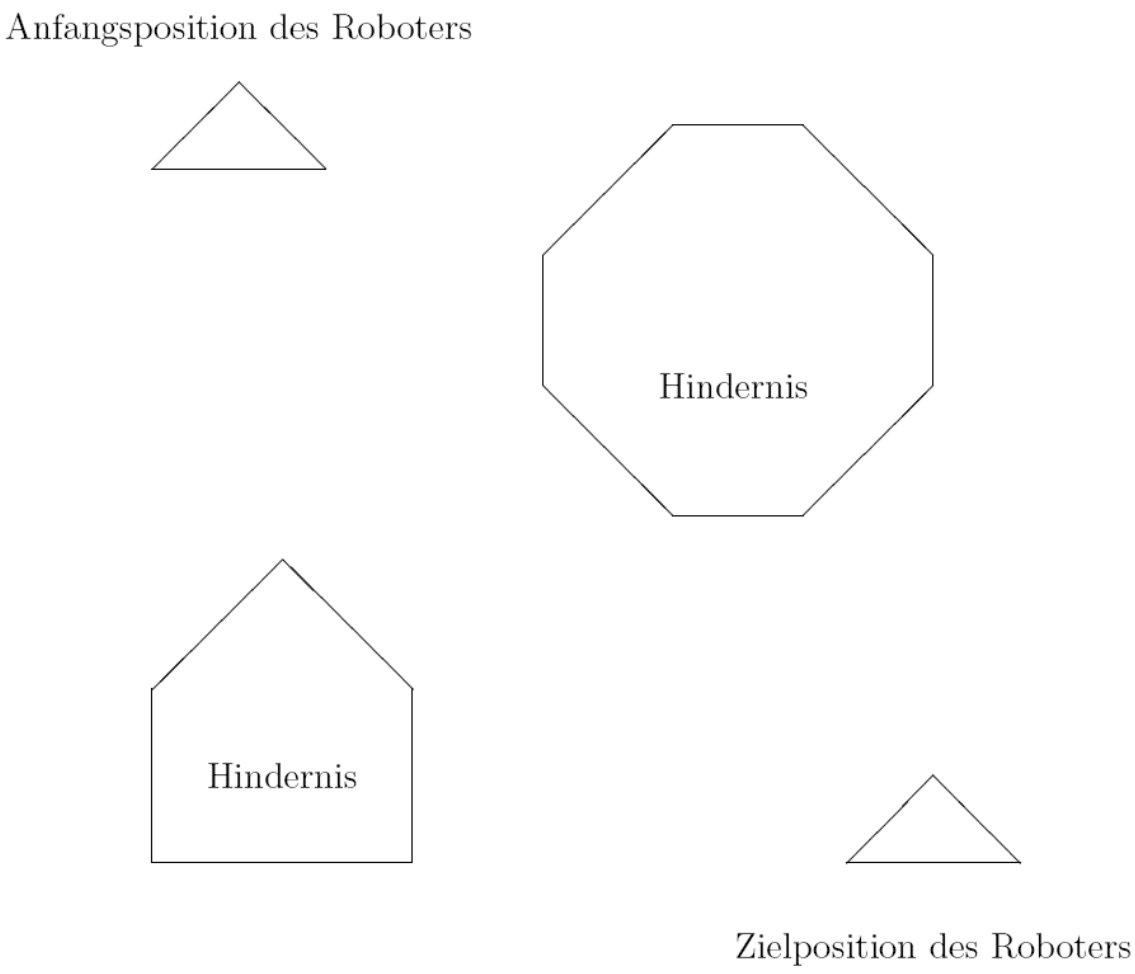
10 Pfadplanung

10.1 Geometrische Wegplanung im Konfigurationsraum

Vorausetzungen

- Roboter bewegt sich in der Ebene, ohne sich zu drehen
- Hindernisse sind konvexe Polygone

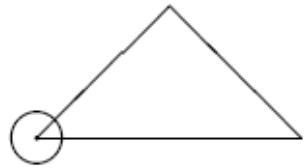
Beispiel


Anfangsposition des Roboters
Hindernis
Hindernis
Zielposition des Roboters

Grundgedanke

- Problem wird in eine andere vereinfachte Darstellung transformiert
  - Roboter wird Punkt
  - Hindernisse werden vergrößert (virtuelle Hindernisse)
  - Konfigurationsraum
- Dort kann es leichter gelöst werden
- Lösung wird zurücktransformiert

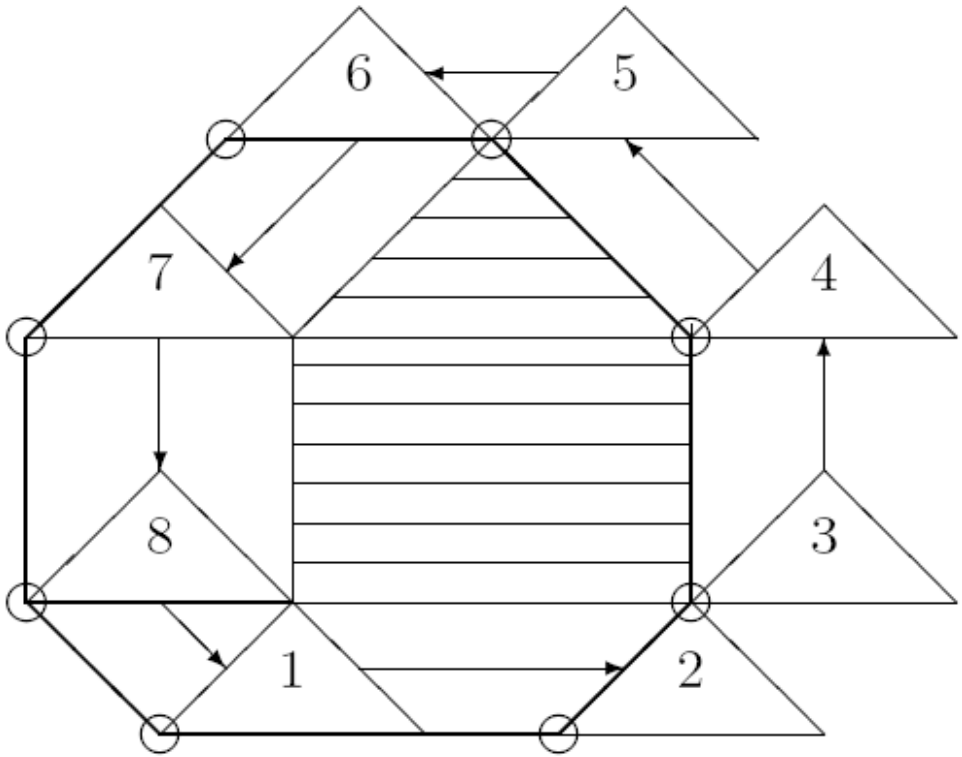
Roboter und Bezugspunkt



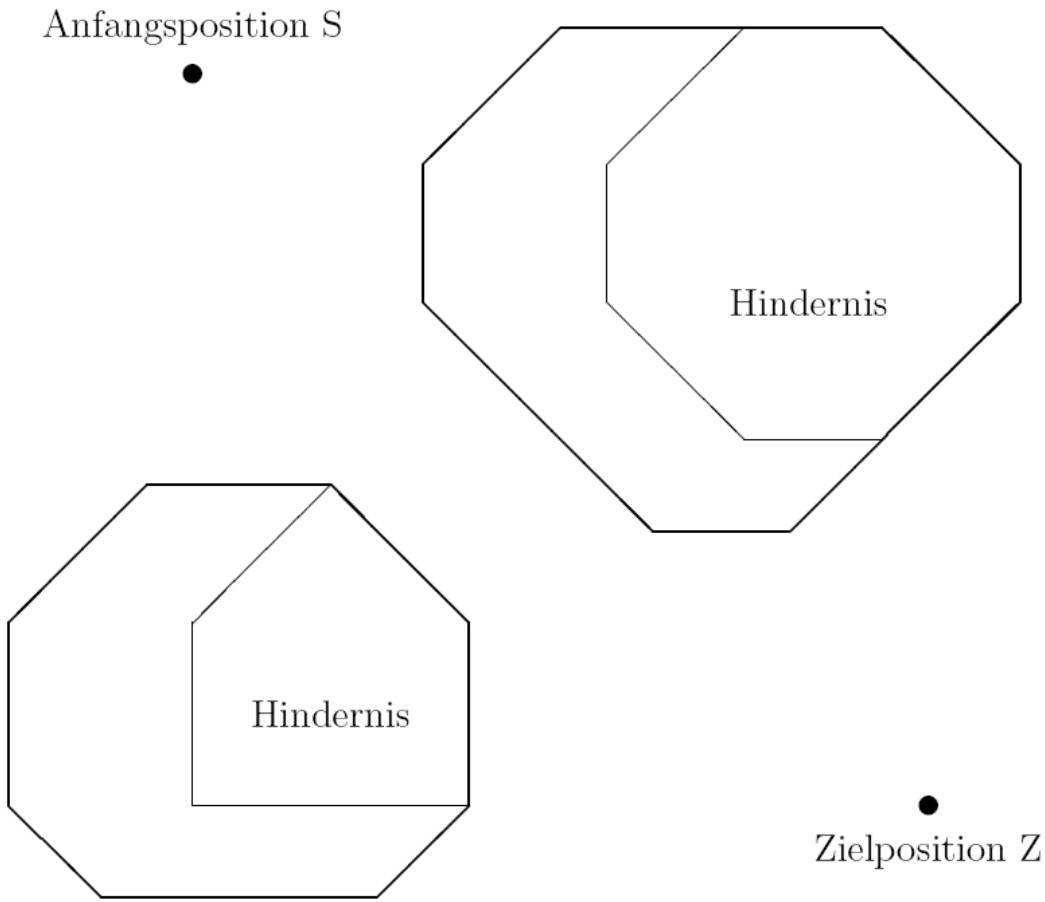
Erzeugung der virtuellen Hindernisse

- Man bewegt den Roboter um ein Hindernis
- Die Bahn des Bezugspunktes (Roboter) ergibt das virtuelle Hindernis
- Wenn der Bezugspunkt außerhalb des virtuellen Hindernisses bleibt, dann bewegt sich auch der gesamte Roboter außerhalb des realen Hindernisses

Hindernisse im Konfigurationsraum



Konfigurationsraum



Gesucht ist ein Weg von S nach Z

Suche im Konfigurationsraum

- Gesucht ist ein Weg von S nach Z
- Konstruktion des Sichtbarkeitsgraphen
- Innerhalb dieses Graphen wird ein (optimaler Weg) vom Startknoten S zum Zielknoten Z entlang der Kanten bestimmt
- z.B. mit A*
- durch Rücktransformation erhält man eine Lösung des ursprünglichen Bewegungsproblems

Sichtbarkeitsgraph

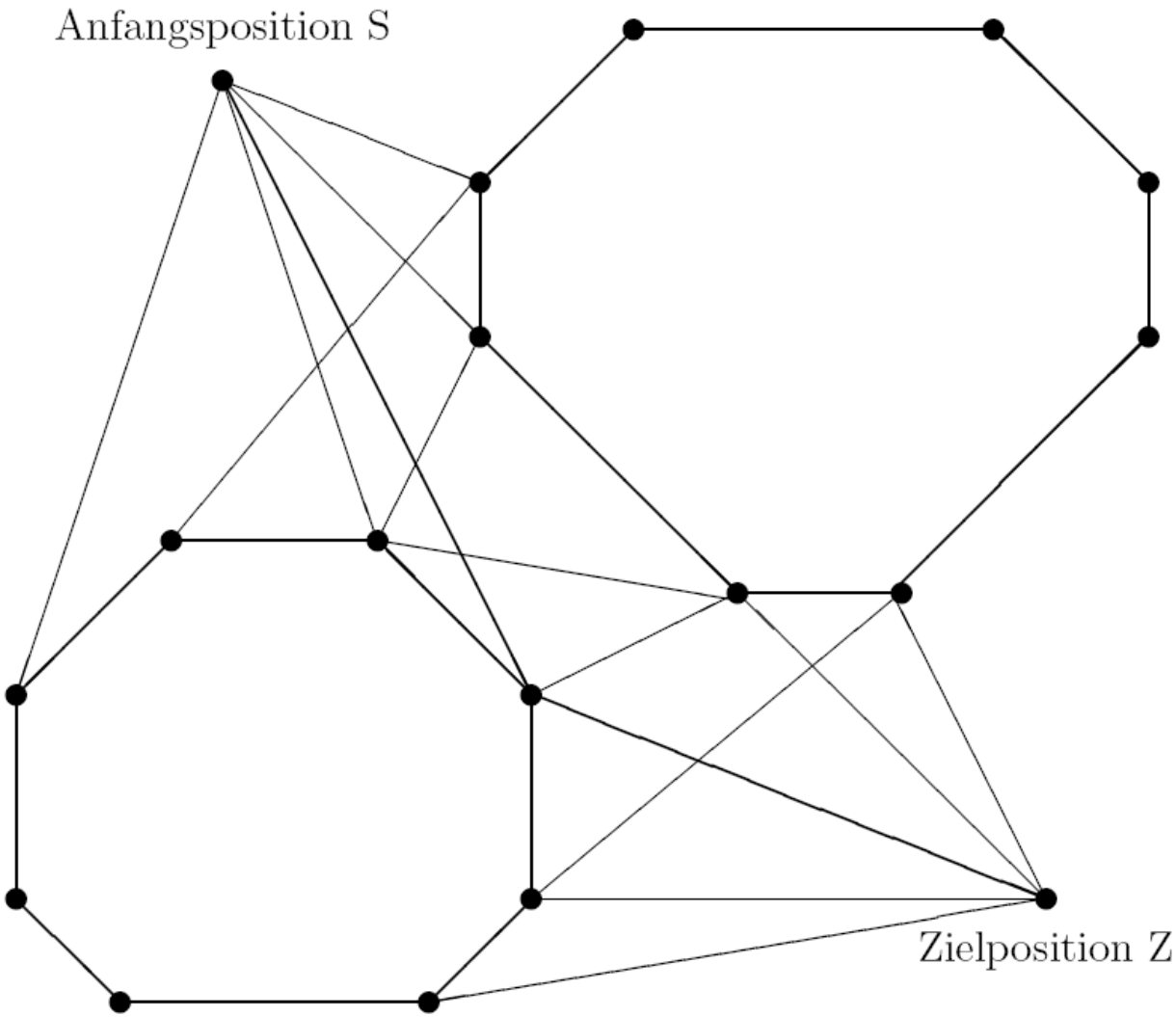
Knoten:

$$N = V \cup \{S, Z\}$$

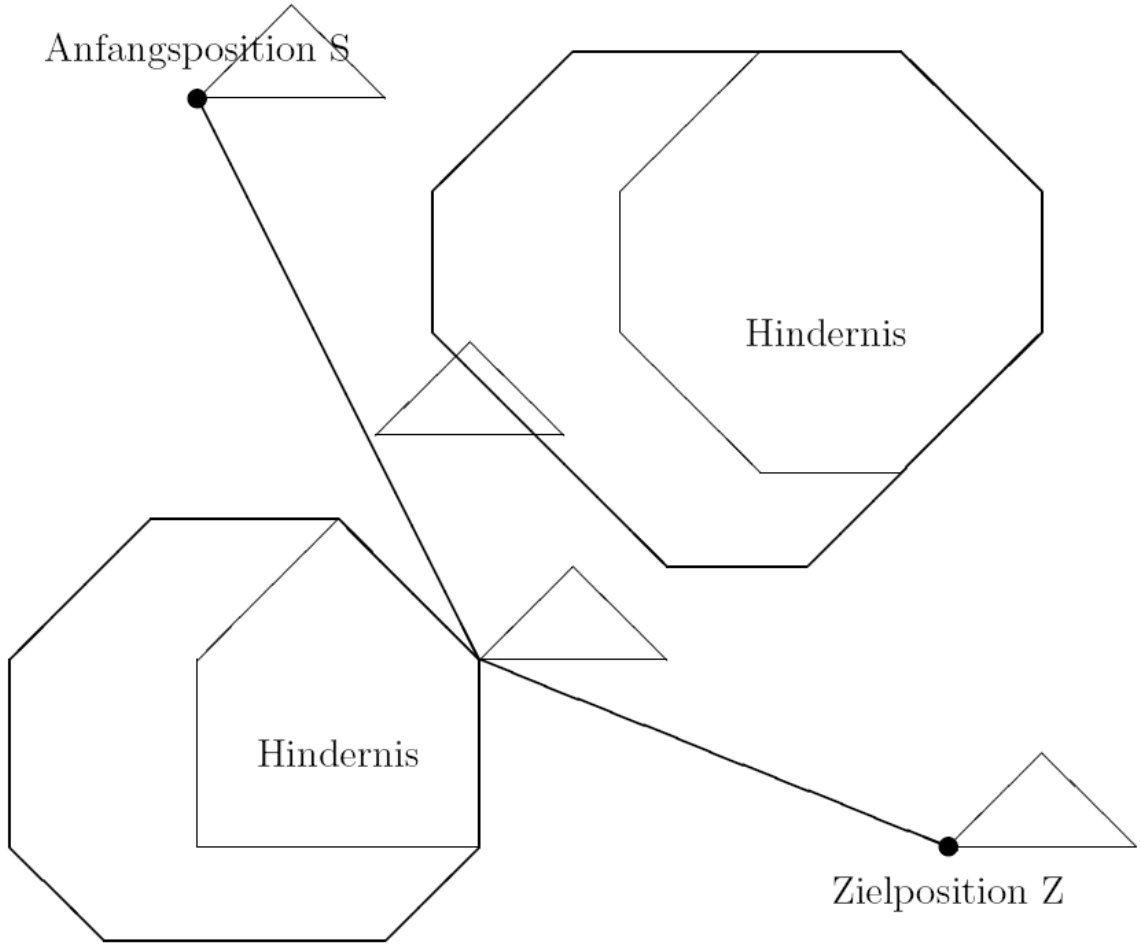
Ecken der virtuellen Hindernisse — Start — Ziel

Kanten: L – geradlinige Verbindungen der Knoten aus N, ohne dabei ein Hindernis zu überschneiden

Sichtbarkeitsgraph



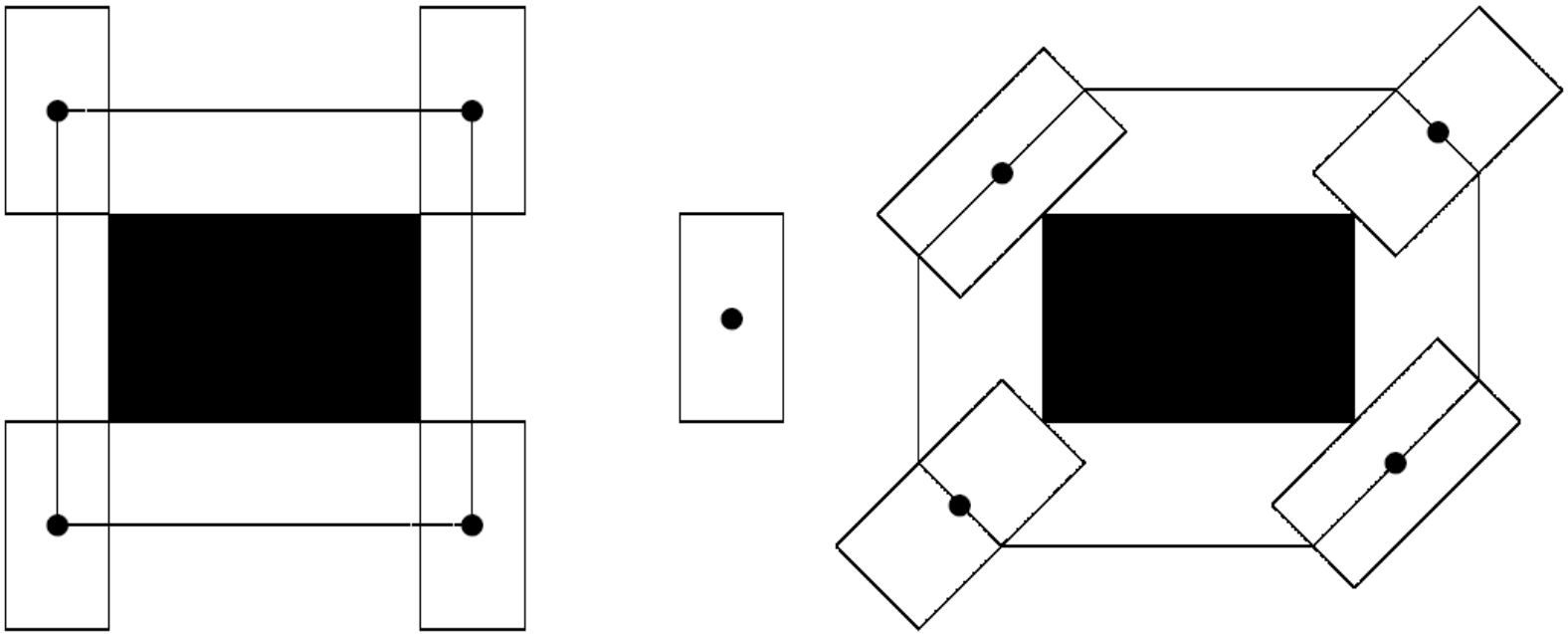
Lösung



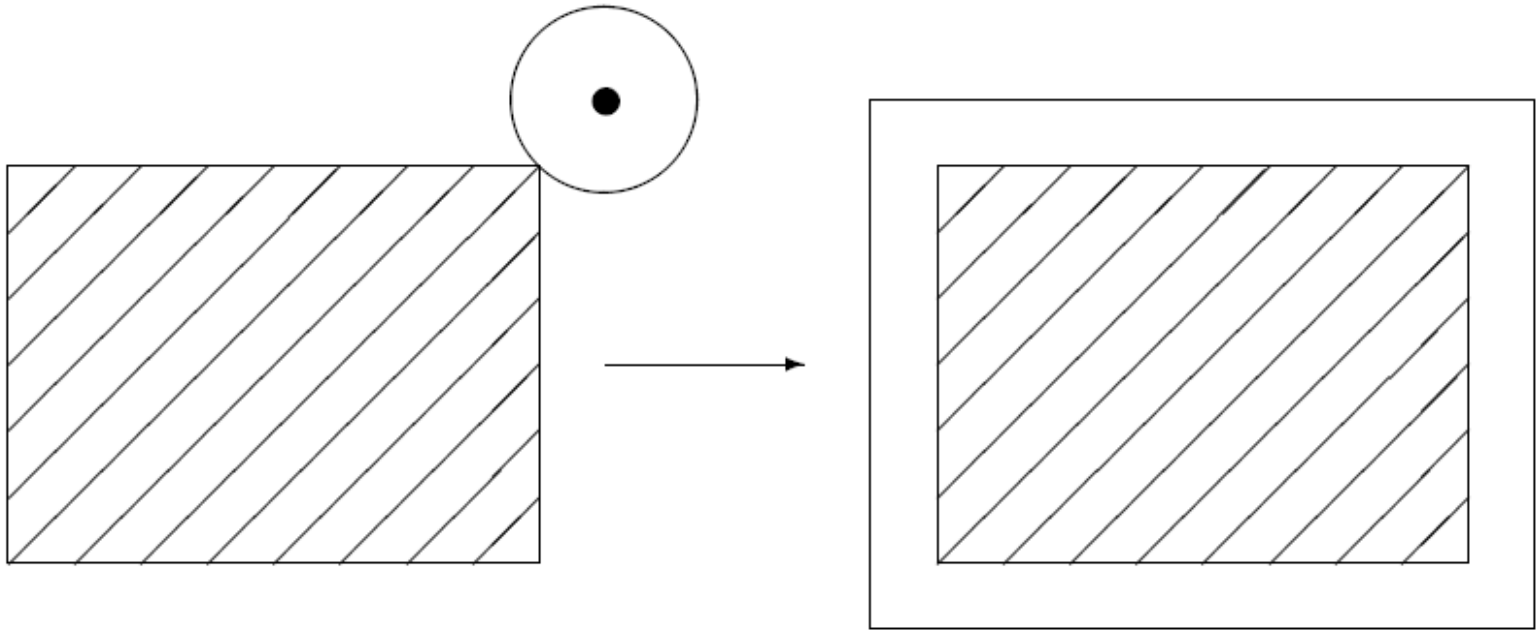
Bemerkungen

- Ein Nachteil dieses Verfahrens ist, dass die Bahn des Roboters an der Hindernisbergrenzung entlangläuft.
- Eine Verallgemeinerung auf 3 Dimensionen ist möglich, aber großer Suchgraph.
- Falls Drehungen zugelassen werden, so betrachtet man mehrere Konfigurationsräume (für jeden Drehwinkel) oder betrachtet den Roboter als Kreis.
- Ansonsten werden die Konfigurationsräume komplizierter, wenn beliebige Drehungen zugelassen werden.

Konfigurationsräume für verschiedene Drehwinkel



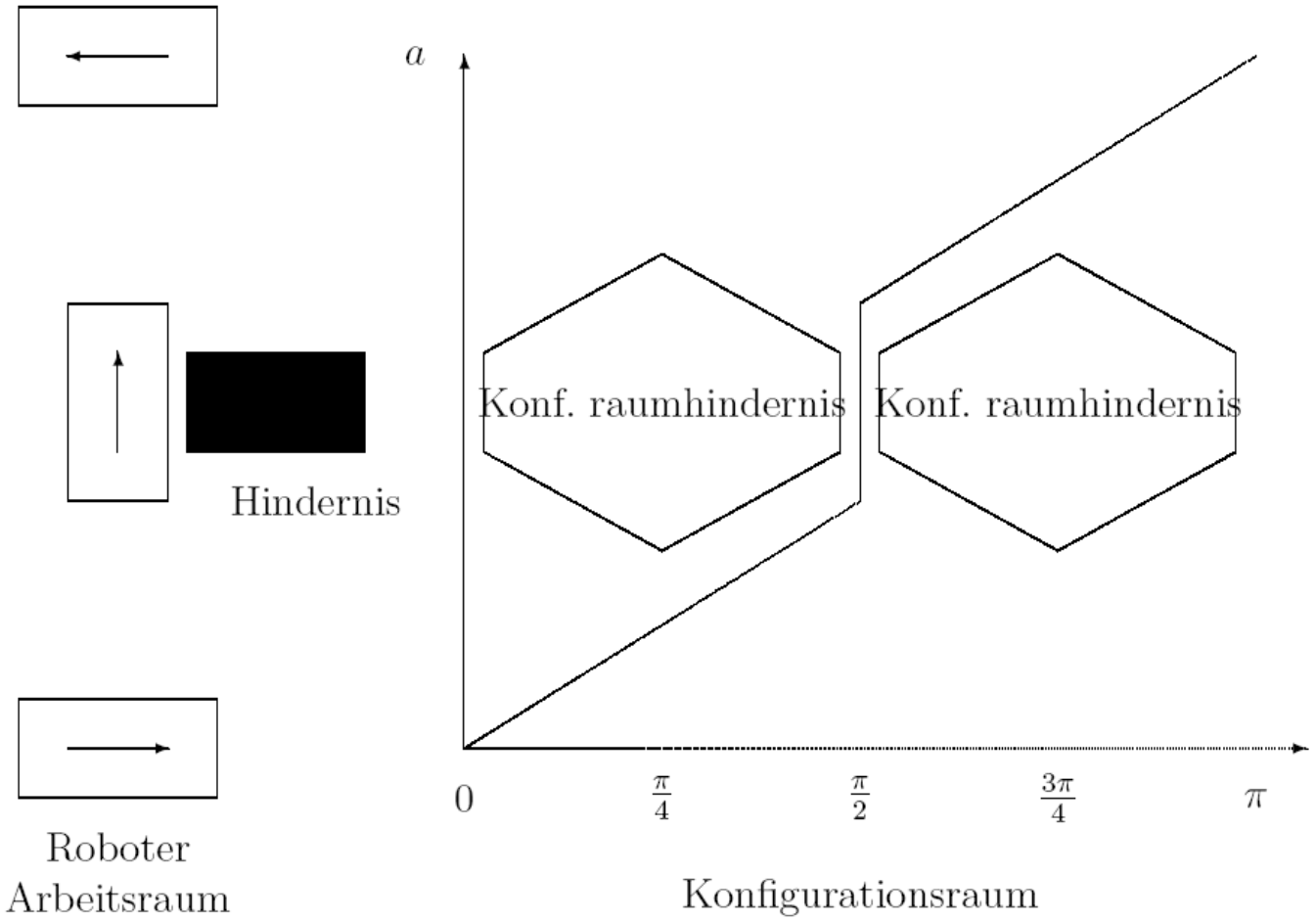
Roboter als Kreis



Konfigurationsraum mit Drehung

- Roboter hat 2 Freiheitsgrade
  - translatorisch (0 … a)
  - rotatorisch (0° …180°)
- Hindernis schränkt rotatorischen Freiheitsgrad ein
- Ein Hindernis erzeugt zwei virtuelle Bereiche im Konfigurationsraum

Konfigurationsraum mit Drehung

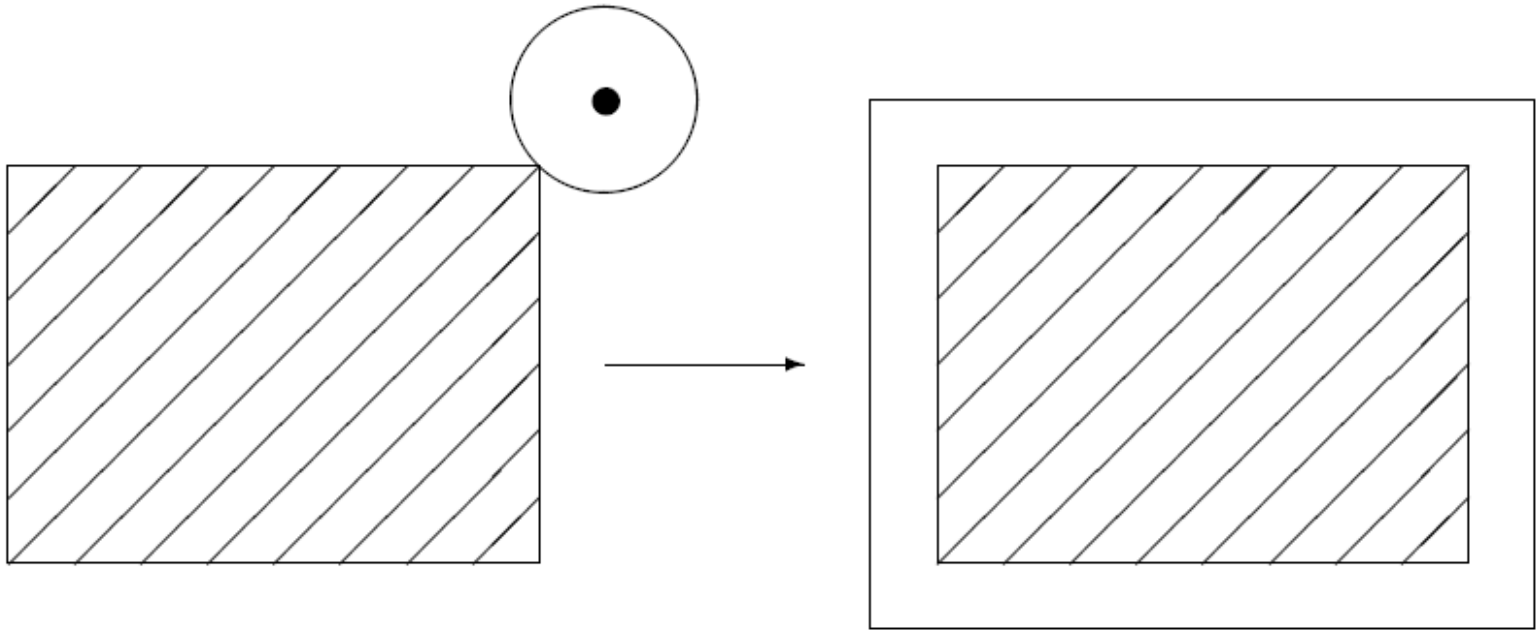


10.2 Modifikationen und andere Verfahren

Roboter als Kreis und konvexe Polygone

- Roboter wird als **Kreis** dargestellt
- Hindernisse werden um den Radius des Kreises vergrößert
- Roboter wird Kreismittelpunkt
- Die hindernisfreien Räume werden in **konvexe Polygone** aufgeteilt.
- 2 Punkte eines konvexen Polygons können durch eine Gerade verbunden werden

Roboter als Kreis



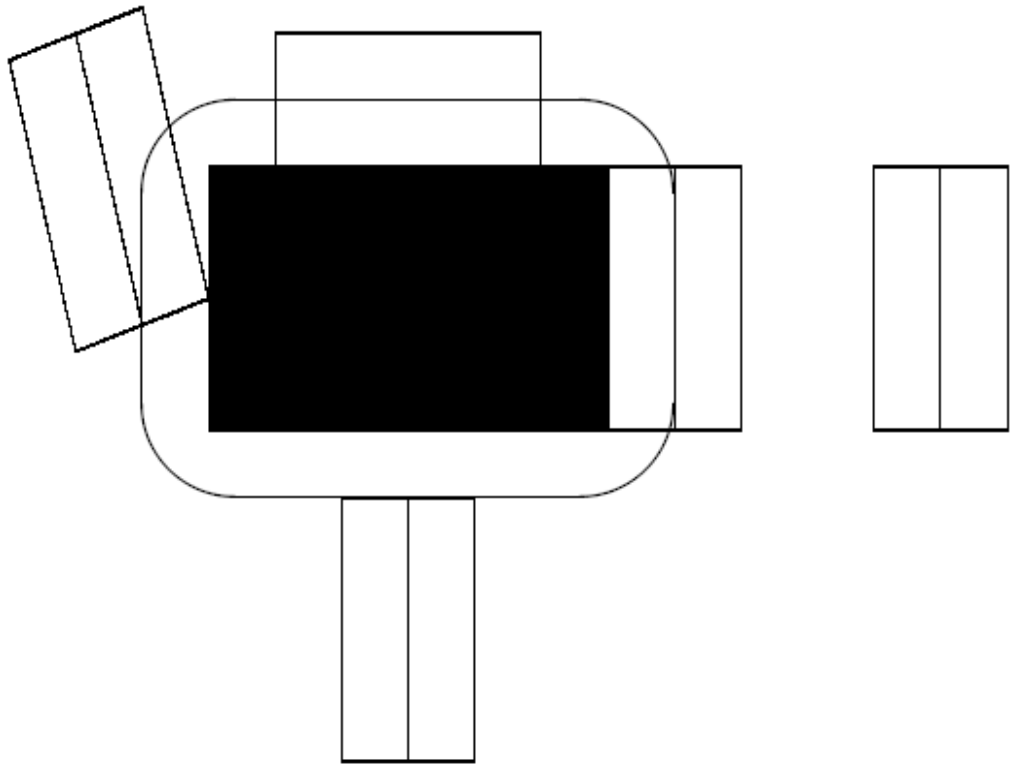
Roboter als Kreis – Suchraum

- Die Knoten sind die konvexen Polygone
- Benachbarte Polygone werden durch eine Kante verbunden
- Ein Nachteil ist die Idealisierung des Roboters als Kreis (mögliche Lösungen werden verhindert)

Roboter als Rechteck

- Roboter wird als Rechteck idealisiert
- Hindernisse werden um die halbe Breite des Rechteckes vergrößert
- Roboter wird auf eine Strecke (Länge des Roboters) reduziert
- Darstellung ist somit unabhängig von der Orientierung des Roboters

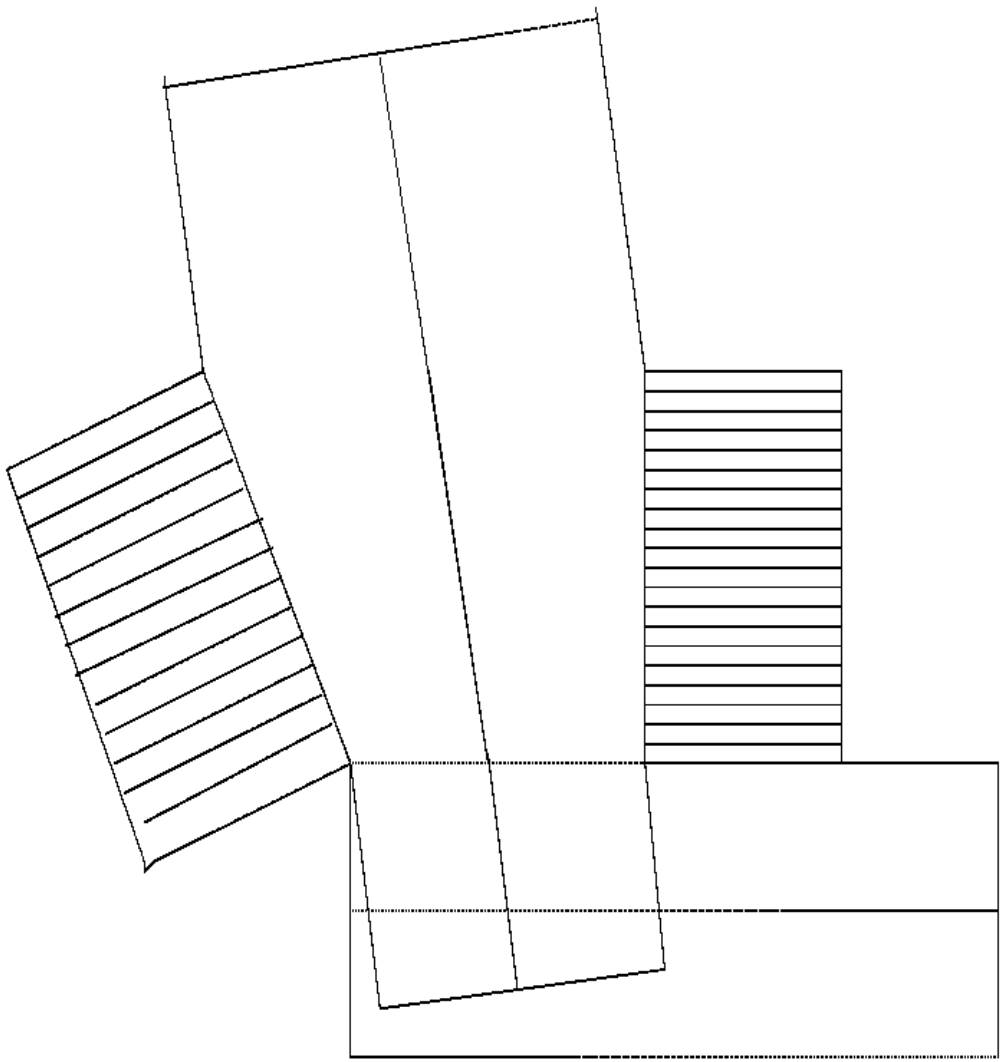
Roboter als Rechteck



Generalisierte Kegel

- Hindernisse sind Rechtecke
- Sie werden nicht vergrößert
- Zwischen 2 Hindernissen wird ein generalisierter Kegel eingefügt
- Über die Enden der Hindernisse werden die Kegel parallel zur Mittellinie verlängert, bis ein anderes Hindernis kommt
- Diese Kegel können sich überlappen

Generalisierte Kegel



Generalisierte Kegel - Suchraum

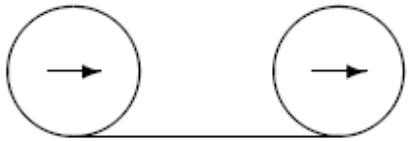
- Knoten sind die Schittpunkte der Mittellinien zweier generalisierter Kegel
- Kante - Abschnitt einer Mittellinie
- Generalisierte Kegel mit einer zu geringen Breite können entfernt werden (Größe des Roboters)
- Die gefundenen Wege befinden sich in der Mitte zwischen 2 Hindernissen

Hindernisse als Kreise

- Hindernisse als Kreise (Hindernis muss ganz im Kreis enthalten sein)
- Wege
  - Direkte Verbindung
  - Tangente an zwei Kreise

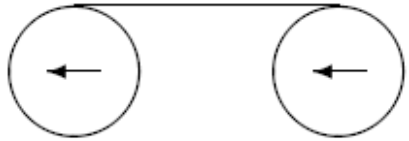
Mögliche Tangenten

1)



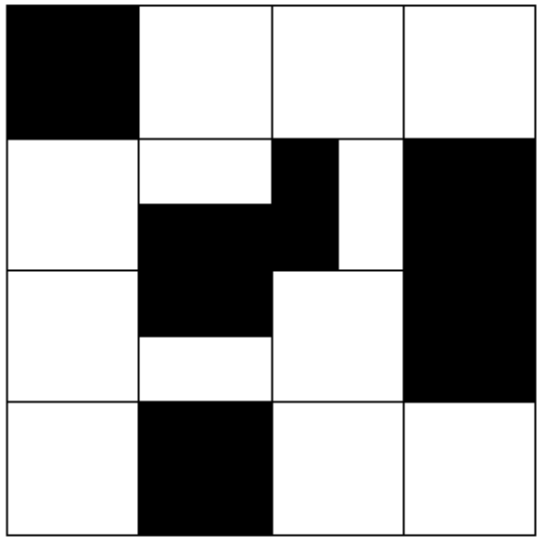
2)

3)



4)

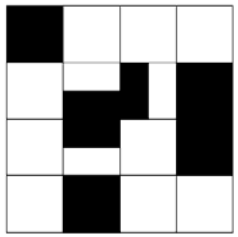
Verwendung von Quadtree Hindernisse

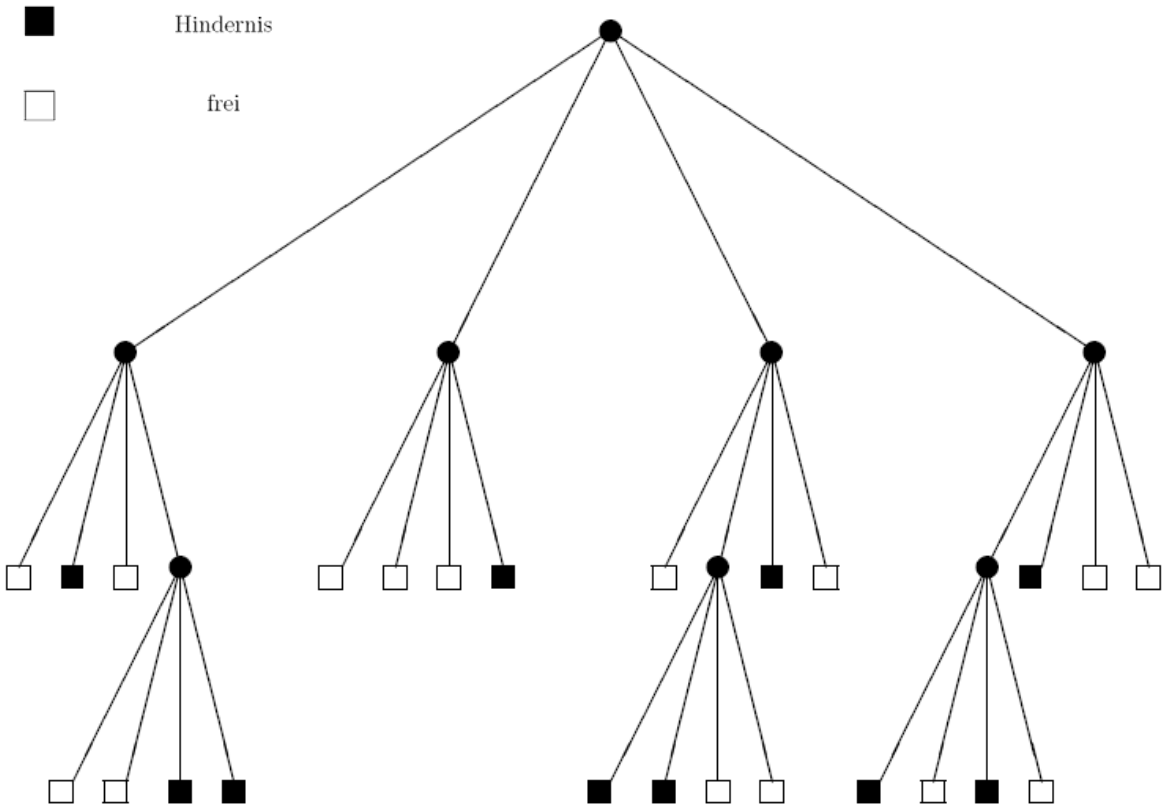


Hindernis

frei

Quadtree





■ Hindernis ● teilweise befahrbar

□ frei

3	4
1	2

Unterteilung eines Quadrats

Verwendung von Quadtree

- Gesucht ist ein Weg von Start zum Ziel über weiße Quadrate
- Man betrachtet den Quadtree auf einer gegebenen Ebene
- Ein teilweise befahrbares Feld kann auf einer tieferen Ebene weiterbetrachtet werden

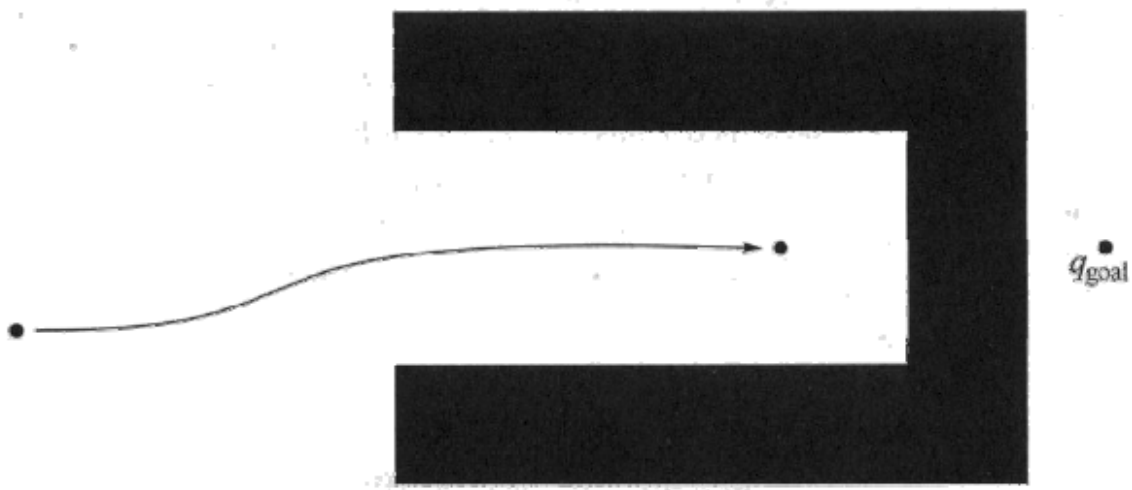
10.3 Lokale Bewegungssteuerung

Potentialfeldmethode

- Man betrachtet den Roboter als einen Massepunkt unter dem Einfluss eines Potentialfeldes
- Es wirken 2 Kräfte:
  - Eine Kraft, die den Roboter von Hindernissen abstößt
  - Eine Kraft, die den Roboter zum Zielpunkt hinzieht
- Roboter versucht gleichzeitig:
  - Abstand zu Hindernissen zu maximieren
  - Entfernung zum Ziel zu minimieren
- Geplanter Roboterweg entspricht dem Weg einer ins Tal rollenden Kugel
- Ein Problem sind lokale Minima (z.B. bei U – förmigen Hindernissen

Lokale Minima

- Wie bei jedem Gradientenverfahren kann der Gradientenabstieg in ein lokales Minimum enden
- Zufallsbewegungen können eventuell aus dem Minimum herausführen



Potentialfeld und Kraftfeld

$$F(\mathrm{c}) = -\nabla U(c) = \begin{pmatrix} \frac{\partial U}{\partial x} \\ \frac{\partial U}{\partial y} \end{pmatrix}$$

Kraft

Potential

Anziehendes Potentialfeld

$$U_{att}(\mathrm{c}) = \frac{1}{2} k_{att} d_{goal}^2(c)$$

$$d_{goal}(c) = \left\| c - c_{goal} \right\|$$

Abstand zum Ziel

$$F_{att}(\mathrm{c}) = -\nabla U_{att}(c)$$
$$= -k_{att} \left\| c - c_{goal} \right\|$$

Abstossendes Potentialfeld

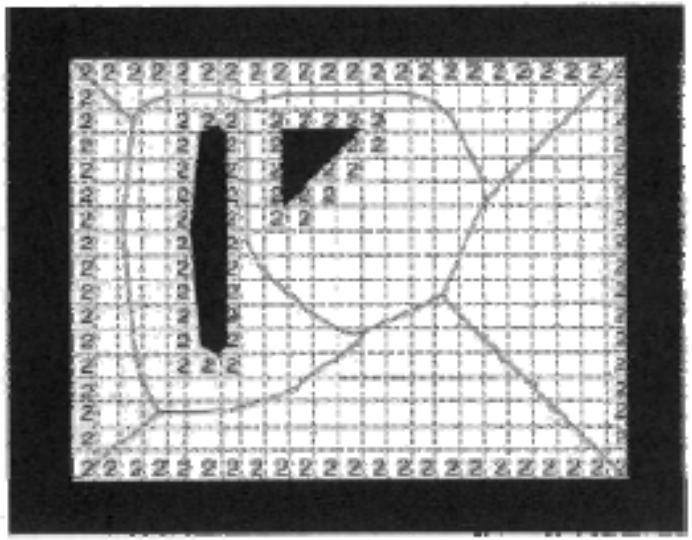
$$U_{rep_i}(c) = \begin{cases} \frac{1}{2} k_{rep} \left(\frac{1}{d_i(c)} - \frac{1}{d^*} \right)^2, & d_i(c) \le d^* \\ 0, & d_i(c) > d^* \end{cases}$$

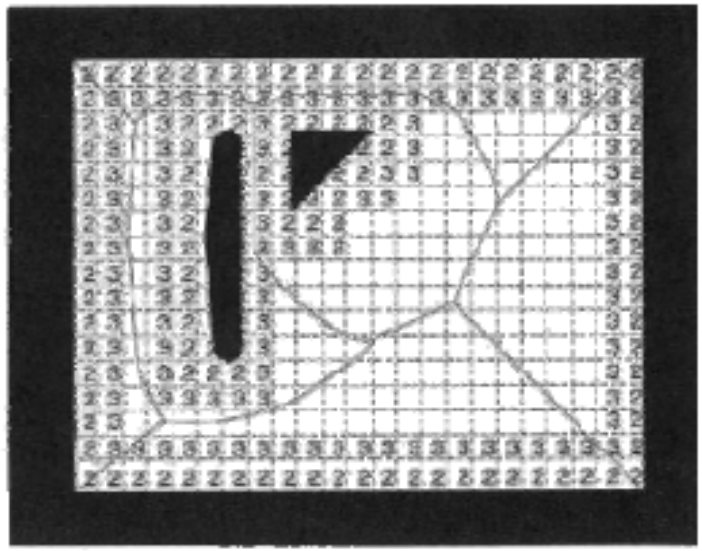
$$F_{rep_i}(c) = -\nabla U_{rep_i} = \begin{cases} k_{rep} \left(\frac{1}{d_i(c)} - \frac{1}{d^*} \right) \frac{1}{d_i^2(c)} \frac{c - c_i}{d_i(c)}, & d_i(c) \le d^* \\ 0, & d_i(c) > d^* \end{cases}$$

Brushfire-Verfahren

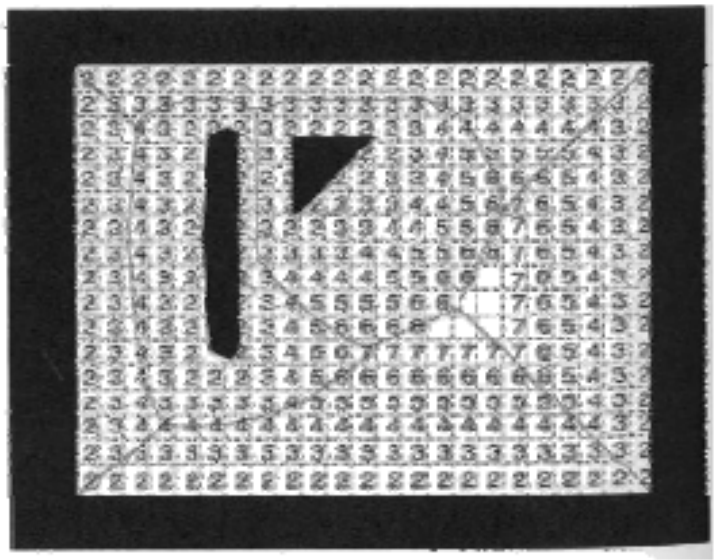
- Die Umgebungskarte m sei als Belegtheitsgitter gegeben.
- Belegte Gitterfelder haben den Wert 1 und nicht belegte Felder den Wert 0.
- Das abstossende Potentialfeld wird wie folgt berechnet:
  - für alle Felder mit Wert 1 werden alle 0-wertigen Nachbarfelder (4-er bzw 8-er-Nachbarschaft) mit 2 besetzt.
  - dann werden für alle Felder mit Wert 2 alle 0-wertigen Nachbarfelder auf 3 gesetzt.
  - fahre so fort, bis kein Feld mehr mit Wert 0 übriggeblieben ist.
- Das Verfahren entspricht einer Breitensuche, die bei den Hindernissen beginnt, oder anschaulich einem Feuer, das nach und nach alle freien Felder erreicht.
- Jedes Gitterfeld enthält jetzt den kürzesten Abstand zum nächsten Hindernispunkt.

Beispiel





Beispiel



Bemerkungen

- Man beachte, dass bei diesem Verfahren nur das am nächsten gelegene Hindernis das Potential an einer Position c bestimmt
- Um das Gesamt-Potential zu erhalten, lässt sich einfach an jedem Gitterfeld noch das anziehende Potential dazuaddieren
- Der Gradient (Bewegungsrichtung des Roboters) kann einfach dadurch bestimmt werden, dass unter allen Nachbarfeldern dasjenige mit dem kleinsten Potentialwert bestimmt wird
- Da der so berechnete Wege ziemlich oszillierend sein kann, ist eine zusätzlich Glättung des Weges ratsam