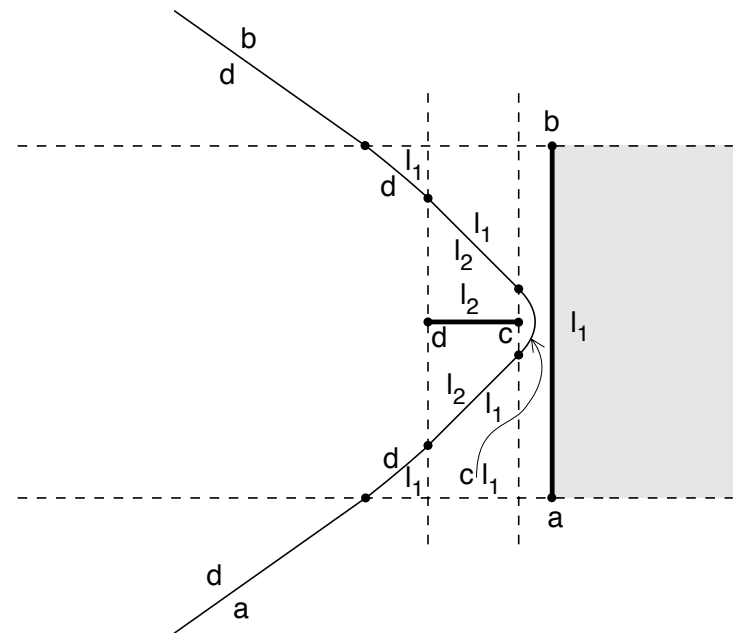
- Halbstreifen, insgesamt 8
- Einmal betreten, einmal verlassen, monoton
- Mind. Segment l_1 liegt auf konvex. Hülle



Bisektor: Maximal 7 Stücke!

Lemma 5.27 Der Bisektor von zwei disjunkten Liniensegmenten l_1 und l_2 ist eine Kurve aus Parabelstücken, Liniensegmenten und zwei Halbgeraden und besteht aus maximal 7 Stücken.

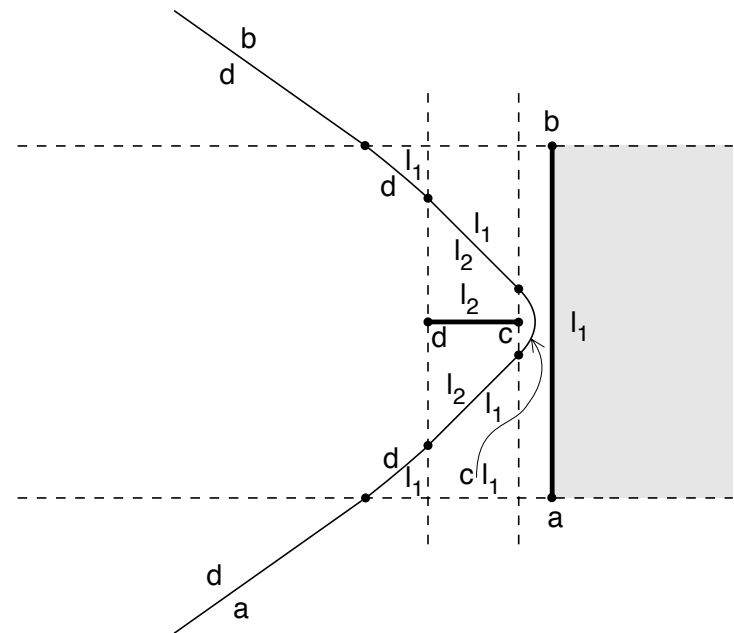
- Halbstreifen, insgesamt 8
- Einmal betreten, einmal verlassen, monoton
- Mind. Segment l_1 liegt auf konvex. Hülle
- Bisektor betritt sukzessive 3 Streifen von l_1



Bisektor: Maximal 7 Stücke!

Lemma 5.27 Der Bisektor von zwei disjunkten Liniensegmenten l_1 und l_2 ist eine Kurve aus Parabelstücken, Liniensegmenten und zwei Halbgeraden und besteht aus maximal 7 Stücken.

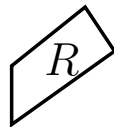
- Halbstreifen, insgesamt 8
- Einmal betreten, einmal verlassen, monoton
- Mind. Segment l_1 liegt auf konvex. Hülle
- Bisektor betritt sukzessive 3 Streifen von l_1
- max. 6 Kanten überqueren



Anwendung Bahnplanung: Kreisförmiger Agent

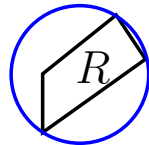
Anwendung Bahnplanung: Kreisförmiger Agent

- Verwende kleinsten Kreis um Roboter



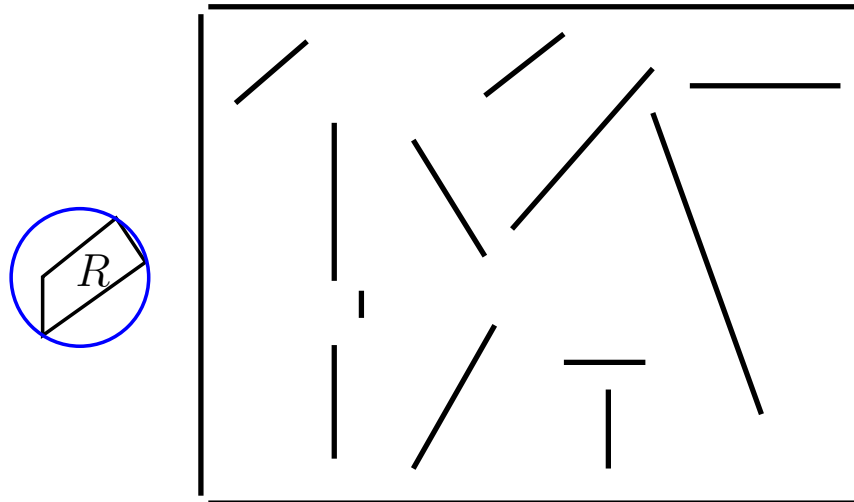
Anwendung Bahnplanung: Kreisförmiger Agent

- Verwende kleinsten Kreis um Roboter
- Voronoi Diagramm der Segmente der Hindernisse



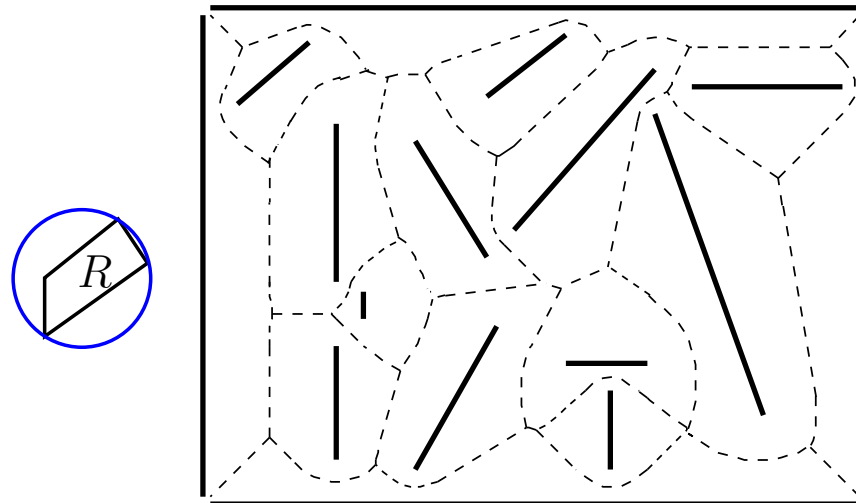
Anwendung Bahnplanung: Kreisförmiger Agent

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



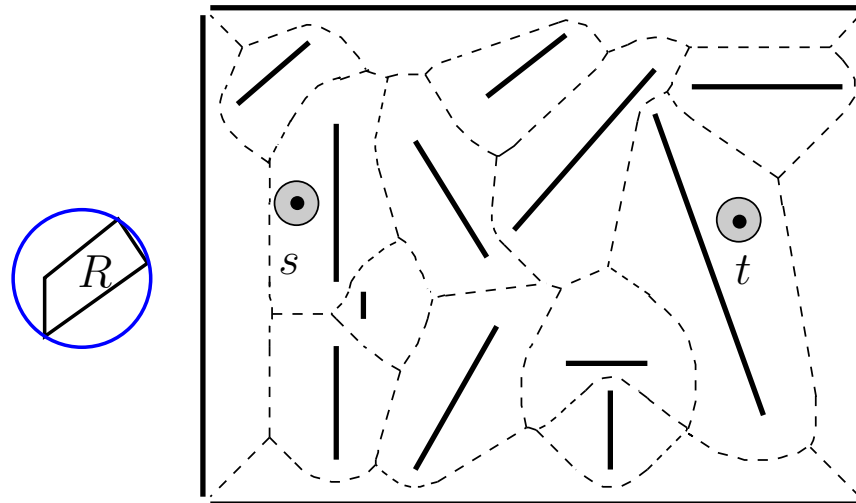
Anwendung Bahnplanung: Kreisförmiger Agent

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



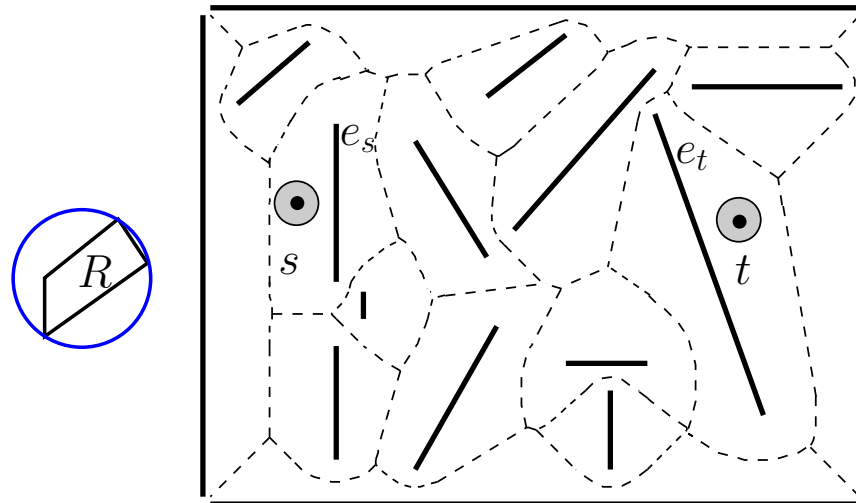
Anwendung Bahnplanung: Kreisförmiger Agent

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



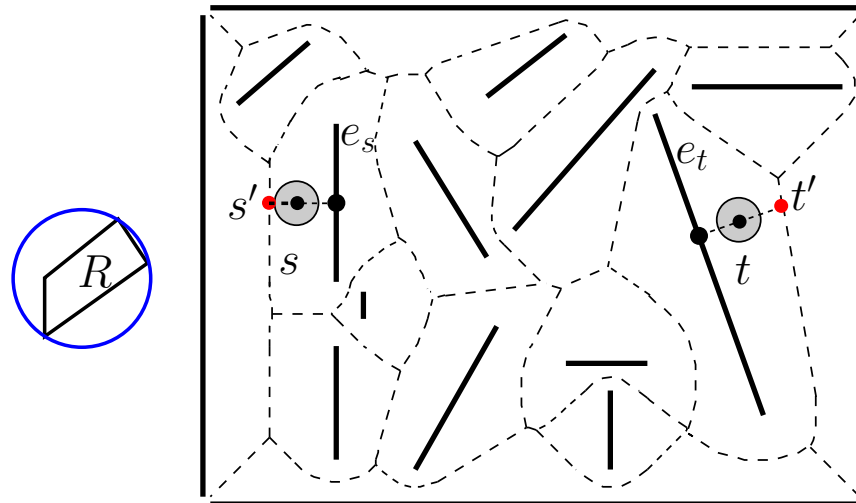
Anwendung Bahnplanung: Kreisförmiger Agent

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



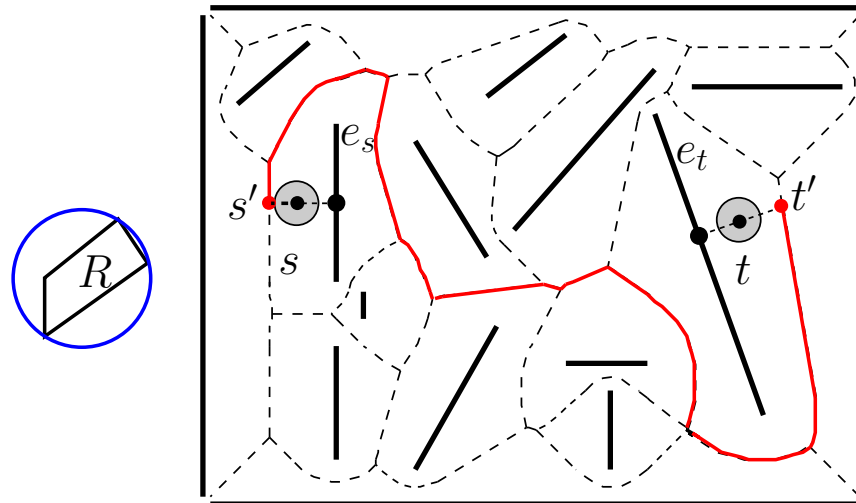
Anwendung Bahnplanung: Kreisförmiger Agent

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



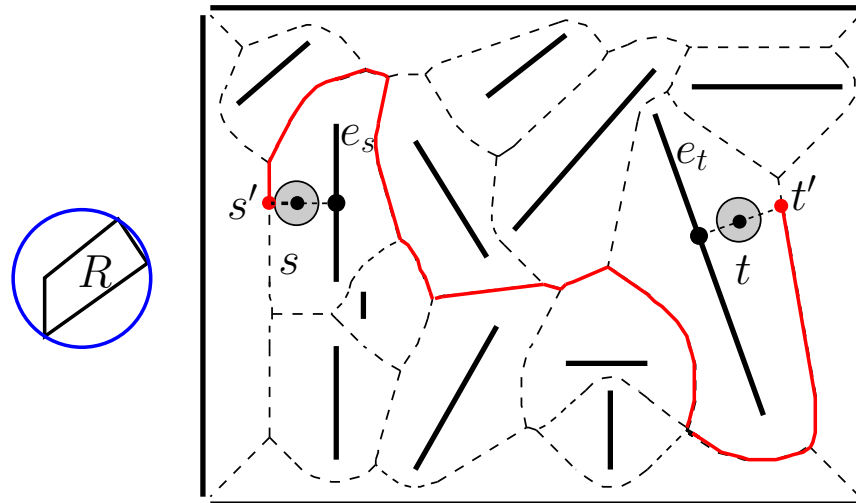
Anwendung Bahnplanung: Kreisförmiger Agent

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

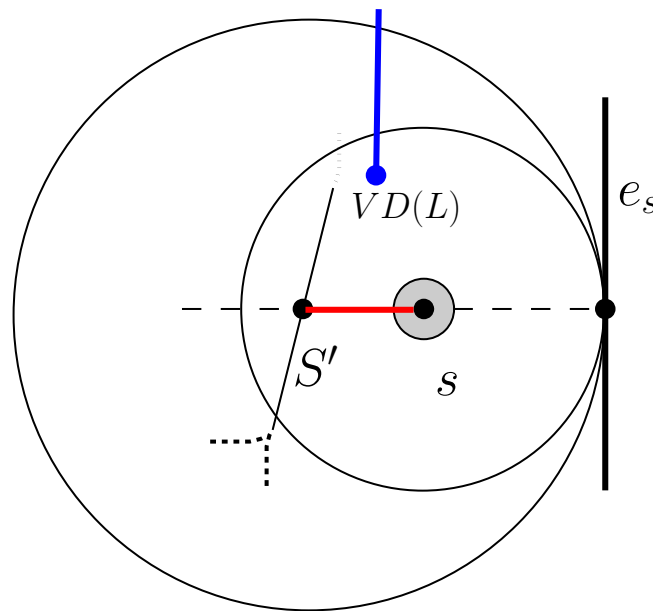


Anwendung Bahnplanung: Kreisförmiger Agent

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

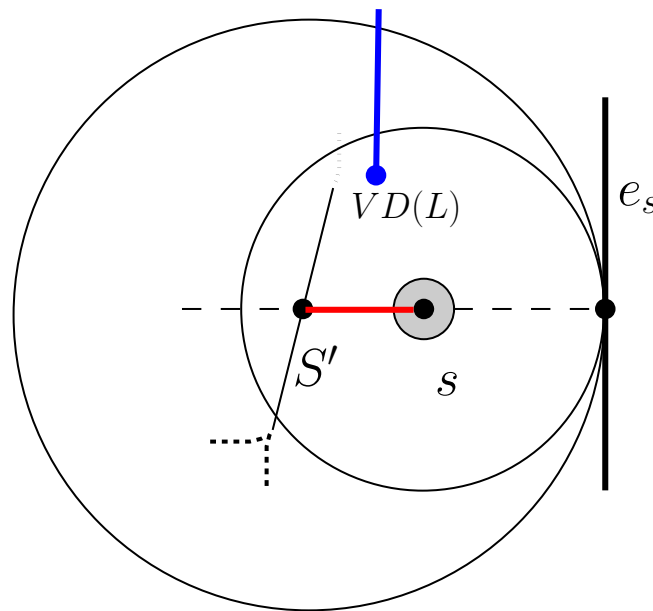


Start s' kann stets angelaufen werden



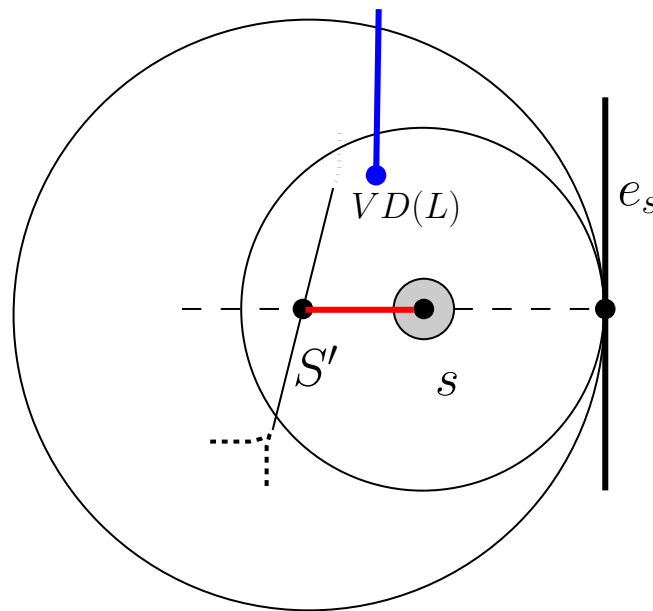
Start s' kann stets angelaufen werden

- s in Region von e_s ,



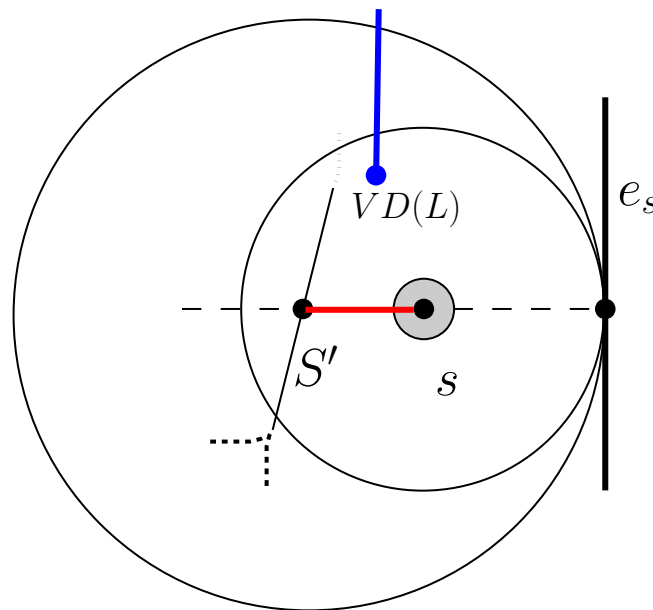
Start s' kann stets angelaufen werden

- s in Region von e_s , Kreis frei



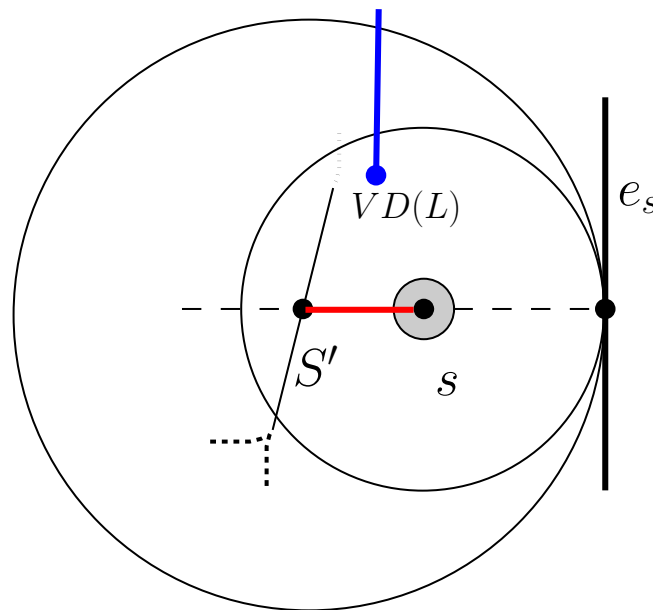
Start s' kann stets angelaufen werden

- s in Region von e_s , Kreis frei
- Kürzester Weg zu e_s , Strahl Richtung Bisektor



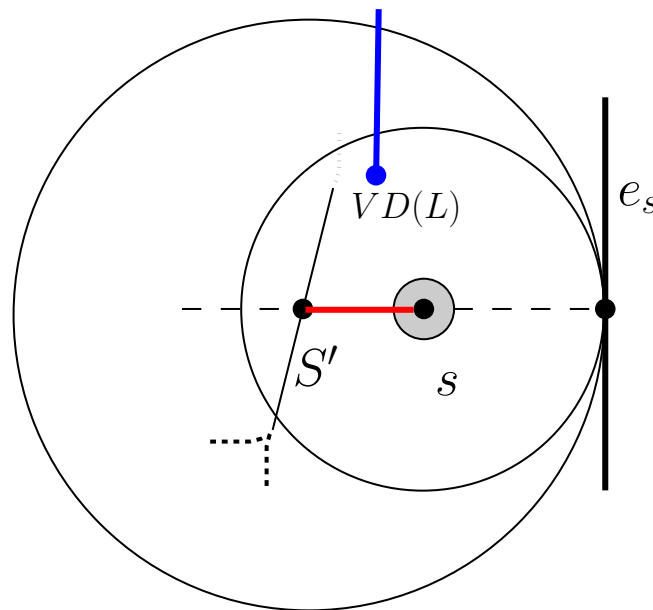
Start s' kann stets angelaufen werden

- s in Region von e_s , Kreis frei
- Kürzester Weg zu e_s , Strahl Richtung Bisektor
- Trifft Bisektor bei S' ,



Start s' kann stets angelaufen werden

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



Bewegungsplanung von Agenten

Bewegungsplanung von Agenten

- Historie: Entkommen aus dem Labyrinth

Bewegungsplanung von Agenten

- Historie: Entkommen aus dem Labyrinth
- Effizientmaße: Länge des Weges/Rechenzeit

Bewegungsplanung von Agenten

- Historie: Entkommen aus dem Labyrinth
- Effizientmaße: Länge des Weges/Rechenzeit
- Offline: Alle Informationen sind vorhanden

Bewegungsplanung von Agenten

- Historie: Entkommen aus dem Labyrinth
- Effizientmaße: Länge des Weges/Rechenzeit
- Offline: Alle Informationen sind vorhanden
- Online: Nur lokale Informationen: Sicht/Tastsensor

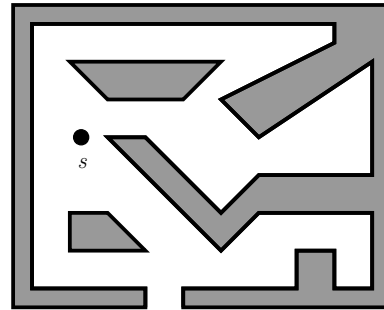
Bewegungsplanung von Agenten

- Historie: Entkommen aus dem Labyrinth
- Effizientmaße: Länge des Weges/Rechenzeit
- Offline: Alle Informationen sind vorhanden
- Online: Nur lokale Informationen: Sicht/Tastsensor
- Modell: Karte aufbauen, Umgebung merken, kein Speicher

Bewegungsplanung von Agenten

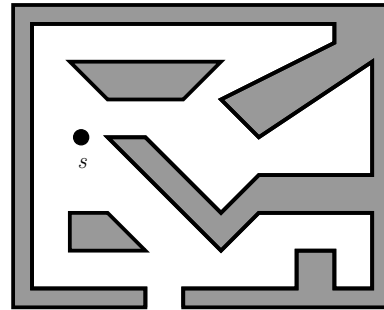
- Historie: Entkommen aus dem Labyrinth
- Effizientmaße: Länge des Weges/Rechenzeit
- Offline: Alle Informationen sind vorhanden
- Online: Nur lokale Informationen: Sicht/Tastsensor
- Modell: Karte aufbauen, Umgebung merken, kein Speicher
- Unvollständige Information

Polygonale Umgebungen



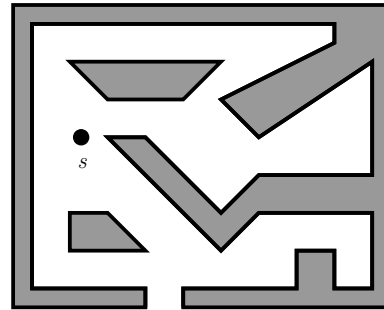
Polygonale Umgebungen

- Umgebung: Menge von disjunkten einfachen Polygonen



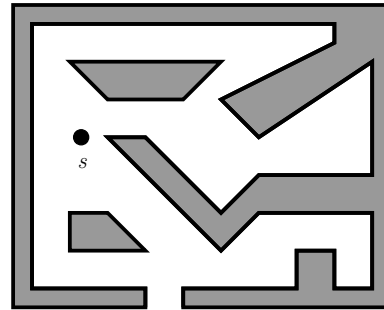
Polygonale Umgebungen

- Umgebung: Menge von disjunkten einfachen Polygonen
- Ein Randpolygon



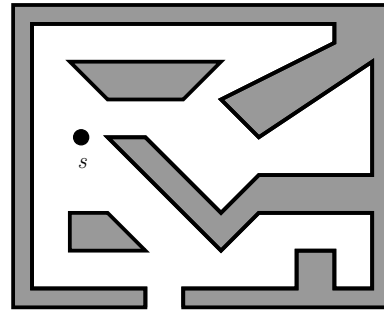
Polygonale Umgebungen

- Umgebung: Menge von disjunkten einfachen Polygonen
- Ein Randpolygon
- Verschiedene Aufgaben: Suchen eines Zielpunktes/Entkommen aus Labyrinth



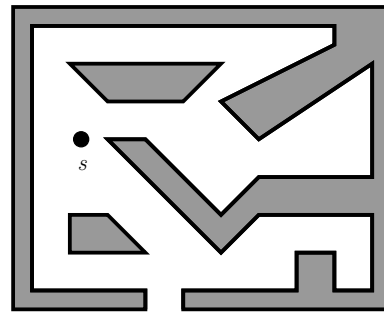
Polygonale Umgebungen

- Umgebung: Menge von disjunkten einfachen Polygonen
- Ein Randpolygon
- Verschiedene Aufgaben: Suchen eines Zielpunktes/Entkommen aus Labyrinth
- Verschiedene Sensormodelle

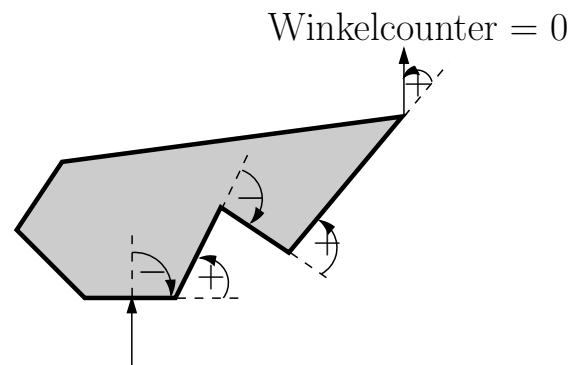


Polygonale Umgebungen

- Umgebung: Menge von disjunkten einfachen Polygonen
- Ein Randpolygon
- Verschiedene Aufgaben: Suchen eines Zielpunktes/Entkommen aus Labyrinth
- Verschiedene Sensormodelle
- Zunächst: Entkommen aus dem Labyrinth

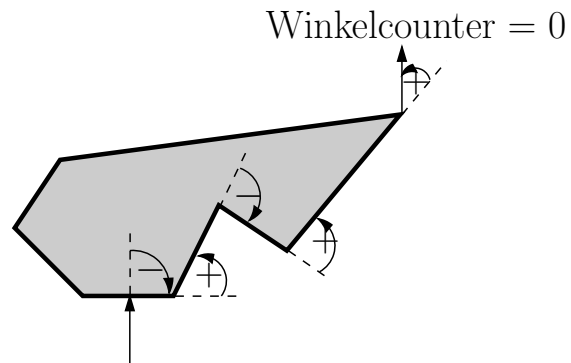


Entkommen aus dem Labyrinth: Modell



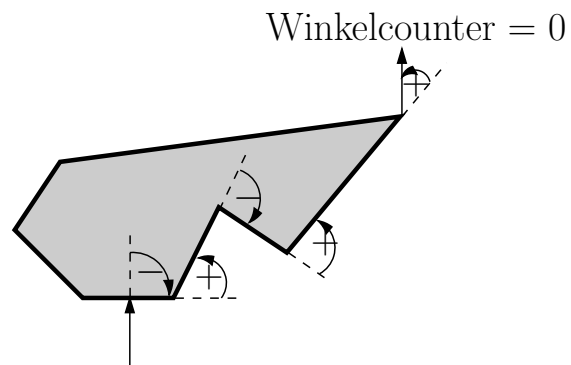
Entkommen aus dem Labyrinth: Modell

- Punktförmiger Agent/kreisförmiger Agent



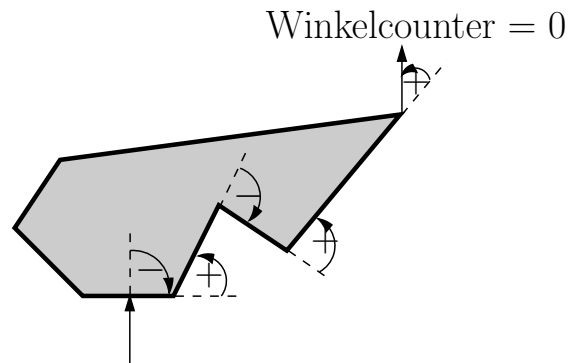
Entkommen aus dem Labyrinth: Modell

- Punktförmiger Agent/kreisförmiger Agent
- Touch Sensor



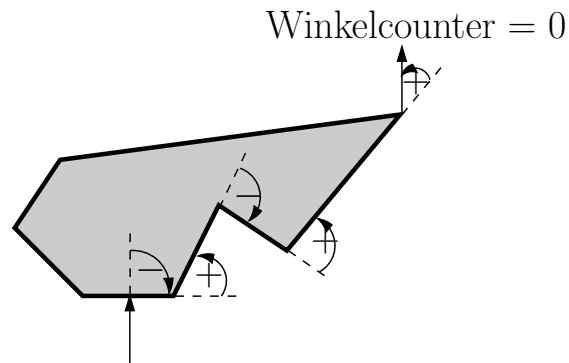
Entkommen aus dem Labyrinth: Modell

- Punktförmiger Agent/kreisförmiger Agent
- Touch Sensor
- Folge einer Wand



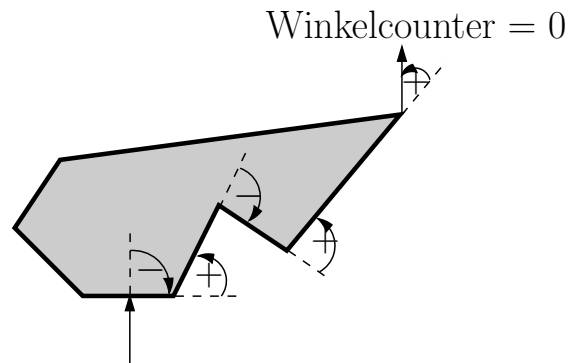
Entkommen aus dem Labyrinth: Modell

- Punktförmiger Agent/kreisförmiger Agent
- Touch Sensor
- Folge einer Wand
- Folge einer Richtung (exakt)



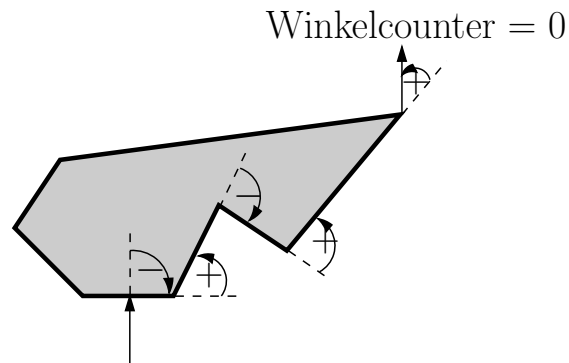
Entkommen aus dem Labyrinth: Modell

- Punktförmiger Agent/kreisförmiger Agent
- Touch Sensor
- Folge einer Wand
- Folge einer Richtung (exakt)
- Drehwinkel-Zähler

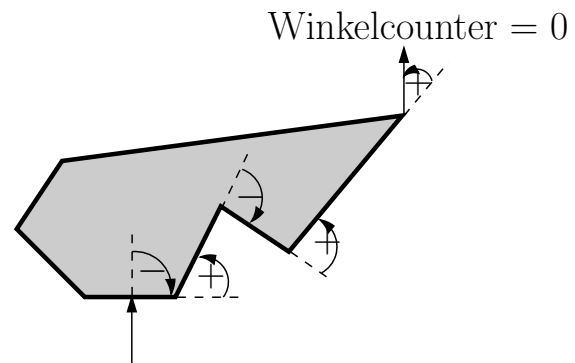


Entkommen aus dem Labyrinth: Modell

- Punktförmiger Agent/kreisförmiger Agent
- Touch Sensor
- Folge einer Wand
- Folge einer Richtung (exakt)
- Drehwinkel-Zähler
- Keinen weiteren Speicher

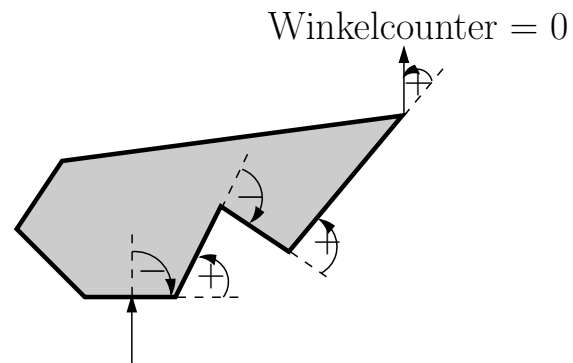


Pledge Algorithmus



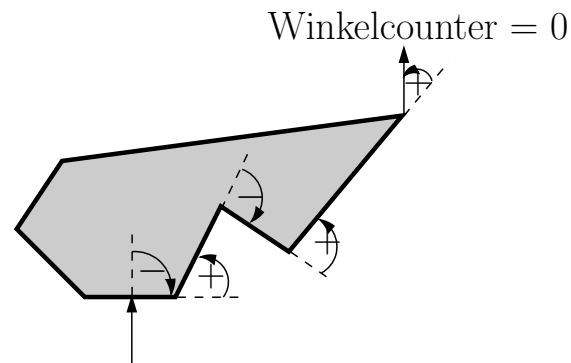
Pledge Algorithmus

1. Wähle Winkel φ und drehe den Roboter in diese Richtung.



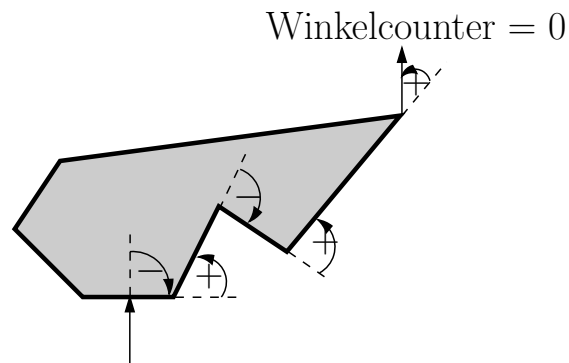
Pledge Algorithmus

1. Wähle Winkel φ und drehe den Roboter in diese Richtung.
2. Gehe in Richtung φ , bis der Roboter ein Hindernis erreicht.



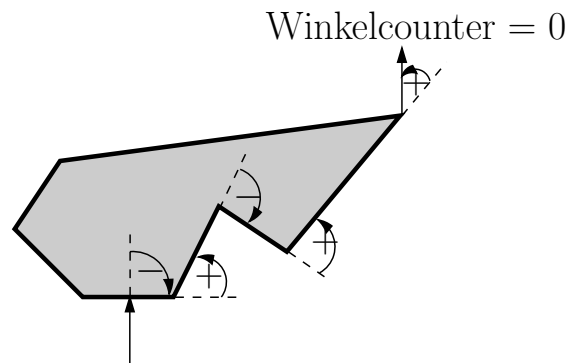
Pledge Algorithmus

1. Wähle Winkel φ und drehe den Roboter in diese Richtung.
2. Gehe in Richtung φ , bis der Roboter ein Hindernis erreicht.
3. Drehe nach rechts und halte den Kontakt mit der Wand an der linken Seite des Roboters.



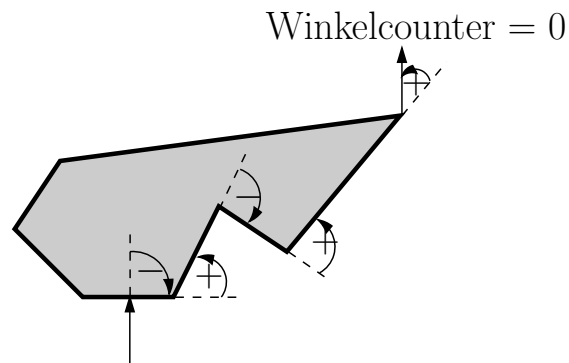
Pledge Algorithmus

1. Wähle Winkel φ und drehe den Roboter in diese Richtung.
2. Gehe in Richtung φ , bis der Roboter ein Hindernis erreicht.
3. Drehe nach rechts und halte den Kontakt mit der Wand an der linken Seite des Roboters.
4. Folge der Wand und addiere dabei die Drehwinkel, bis der **totale Drehwinkel** Null ist, dann GOTO (2).



Pledge Algorithmus

1. Wähle Winkel φ und drehe den Roboter in diese Richtung.
2. Gehe in Richtung φ , bis der Roboter ein Hindernis erreicht.
3. Drehe nach rechts und halte den Kontakt mit der Wand an der linken Seite des Roboters.
4. Folge der Wand und addiere dabei die Drehwinkel, bis der **totale Drehwinkel** Null ist, dann GOTO (2).



Beweis Korrektheit: Winkelzähler nicht positiv

Beweis Korrektheit: Winkelzähler nicht positiv

Lemma 7.2 Der Winkelzähler im Pledge-Algorithmus nimmt niemals einen positiven Wert an.

Beweis Korrektheit: Winkelzähler nicht positiv

Lemma 7.2 Der Winkelzähler im Pledge-Algorithmus nimmt niemals einen positiven Wert an.

Beweis:

Beweis Korrektheit: Winkelzähler nicht positiv

Lemma 7.2 Der Winkelzähler im Pledge-Algorithmus nimmt niemals einen positiven Wert an.

Beweis:

- Zu Beginn Null

Beweis Korrektheit: Winkelzähler nicht positiv

Lemma 7.2 Der Winkelzähler im Pledge-Algorithmus nimmt niemals einen positiven Wert an.

Beweis:

- Zu Beginn Null
- Null bei Verlassen des Hindernisses

Beweis Korrektheit: Winkelzähler nicht positiv

Lemma 7.2 Der Winkelzähler im Pledge-Algorithmus nimmt niemals einen positiven Wert an.

Beweis:

- Zu Beginn Null
- Null bei Verlassen des Hindernisses
- Beim Auftreffen Rechtsdrehung \Rightarrow negativ

Beweis Korrektheit: Winkelzähler nicht positiv

Lemma 7.2 Der Winkelzähler im Pledge-Algorithmus nimmt niemals einen positiven Wert an.

Beweis:

- Zu Beginn Null
- Null bei Verlassen des Hindernisses
- Beim Auftreffen Rechtsdrehung \Rightarrow negativ
- Stetige Änderung: Null \Rightarrow Weg frei