Karlsruhe Institute of Technology

Robotik I: Einführung in die Robotik
Bewegungsplanung II

Tamim Asfour, Rüdiger Dillmann

KIT-Fakultät für Informatik, Institut für Anthropomatik und Robotik (IAR)
Hochperformante Humanoide Technologien (H^2T)

Inhalt

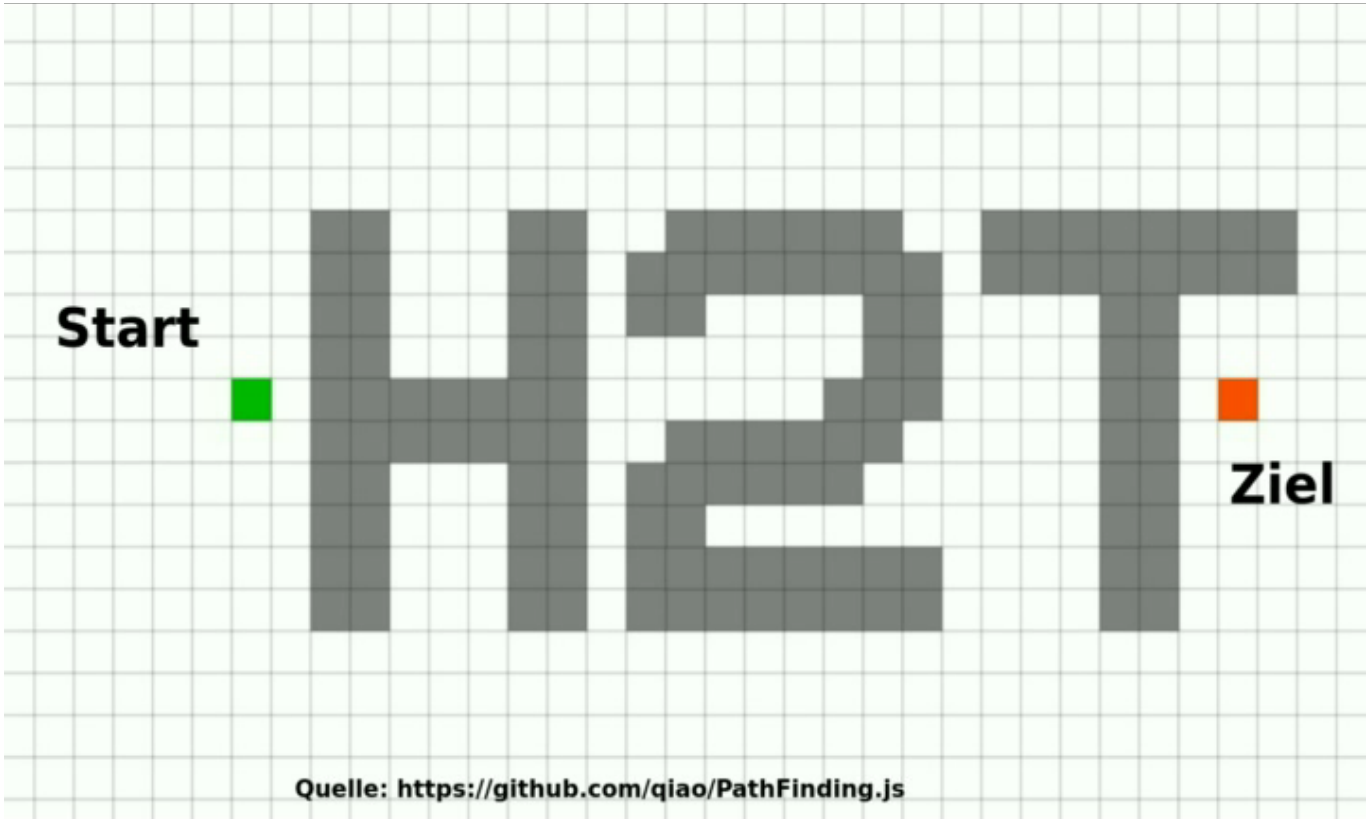
- Motivation
- Grundlagen der Bewegungsplanung
- Pfadplanung für mobile Roboter
  - Voronoi-Diagramme
  - Sichtgraphen
  - Zellzerlegung
  - Potentialfelder
  - A*
- Bewegungsplanung für Manipulatoren

Voronoi-Diagramme – Hausaufgabe

- Welche Punkte werden gewählt, um das Diagramm zu erstellen?

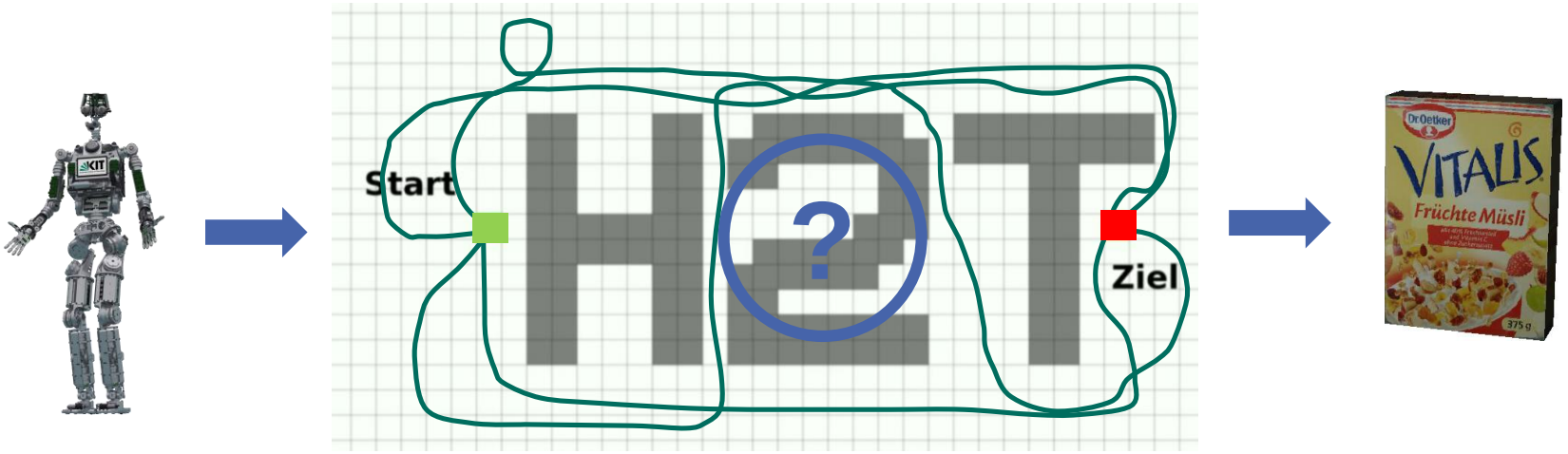
- Antwort: Ecken der Hindernisse

A*-Algorithmus: Motivation



A*-Algorithmus

- Motivation: Kürzester Pfad von Start nach Ziel

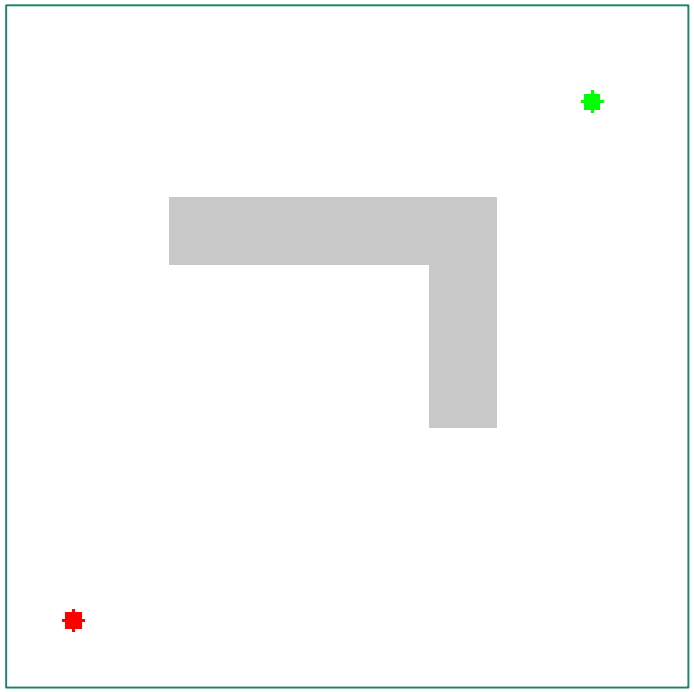


- A* ist einer der beliebtesten Algorithmen zur Routenplanung

- Kostenfunktion ist $f(x) = g(x) + h(x)$
  - $g(x)$ entspricht Kosten von Startknoten zum Knoten x
  - $h(x)$ entspricht **geschätzten** Kosten von Knoten x zum Zielknoten

A* I

- A* (“*A Stern*”) ist ein Algorithmus zur Bestensuche
- Findet den optimalen Pfad von einem Startknoten v_{start} zu einem Zielknoten v_{ziel}
- Optimalität in Bezug auf die Pfadkosten (z.B., kürzester Weg, kürzeste Zeit, kleinste Kantengewichte, etc)



https://de.wikipedia.org/wiki/A*-Algorithmus

A* II

- Iterativer Ansatz
- Es werden zwei Knotenlisten verwaltet
  - Open set O: zu besuchende Knoten
  - Closed set C: bereits besuchte Knoten

- Update: Für einen besuchten Knoten v_n:
  - Vorgängerknoten $pred(v_n)$
  - Akkumulierte Kosten, um v_n zu erreichen: $g(v_n)$
  - Heuristik für die erwarteten Kosten zum Ziel: $h(v_n)$

- Initialisierung
  - $O = \{v_{start}\}$
  - $C = \{\}$
  - $g(v_i) = \infty, 1 \leq i \leq K$
  - $g(v_s) = 0$

A* III

- Algorithmus
  Solange $O \neq \emptyset$
  - Bestimme den zu erweiternden Knoten
    - Finde $v_i \in O$ mit minimalem $f(v_i) = g(v_i) + h(v_i)$
  - Wenn $v_i = v_{ziel}$
    Lösung gefunden: Traversiere Vorgänger von v_i bis v_{start} erreicht ist.
  - $O.remove(v_i)$
  - $C.add(v_i)$
  - Update für alle Nachfolger v_j von v_i durchführen
    - Wenn $v_j \in C$, dann überspringe v_j
    - Wenn $v_j \notin O$, dann $O.add(v_j)$
    - Wenn $g(v_i) + cost(v_i, v_j) < g(v_j)$
      - $g(v_j) = g(v_i) + cost(v_i, v_j)$
      - $h(v_j) = heuristic(v_j, v_{ziel})$
      - $pred(v_j) = v_i$

A*: Beispiel

- Gitter mit 15 Knoten
- Finde den optimalen Pfad von v_2 nach v_{13}
  - Nur horizontale und vertikale Bewegungen erlaubt
  - Kosten:
    - Betreten einer grauen Zelle: 1
    - Betreten einer gelben Zelle: 4
  - Heuristik h: Euklidische Distanz zu v_{13}
    (z.B. $h(v_{11}) = \sqrt{2}$)

v_1	v_2	v_3
v_4	v_5	v_6
v_7	v_8	v_9
v_{10}	v_{11}	v_{12}
v_{13}	v_{14}	v_{15}

A*: Eigenschaften

- Findet eine optimale Lösung
- Heuristik h darf die minimalen Kosten, das Ziel zu erreichen, nicht überschätzen.
  - h ist zulässig

- A* ist auch optimal effizient für jede (zulässige) Heuristik h
  - Kein optimaler Algorithmus, der die gleiche Heuristik verwendet, besucht weniger Knoten als A*

- Wenn $h(x) = 0 \; \forall x$: Dijkstra's algorithm, d.h. $f = g$
  - Greedy Algorithmus: geachtet die Kosten nicht, d.h. $f = h; \; (g = 0)$
  - Ignoriert Schätzung der Distanz zum Zielknoten
  - Besucht mehr Knoten als notwendig

Inhalt

- Motivation
- Grundlagen der Bewegungsplanung
- Pfadplanung für mobile Roboter
- Bewegungsplanung für Manipulatoren
  - Probabilistic Roadmaps (PRM)
  - Rapidly-exploring Random Trees (RRT)
  - Erweiterungen
    - Constrained RRT
    - RRT*
    - Dynamic Domain RRT
    - Bridge Sampling

Grundlagen der Bewegungsplanung: Begriffsbildung

- **Pfadplanung**
  - Starres Objekt (z.B. mobiler Roboter, autonomes Fahrzeug)
  - 2D Problem (Position: x,y)
  - 3D Problem (Position: x,y; Rotation: α)
    → Piano Mover‘s Problem

- **Bewegungsplanung**
  - Mehrkörpersystem (z.B. Roboterarme, Systeme mit mehreren Robotern)
  - Hochdimensionale Problemstellungen

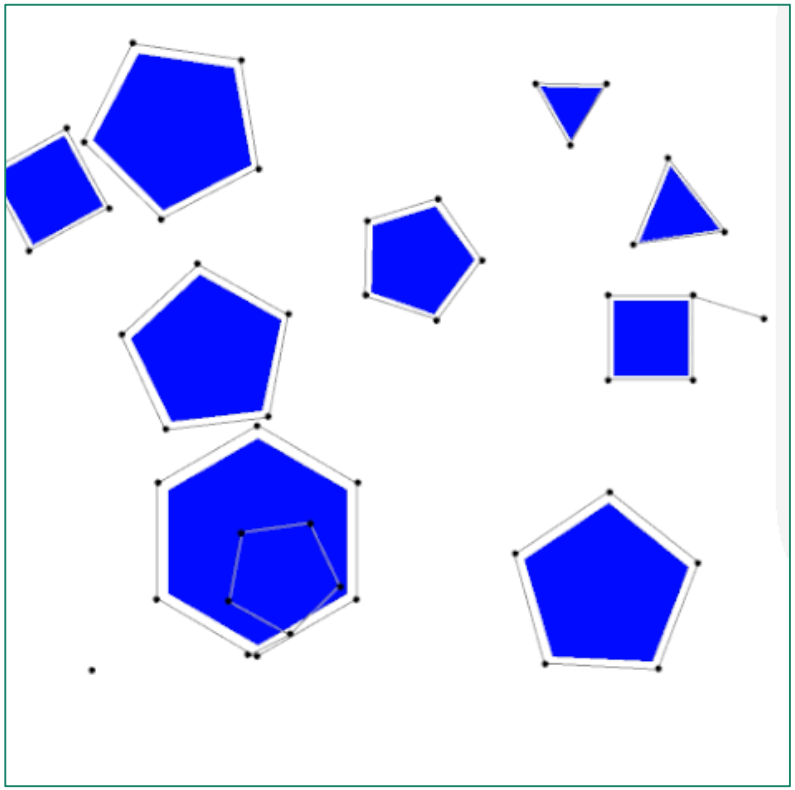
- **Randbedingungen, auch Zwangsbedingungen**
  - Globale Randbedingungen: Limitieren den gültigen Konfigurationsraum
    z.B. aufrechte Position des Endeffektors, maximale Motorströme, etc.
  - Lokale Randbedingungen: Schränken die Übergänge zwischen Konfigurationen ein z.B. Nicht-holonome Fahrzeuge, max. Geschwindigkeit/Beschleunigung

Inhalt

- Motivation
- Grundlagen der Bewegungsplanung
- Pfadplanung für mobile Roboter
- Bewegungsplanung für Manipulatoren
  - Probabilistic Roadmaps (PRM)
  - Rapidly-exploring Random Trees (RRT)
  - Erweiterungen
    - Constrained RRT
    - RRT*
    - Dynamic Domain RRT
    - Bridge Sampling

Probabilistic Roadmaps (PRM)

- Kavraki, L. E.; Svestka, P.; Latombe, J.-C.; Overmars, M. H. (1996), "Probabilistic roadmaps for path planning in high-dimensional configuration spaces", IEEE Transactions on Robotics and Automation, 12 (4): 566–580, doi:10.1109/70.508439



https://en.wikipedia.org/wiki/Probabilistic_roadmap

Probabilistic Roadmaps (PRM)

- PRM basiert auf eine **Approximation des Freiraumes** durch den Graphen (Roadmap) → Effizienter als die Erzeugung einer expliziten Repräsentation des Freiraumes
- PRM is **probabilistisch**: Zufallsgesteuerte Erzeugung durch das Sampling

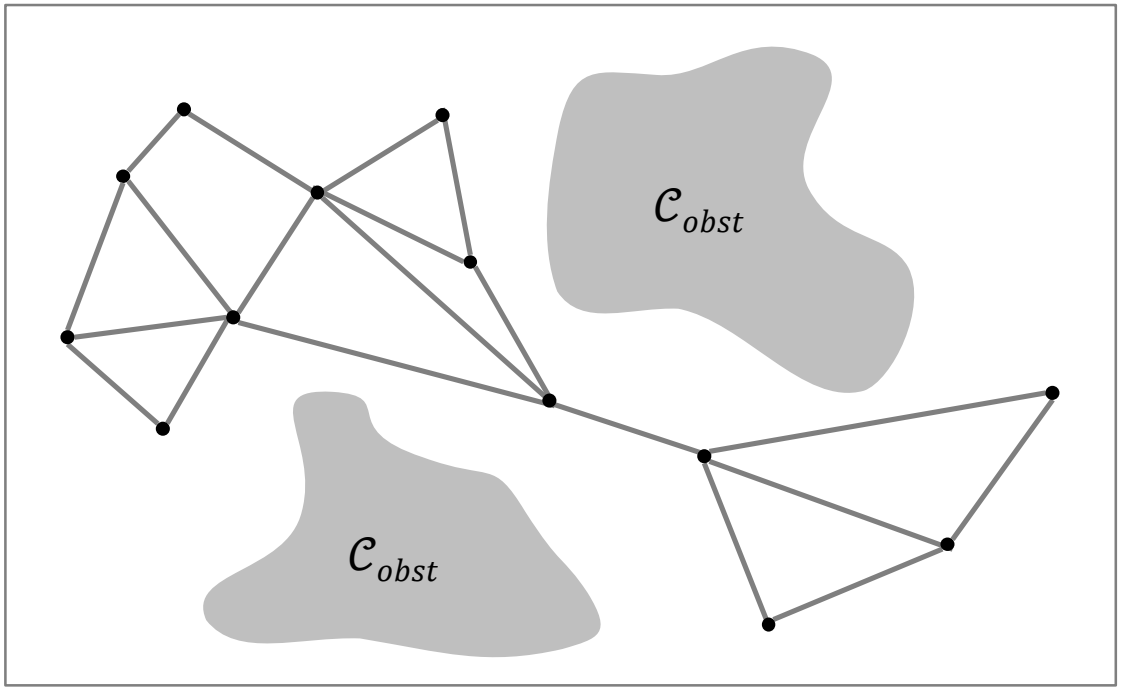
PRM Algorithmus

- **Schritt 1**: Vorverarbeitung
  - Erzeugung einer kollisionsfreien Straßenkarte
    → Graph

- **Schritt 2**: Anfrage
  - Verbinde $\boldsymbol{q}_{start}$ und $\boldsymbol{q}_{ziel}$ mit dem Graphen
  - Suche einen Weg von $\boldsymbol{q}_{start}$ nach $\boldsymbol{q}_{ziel}$ durch den Graphen

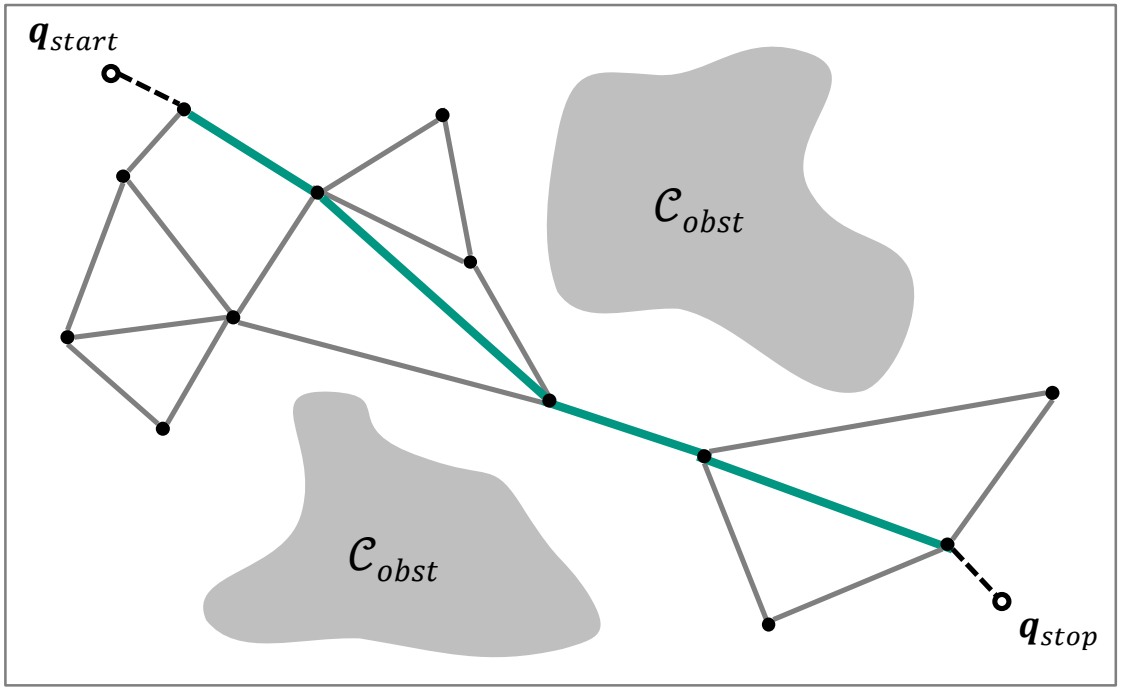
PRM: Vorverarbeitung

- Zufällige Erzeugung von kollisionsfreien Stichproben (Sampling)
- Lokale Planung: Stichproben werden über kollisionsfreie Pfade miteinander verbunden



PRM: Anfrage

- Verbinde $\boldsymbol{q}_{start}$ und $\boldsymbol{q}_{ziel}$ mit dem Wegenetz
- Suche im Graphen (z.B. mit A*)



PRM: Konstruktion des Graphen

- N: Anzahl der Knoten im Graphen
- R: PRM, Graph

- Algorithmus:
  - Erzeugen von N zufälligen Konfigurationen in C_{free}
  - Einfügen der erzeugten Konfigurationen als Knoten in R
  - Für jeden Knoten $v_i \in R$
    - Finde die k nächsten Nachbarn von v_i aus R: $\mathcal{N}(v_i)$
    - Für jeden Knoten $v \in \mathcal{N}(v_i)$
      - Wenn es einen (neuen) kollisionsfreien Pfad von v nach v_i gibt, dann füge die Kante (v, v_i) in R ein
  - Ergebnis: R

Lokale Planung

PRM: Eigenschaften

- Einmalige Konstruktion des Graphen
  - Anfragen können effizient bearbeitet werden
- Randomisierter Ansatz zur Konstruktion
  - Exponentieller Anstieg der Laufzeit mit der Dimension des Konfigurationsraums wird vermieden
- Verfahren hängt stark vom verwendeten Sampling ab
  - Problem: Schmale Passagen zwischen Hindernissen
  - Lösungsansatz: Sampling in der Nähe von Hindernissen erhöhen

PRM: Unterschiedliche Sampling-Strategien

- Zufällig:
  - Konfiguration wird zufällig generiert und auf Kollision geprüft
- Grid:
  - Konfigurationen werden mit diskreter Auflösung erzeugt
  - Auflösung einzelner Zellen wird hierarchisch bestimmt
- Halton:
  - Halton-Menge: Menge von Punkten, die ein Bereich besser abdeckt als Grid
  - Basiert auf dem mathematischen Konzept der Diskrepanz
- Zellenbasiert:
  - Sampling in Zellen mit kleiner werdenden Ausmaßen
  - Zellgröße wird mit jeder Iteration verkleinert (z. B. auf 1/8)

Geraerts, Roland, and Mark H. Overmars. "A comparative study of probabilistic roadmap planners." Algorithmic Foundations of Robotics V. Springer Berlin Heidelberg, 2004. 43-57.

Inhalt

- Motivation
- Grundlagen der Bewegungsplanung
- Pfadplanung für mobile Roboter
- Bewegungsplanung für Manipulatoren
  - Probabilistic Roadmaps (PRM)
  - Rapidly-exploring Random Trees (RRT)
  - Erweiterungen
    - Constrained RRT
    - RRT*
    - Dynamic Domain RRT
    - Bridge Sampling

Rapidly-exploring Random Trees (RRTs)

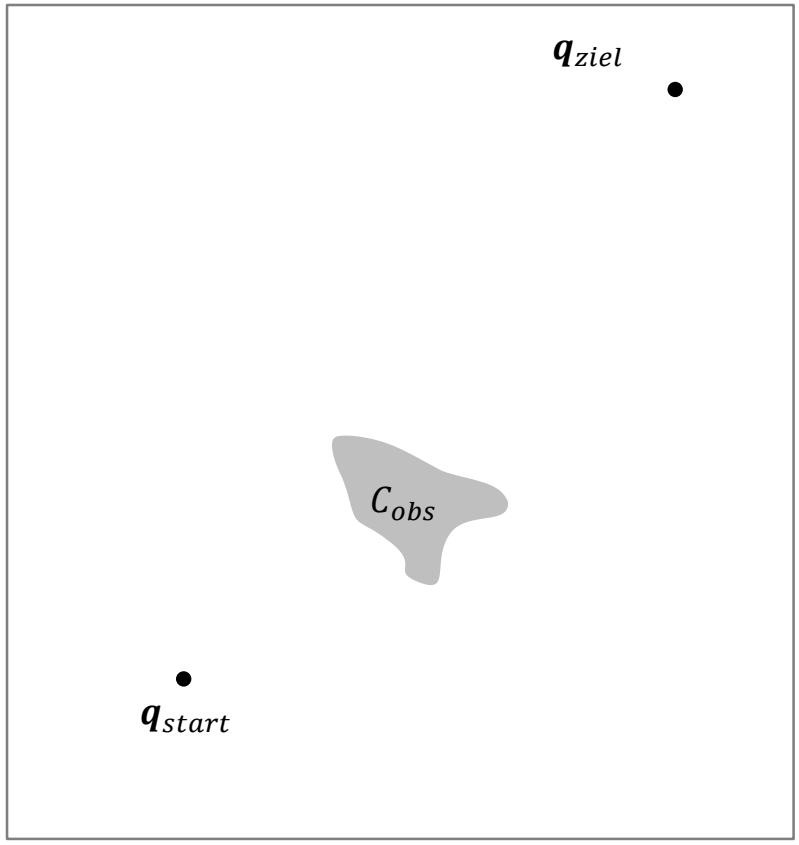
- Algorithmus zur Einmalanfrage
  - Im Gegensatz zu PRMs
  - Keine Vorverarbeitung nötig
  - Keine Probleme mit sich verändernden Umgebungen
- Probabilistisch vollständiger, randomisierter Algorithmus
  - Keine Garantie, dass eine Lösung innerhalb eines Zeitlimits gefunden wird
  - Wenn eine Lösung existiert, wird sie gefunden (Laufzeit geht gegen Unendlich)
  - Terminiert nicht, wenn keine Lösung existiert
- Effizient für hochdimensionale Problemstellungen
- Erweiterungen der klassischen RRT für spezifische Problemstellungen
  z.B. enge Durchgänge

RRT: Prinzip I

- Die Form von C_{obs} im Konfigurationsraum ist unbekannt

- Initialisierung des RRT
  - Erzeuge leeren Baum T
  - Füge $\boldsymbol{q}_{start}$ in T ein

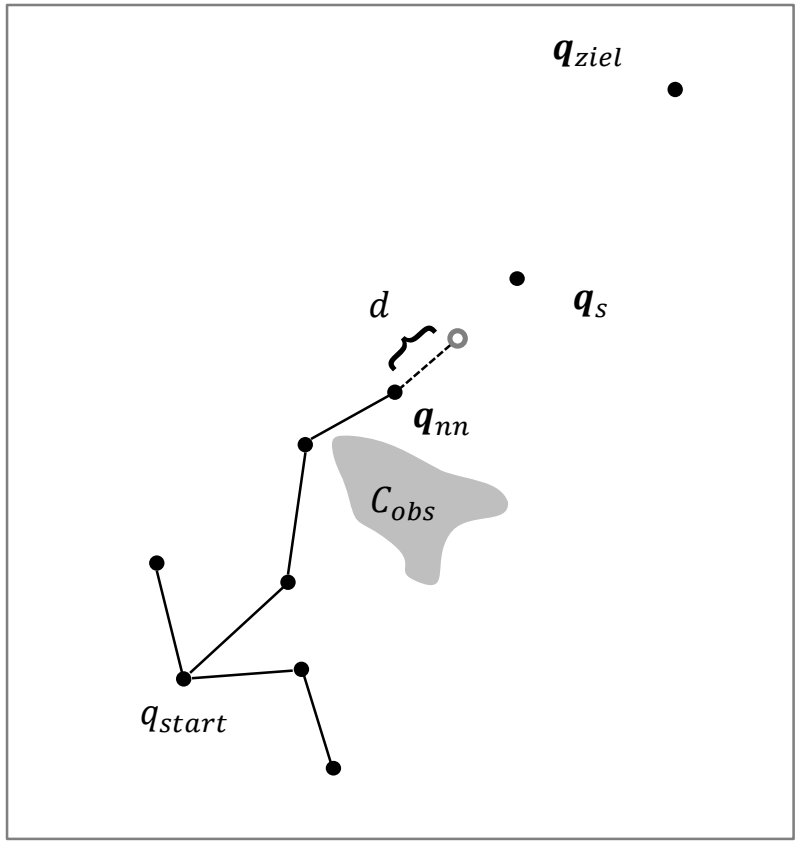
Beispiel mit 2D Konfigurationsraum



RRT: Prinzip II

- Iteration
  1. Erzeuge einen zufälligen Punkt $\boldsymbol{q}_s$
  2. Bestimme den nächsten Nachbarn $\boldsymbol{q}_{nn}$ in T
  3. Füge Punkte auf der Verbindung zwischen $\boldsymbol{q}_s$ und $\boldsymbol{q}_{nn}$ in T ein
     - Mit der Schrittweite d
     - Prüfe jeden der Teilpfade auf Kollision mit C_{obs}.
     - Stoppe, wenn eine Kollision erkannt wurde.
  4. Gehe zu 1.

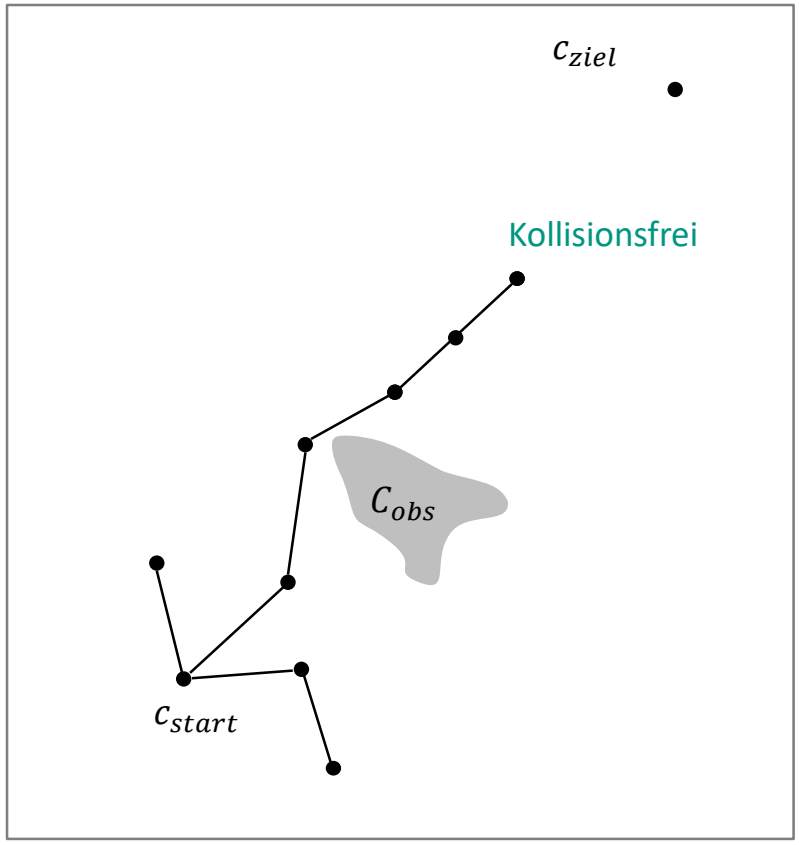
Beispiel mit 2D Konfigurationsraum



RRT: Prinzip III

- Iteration
  1. Erzeuge einen zufälligen Punkt $\boldsymbol{q}_s$
  2. Bestimme den nächsten Nachbarn $\boldsymbol{q}_{nn}$ in T
  3. Füge Punkte auf der Verbindung zwischen $\boldsymbol{q}_s$ und $\boldsymbol{q}_{nn}$ in T ein
     - Mit der Schrittweite d
     - Prüfe jeden der Teilpfade auf Kollision mit C_{obs}. Stoppe, wenn eine Kollision erkannt wurde.
  4. Gehe zu 1.

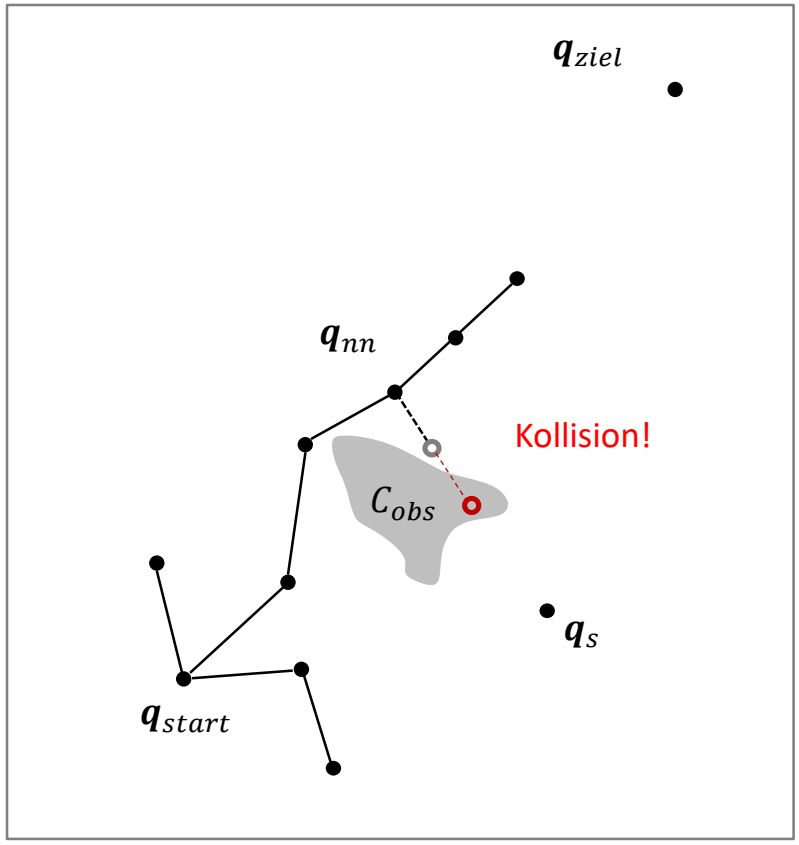
Beispiel mit 2D Konfigurationsraum



RRT: Prinzip IV

- Iteration
  1. Erzeuge einen zufälligen Punkt $\boldsymbol{q}_s$
  2. Bestimme den nächsten Nachbarn $\boldsymbol{q}_{nn}$ in T
  3. Füge Punkte auf der Verbindung zwischen $\boldsymbol{q}_s$ und $\boldsymbol{q}_{nn}$ in T ein
     - Mit der Schrittweite d
     - Prüfe jeden der Teilpfade auf Kollision mit C_{obs}. Stoppe, wenn eine Kollision erkannt wurde.
  4. Gehe zu 1.

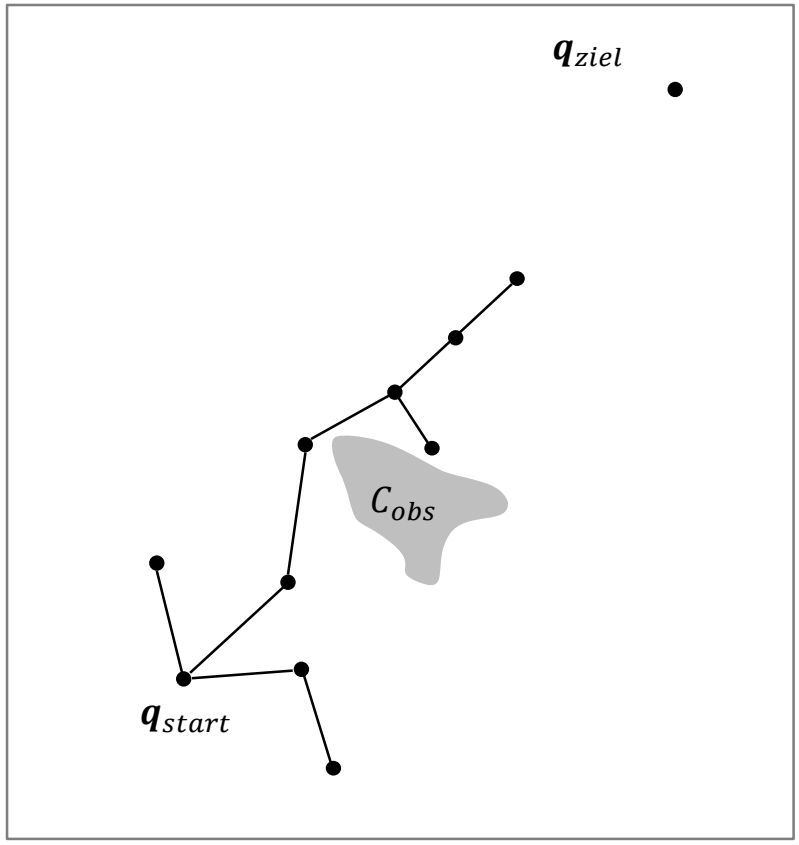
Beispiel mit 2D Konfigurationsraum



RRT: Prinzip V

- Iteration
  1. Erzeuge einen zufälligen Punkt $\boldsymbol{q}_s$
  2. Bestimme den nächsten Nachbarn $\boldsymbol{q}_{nn}$ in T
  3. Füge Punkte auf der Verbindung zwischen $\boldsymbol{q}_s$ und $\boldsymbol{q}_{nn}$ in T ein
     - Mit der Schrittweite d
     - Prüfe jeden der Teilpfade auf Kollision mit C_{obs}. Stoppe, wenn eine Kollision erkannt wurde.
  4. Gehe zu 1.

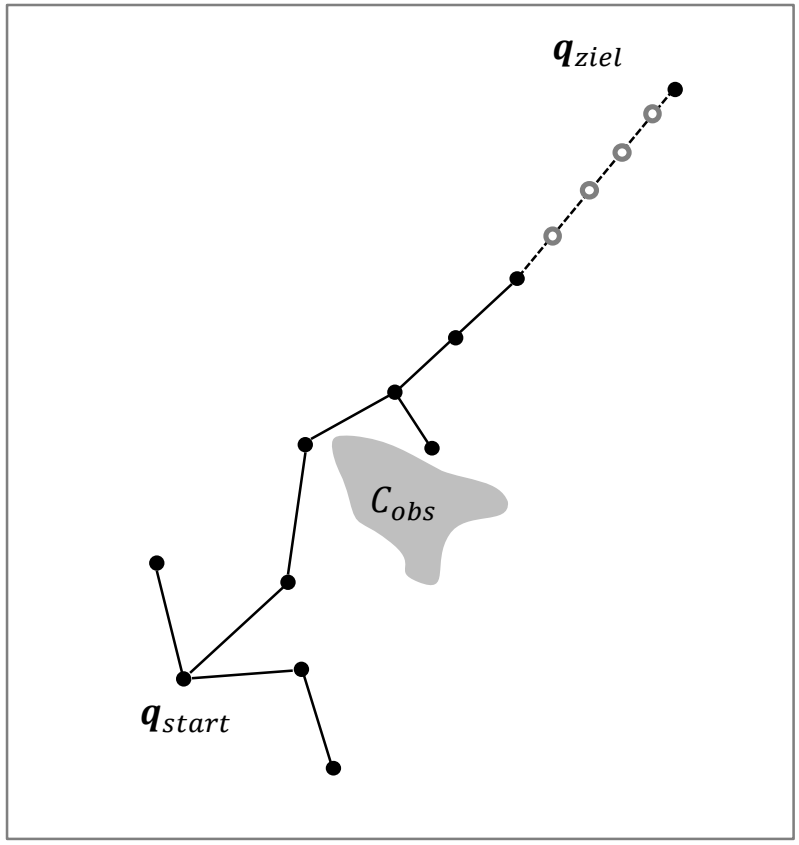
Beispiel mit 2D Konfigurationsraum



RRT: Prinzip VI

- Iteration
  1. Erzeuge einen zufälligen Punkt $\boldsymbol{q}_s$
  2. Bestimme den nächsten Nachbarn $\boldsymbol{q}_{nn}$ in T
  3. Füge Punkte auf der Verbindung zwischen $\boldsymbol{q}_s$ und $\boldsymbol{q}_{nn}$ in T ein
     - Mit der Schrittweite d
     - Prüfe jeden der Teilpfade auf Kollision mit C_{obs}. Stoppe, wenn eine Kollision erkannt wurde.
  4. Gehe zu 1.
- Prüfe in jedem k-ten Schritt, ob $\boldsymbol{q}_{ziel}$ mit T verbunden werden kann

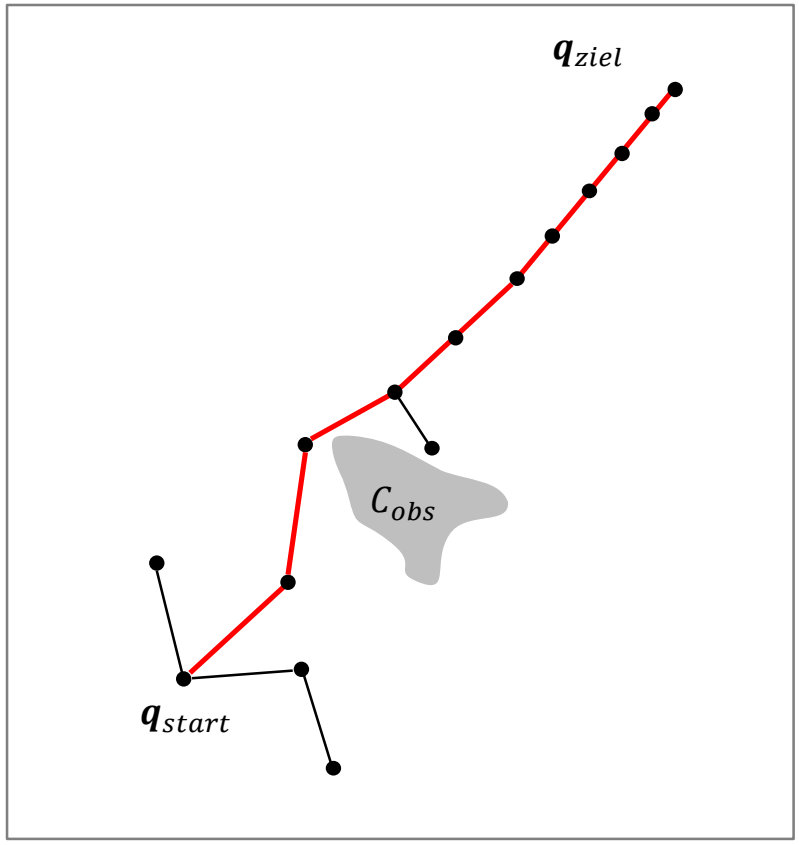
Beispiel mit 2D Konfigurationsraum



RRT: Prinzip VII

- Iteration
  1. Erzeuge einen zufälligen Punkt $\boldsymbol{q}_s$
  2. Bestimme den nächsten Nachbarn $\boldsymbol{q}_{nn}$ in T
  3. Füge Punkte auf der Verbindung zwischen $\boldsymbol{q}_s$ und $\boldsymbol{q}_{nn}$ in T ein
     - Mit der Schrittweite d
     - Prüfe jeden der Teilpfade auf Kollision mit C_{obs}. Stoppe, wenn eine Kollision erkannt wurde.
  4. Gehe zu 1.
- Prüfe in jedem k-ten Schritt, ob $\boldsymbol{q}_{ziel}$ mit T verbunden werden kann

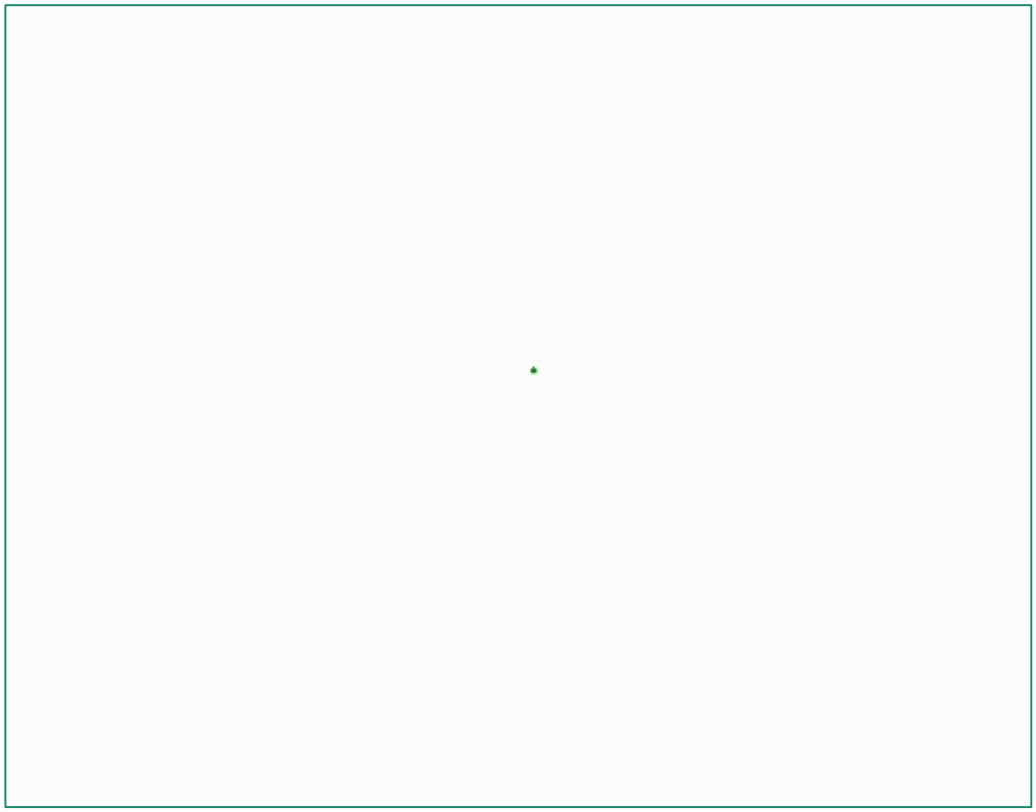
Beispiel mit 2D Konfigurationsraum



Lösung gefunden

RRT: Probabilistische Vollständigkeit I

- RRTs sind probabilistisch vollständig
- Intuition: T breitet sich gleichmäßig im Konfigurationsraum aus



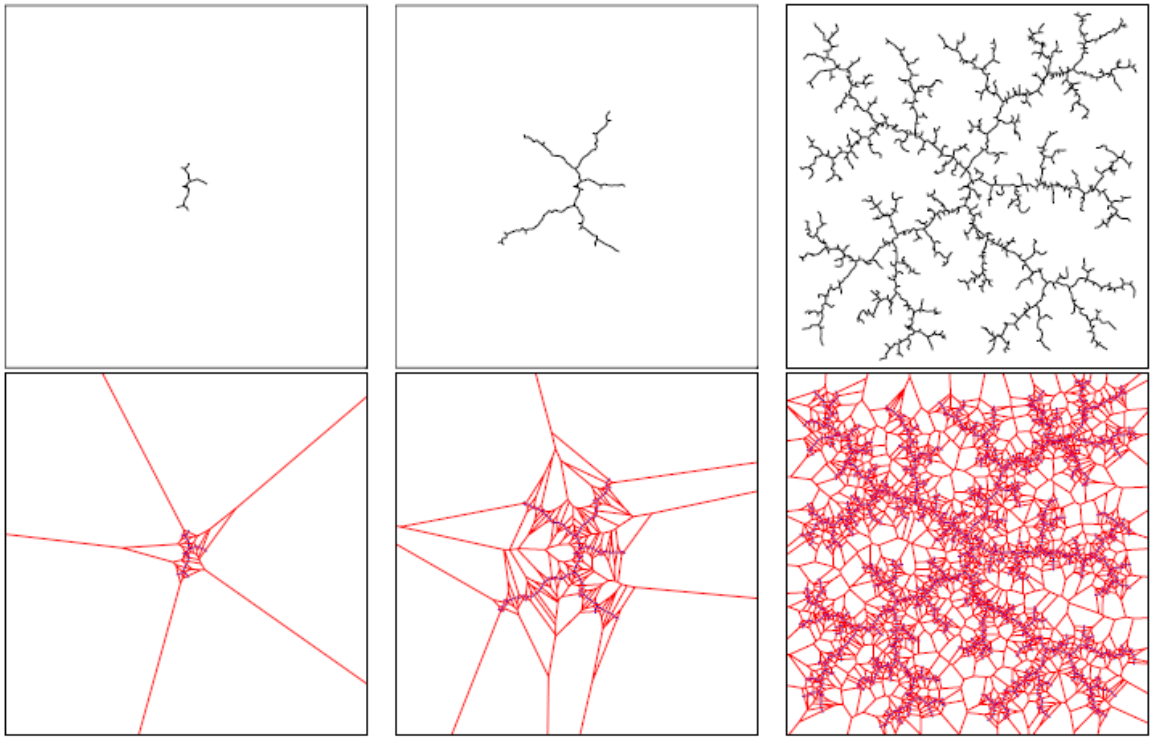
https://en.wikipedia.org/wiki/Rapidly-exploring_random_tree

RRT: Probabilistische Vollständigkeit II

- Die Wahrscheinlichkeit, dass ein Knoten von T erweitert wird, ist proportional zur Größe seiner Voronoi-Region

RRT

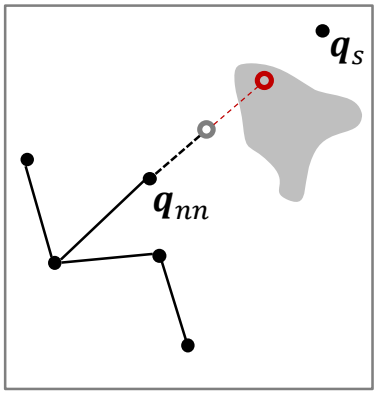
Voronoi-Regionen



J. J. Kuffner and S. M. LaValle, *RRT-connect: An efficient approach to single-query path planning*, *Proceedings IEEE International Conference on Robotics and Automation.* 2000, pp. 995-1001

RRT: Kollisionsprüfung I

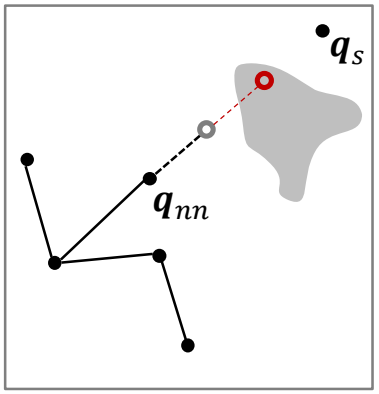
- C_{obs} ist nicht bekannt, wie kann geprüft werden, ob $\boldsymbol{q}_s \in C_{obs}$?



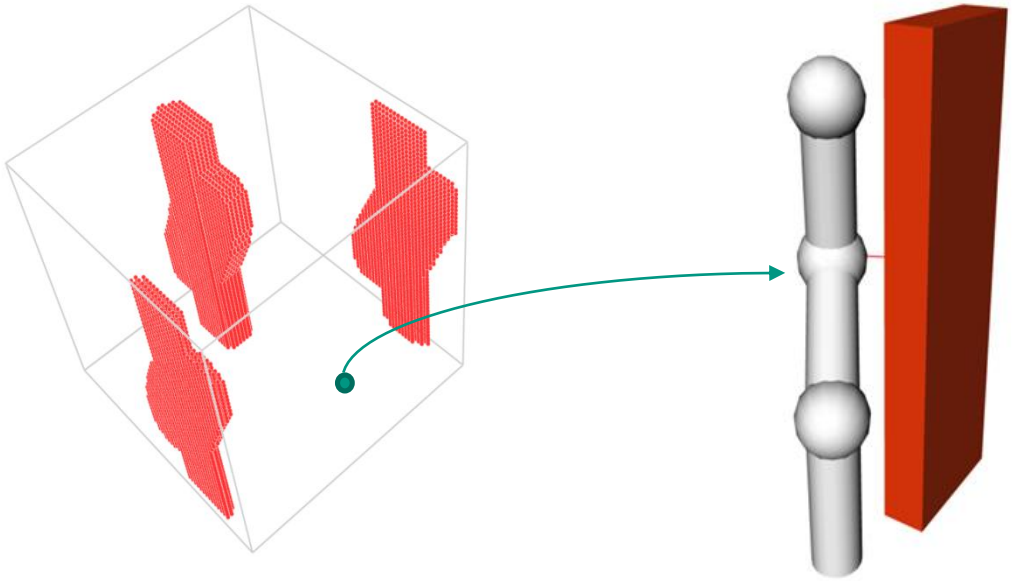
Kollision!

RRT: Kollisionsprüfung II

- C_{obs} ist nicht bekannt, wie kann geprüft werden, ob $\boldsymbol{q}_s \in C_{obs}$?
- Jeder Punkt $c_s \in C$ beschreibt eine Konfiguration des Roboters
  - Kollisionsprüfung im Arbeitsraum durchführen

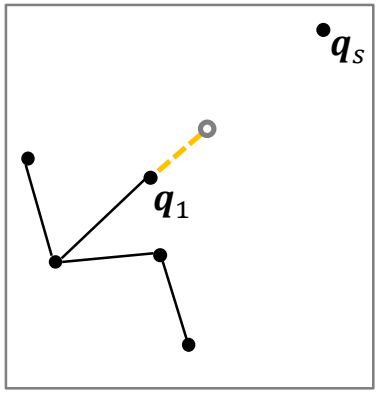


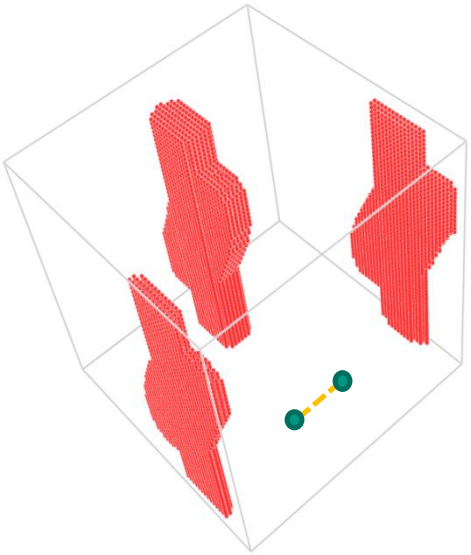
Kollision!

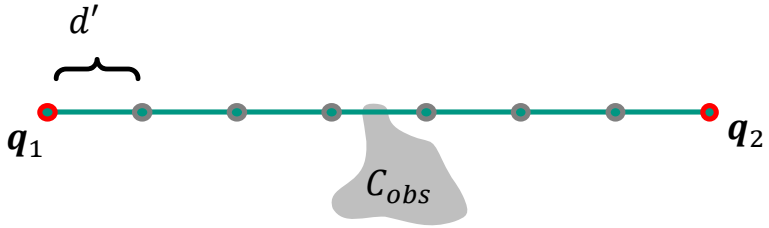


RRT: Kollisionsprüfung III

- Es müssen auch Pfadsegmente auf Kollision geprüft werden
  - Continuous collision detection (CCD)
    Exakt, aber langsam
  - Sampling-basiertes Verfahren
    - Einzelne Punkte auf dem Pfad werden geprüft
    - Schnell, aber nicht exakt
    - Beinhaltet einen Parameter (Sampling-Distanz d')

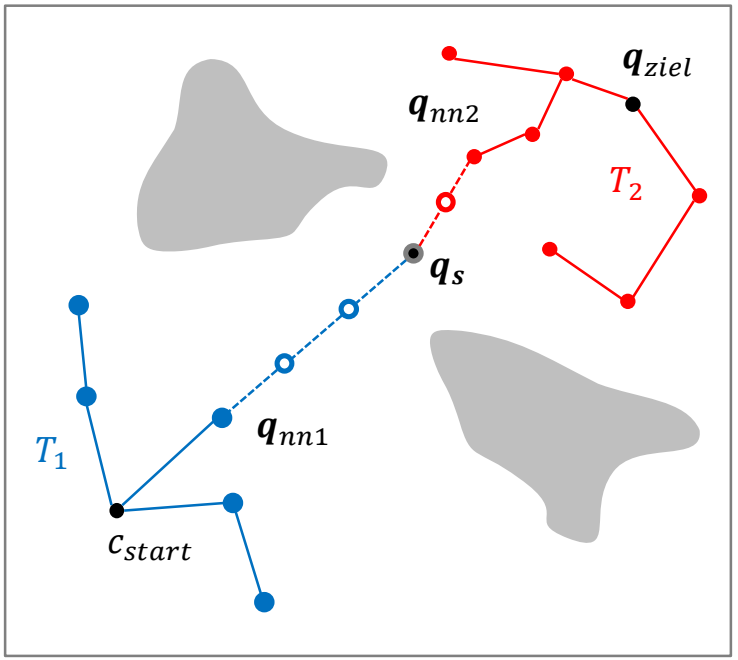






Bidirektionale RRTs

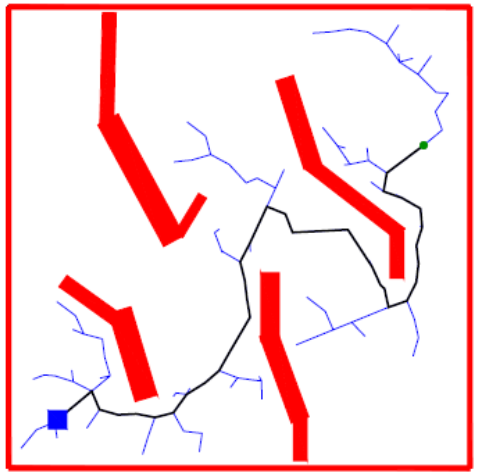
- Es werden zwei Bäume aufgebaut
  - T_1 ausgehend von $\boldsymbol{q}_{start}$
  - T_2 ausgehend von $\boldsymbol{q}_{ziel}$
- Zufällig gewählte Punkte $\boldsymbol{q}_s$ erweitern beide Bäume über:
  - $\boldsymbol{q}_{nn,1}$ (Nächster Nachbar in T_1)
  - $\boldsymbol{q}_{nn,2}$ (Nächster Nachbar in T_2)
- Eine Lösung ist gefunden, wenn beide Bäume mit $\boldsymbol{q}_s$ verbunden wurden
- Original-Version (ähnlich): RRT-Connect



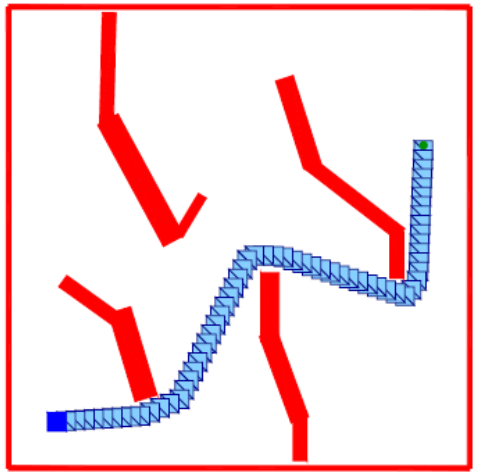
J. J. Kuffner and S. M. LaValle, *RRT-connect: An efficient approach to single-query path planning*, *Proceedings IEEE International Conference on Robotics and Automation.* 2000, pp. 995-1001

RRT: Nachbearbeitung

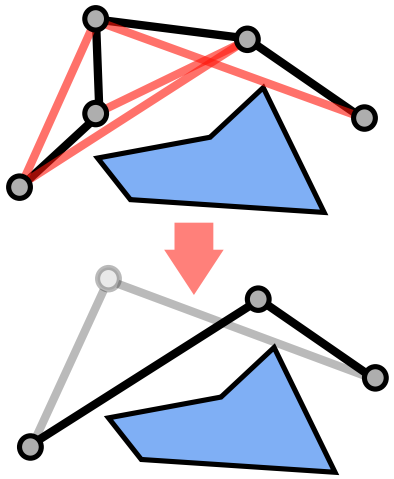
- Lösungen können durch Nachbearbeitung verbessert werden
  - Zufällige Wahl zweier Knoten im Lösungsweg
  - Falls die Verbindung kollisionsfrei ist, verbinde beide Knoten und lösche den dazwischenliegenden Knoten aus dem Lösungspfad
  - Erzeugt glattere Trajektorien



Ursprünglicher Lösungspfad

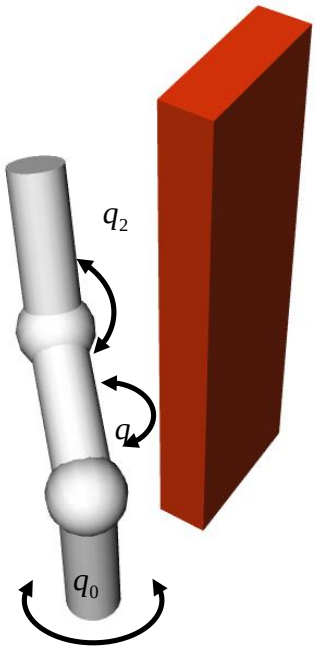


Geglätteter Lösungspfad

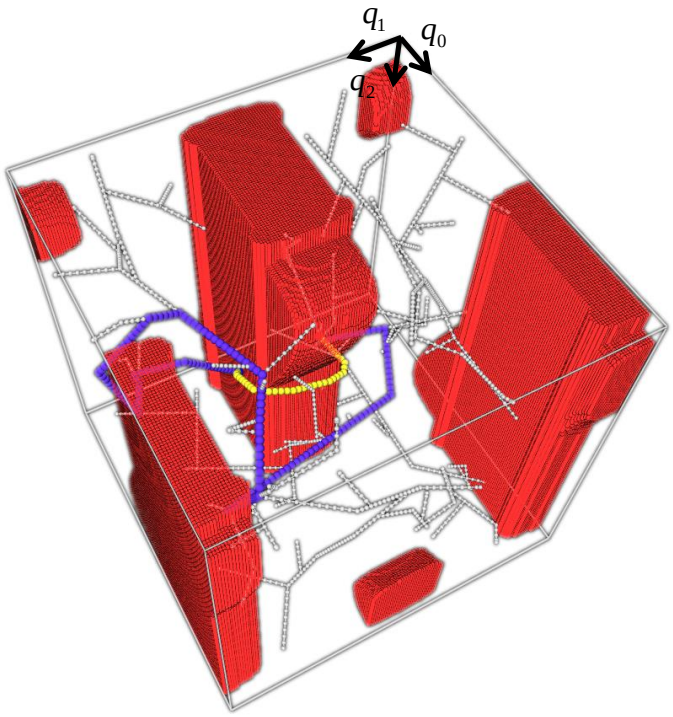


RRT: Beispiel

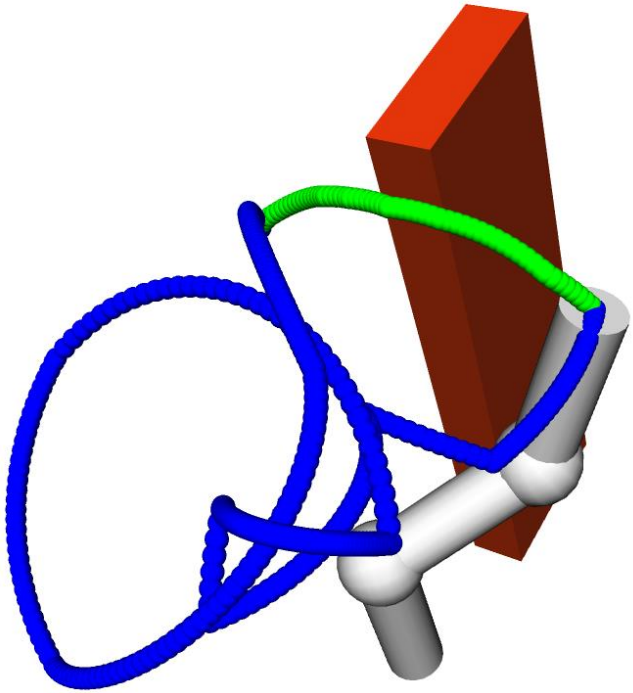
- Beispiel mit 3D Konfigurationsraum



Start



RRT im Konfigurationsraum
(rot: C_{obs})



Gefundene Lösung (blau) und
geglättete Lösungstrajektorie (grün)

Inhalt

- Motivation
- Grundlagen der Bewegungsplanung
- Pfadplanung für mobile Roboter
- Bewegungsplanung für Manipulatoren
  - Probabilistic Roadmaps (PRM)
  - Rapidly-exploring Random Trees (RRT)
  - Erweiterungen
    - Constrained RRT
    - RRT*
    - Dynamic Domain RRT
    - Bridge Sampling

Constrained RRT

- Bei der Bewegungsplanung müssen evtl. Nebenbedingungen (Constraints) erfüllt werden, z.B.:
  - Gleichbleibende Orientierung des Endeffektors
  - (statische) Stabilität eines zweibeinigen Roboters

- Problem: Nebenbedingungen können niederdimensionale Gebilde im Konfigurationsraum darstellen
  - z.B. Die Menge aller Konfigurationen $\boldsymbol{q}$, die eine Nebenbedingung erfüllen bilden eine Ebene im dreidimensionalen Konfigurationsraum
  - Sampling-basierte Ansätze können diese Nebenbedingungen prinzipiell nicht erfüllen

- Lösungsansätze:
  - Randomized Gradient Descent (RGD)
  - First Order Retraction (FR)

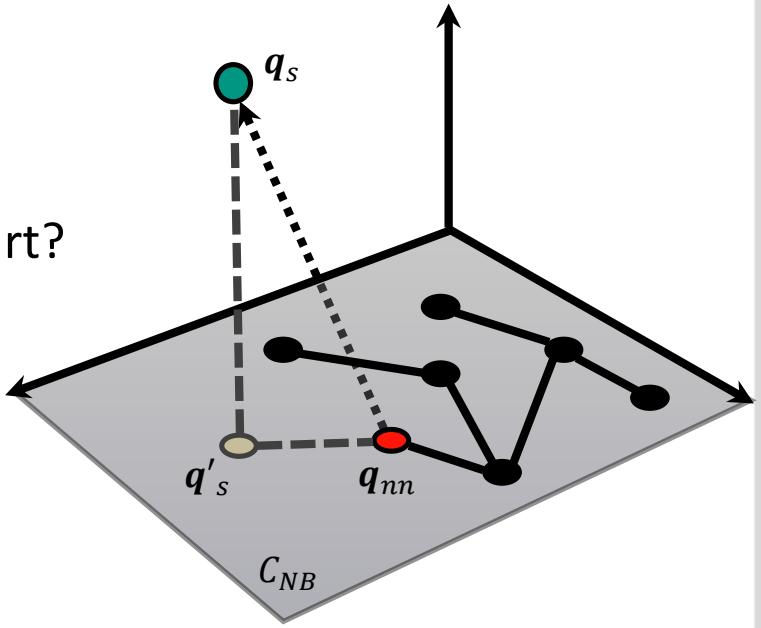
Constrained RRT

- Idee: Projiziere eine Stichprobe $\boldsymbol{q}_s$ auf eine Konfiguration $\boldsymbol{q}'_s$, die die Nebenbedingung erfüllt

- Beispiel: Eine Nebenbedingung NB bilde eine 2D Mannigfaltigkeit in $C_{\mathrm{NB}} \subseteq C$

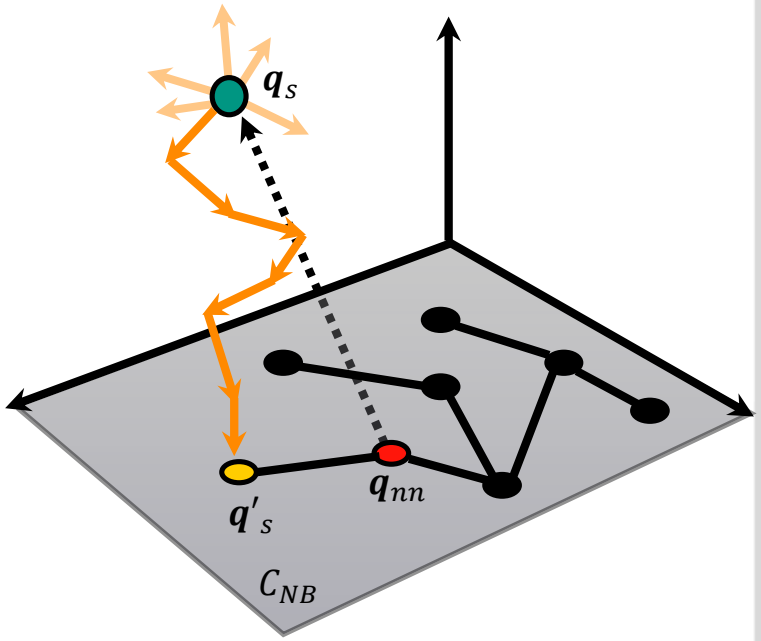
$$C_{NB} = \{\boldsymbol{q} \in C : \boldsymbol{q}\ erfüllt\ NB\}$$

- Problem: Wie wird die Projektion durchgeführt?
  - Randomized Gradient Descent
  - First Order Retraction



Constrained RRT: Randomized Gradient Descent

- Toleranzwert für Nebenbedingung: α
- Zufällige Bestimmung von n Nachbarn von $\boldsymbol{q}_s$ (in Hyperkugel mit Radius d_{max})
- Falls die Distanz eines Nachbarn zu C_{NB} kleiner als die Distanz von $\boldsymbol{q}_s$ zu C_{NB}, ersetze $\boldsymbol{q}_s$ mit diesem Nachbarn
- Wiederholen bis maximale Iterationszahl erreicht oder die Distanz von $\boldsymbol{q}_s$ zu C_{NB} kleiner ist als α
- Distanzmaß zu C_{NB} im Arbeitsraum (!) notwendig
- Keine Richtungsinformation notwendig

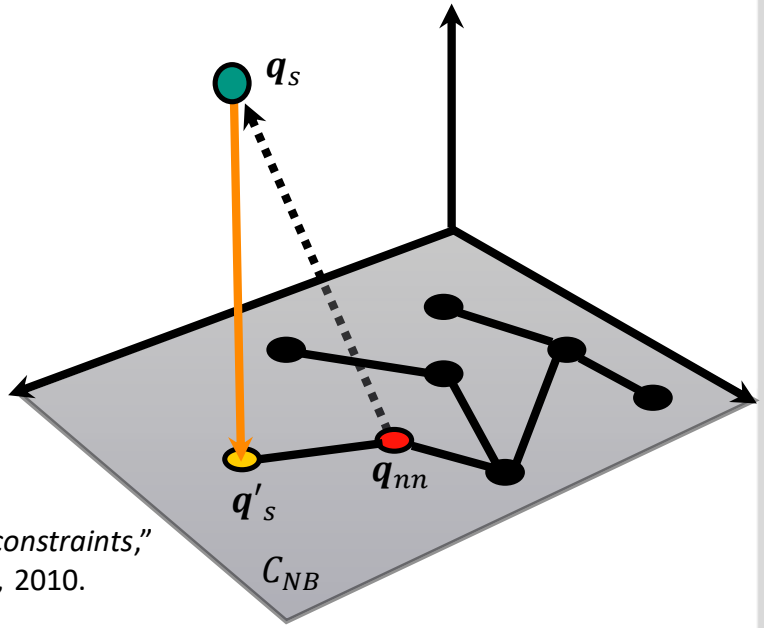


Constrained RRT: First Order Retraction

- Toleranzwert für die Nebenbedingung: α
- Jacobi-Matrix J liefert Richtungsinformation

- Berechnung wie bei Bestimmung der inversen Kinematik
  - $\boldsymbol{q}'_s = \boldsymbol{q}_s - J(\boldsymbol{q}_s)^{\#}\Delta\boldsymbol{x}_s$
  - $\Delta\boldsymbol{x}_s$ ist der Abstand von $\boldsymbol{q}_s$ zu C_{NB} im Arbeitsraum

- Distanzmaß zu C_{NB} im Arbeitsraum notwendig



M. Stilman, "*Global manipulation planning in robot joint space with task constraints*," IEEE Transactions on Robotics and Automation , vol. 26, no. 3, pp. 576–584, 2010.

Inhalt

- Motivation
- Grundlagen der Bewegungsplanung
- Pfadplanung für mobile Roboter
- Bewegungsplanung für Manipulatoren
  - Probabilistic Roadmaps (PRM)
  - Rapidly-exploring Random Trees (RRT)
  - Erweiterungen
    - Constrained RRT
    - RRT*
    - Dynamic Domain RRT
    - Bridge Sampling

RRT*

- **Problem:** RRTs finden Trajektorien, die üblicherweise nicht optimal sind
- **RRT*** optimiert den Suchbaum iterativ während der Suche
  - Mit ausreichender Zeit wird der optimale Pfad zwischen q_{start} und q_{ziel} gefunden ⇒ asymptotische Optimalität
- Optimierung des Suchbaums aufgeteilt in zwei Schritte
  - Ermittle zu jedem neuen Knoten die Kosten (zB Wegstrecke vom Startknoten)
  - Rewiring des Suchbaums beim Hinzugefügen neuer Knoten:
    Die Verbindungen des Suchbaumes werden in einer lokalen Umgebung des neuen Knoten optimiert (bzgl. der Wegkosten).
- **Nachteil:**
  - Uni-direktionaler Ansatz
  - Längere Laufzeiten (bis zu Faktor 30 im Vergleich zu uni-direktionalem RRT)

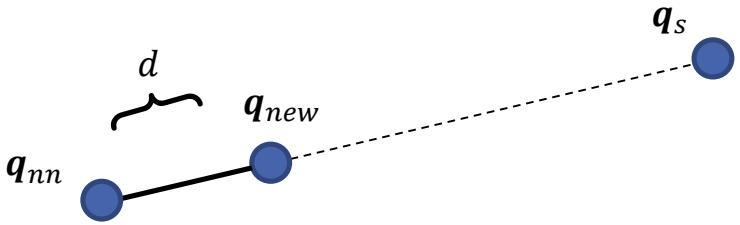
S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning.
The International Journal of Robotics Research, 30(7):846–894, Jan. 2011.

RRT*: Algorithmus

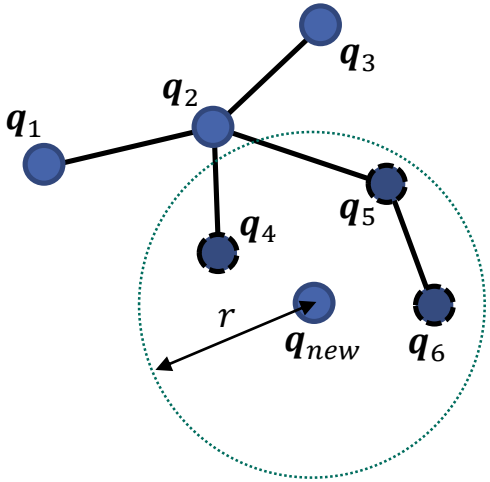
1. $\boldsymbol{q}_s = SampleRandom(C)$ // Erzeuge zufällige Konfiguration
2. $\boldsymbol{q}_{nn} = NearestNeighbor(\boldsymbol{q}_s, T)$ // Bestimme nächsten Nachbar
3. $\boldsymbol{q}_{new} = Steer(\boldsymbol{q}_{nn}, \boldsymbol{q}_s, d)$ // Gehe einen Schritt in Richtung $\boldsymbol{q}_s$
4. $if\ !CollisionFreePath(\boldsymbol{q}_{nn}, \boldsymbol{q}_{new}): goto\ 1$ // Kollisionsfreier Pfad?
5. $Q_{near} = Near(T, \boldsymbol{q}_{new}, r)$ // Alle Punkte mit max. Abstand r zu $\boldsymbol{q}_{new}$
6. $\boldsymbol{q}_{min} = MinCostPath(Q_{near}, \boldsymbol{q}_{new})$ // $Cost(\boldsymbol{q}_{min}) + Cost(\boldsymbol{q}_{min}, \boldsymbol{q}_{new})$ minimal
7. $AddPath(T, \boldsymbol{q}_{min}, \boldsymbol{q}_{new})$ // Füge Pfad von $\boldsymbol{q}_{min}$ zu $\boldsymbol{q}_{new}$ hinzu
8. $Rewire(T, \boldsymbol{q}_{new}, Q_{near})$ // Überprüfe Kanten zu Knoten in Q_{near}
9. $if\ !Timeout: goto\ 1$ // Nächste Iteration

RRT*: Funktionen

- $q_{new} = Steer(q_{nn}, q_s, d)$
  - Erzeuge Knoten q_{new}
  - Gehe von q_{nn} in Richtung q_s
  - Schrittweite d



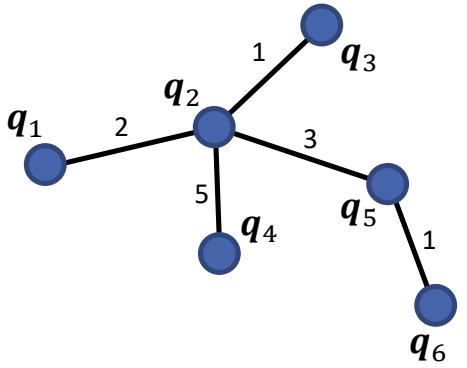
- $Q_{near} = Near(T, q_{new}, r)$
  - Bestimme alle Knoten aus T deren Abstand zu q_{new} maximal r beträgt



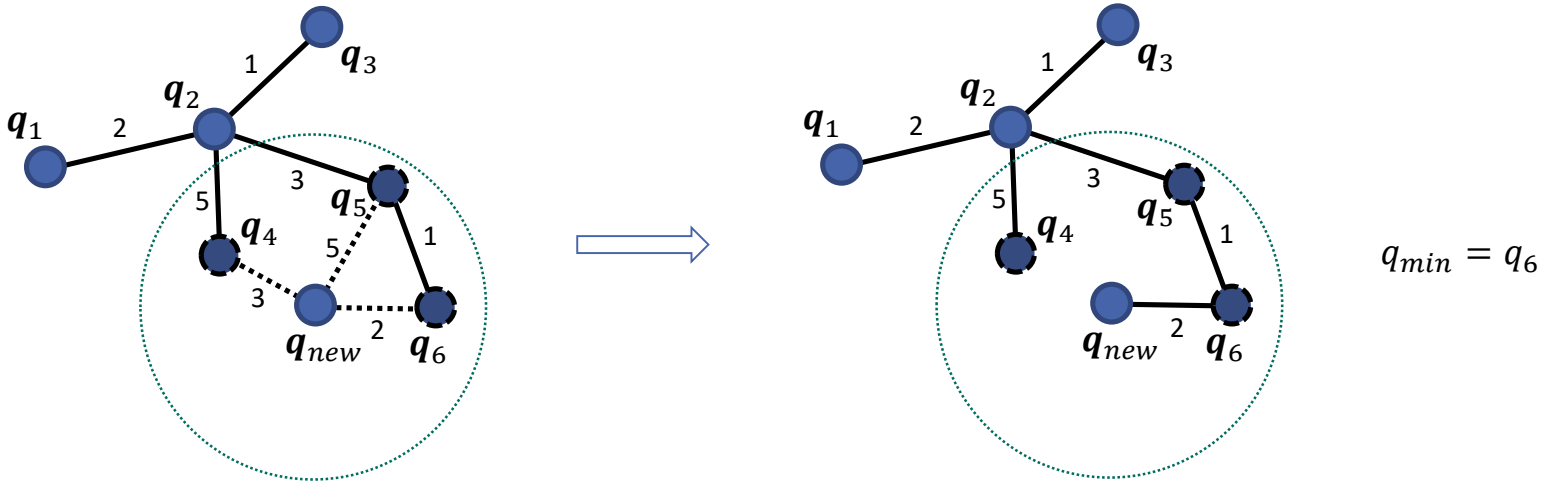
$Q_{near} = \{q_4, q_5, q_6\}$

RRT*: Funktionen (II)

- $Cost(\boldsymbol{q}_i)$
  - Pfadkosten von $\boldsymbol{q}_{start}$ zu $\boldsymbol{q}_i$
- $Cost(\boldsymbol{q}_a, \boldsymbol{q}_b)$
  - Kosten der Verbindung von $\boldsymbol{q}_a$ zu $\boldsymbol{q}_b$
- $\boldsymbol{q}_{min} = MinCostPath(Q_{near}, \boldsymbol{q}_{new})$
  - Bestimme $\boldsymbol{q}_{min} \in Q_{near}$ so dass Pfadkosten $Cost(\boldsymbol{q}_{min})$+$Cost(\boldsymbol{q}_{min}, \boldsymbol{q}_{new})$ minimal sind (sowie kollisionsfrei)



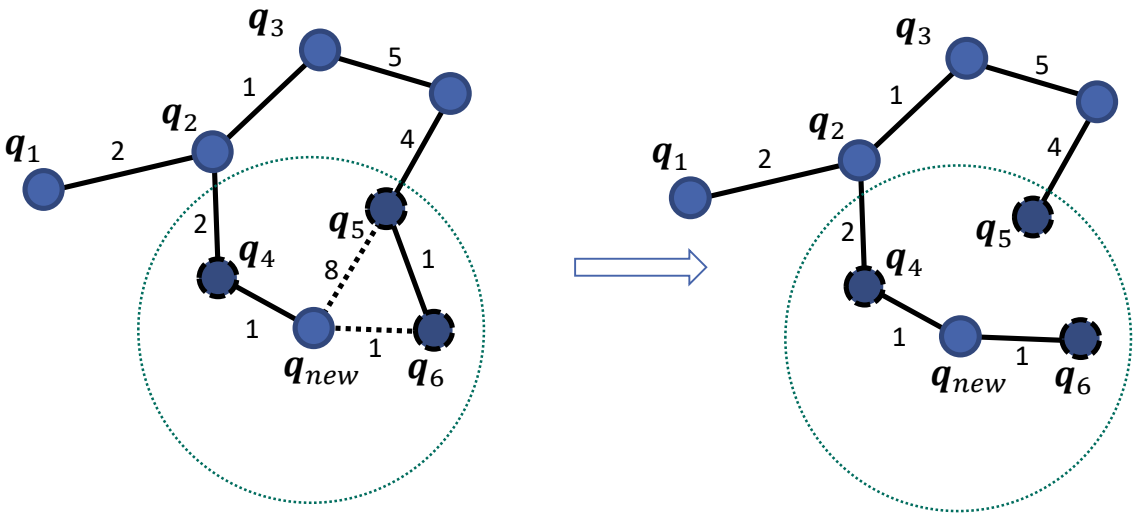
$Cost(\boldsymbol{q}_6) = 2 + 3 + 1 = 6$



$q_{min} = q_6$

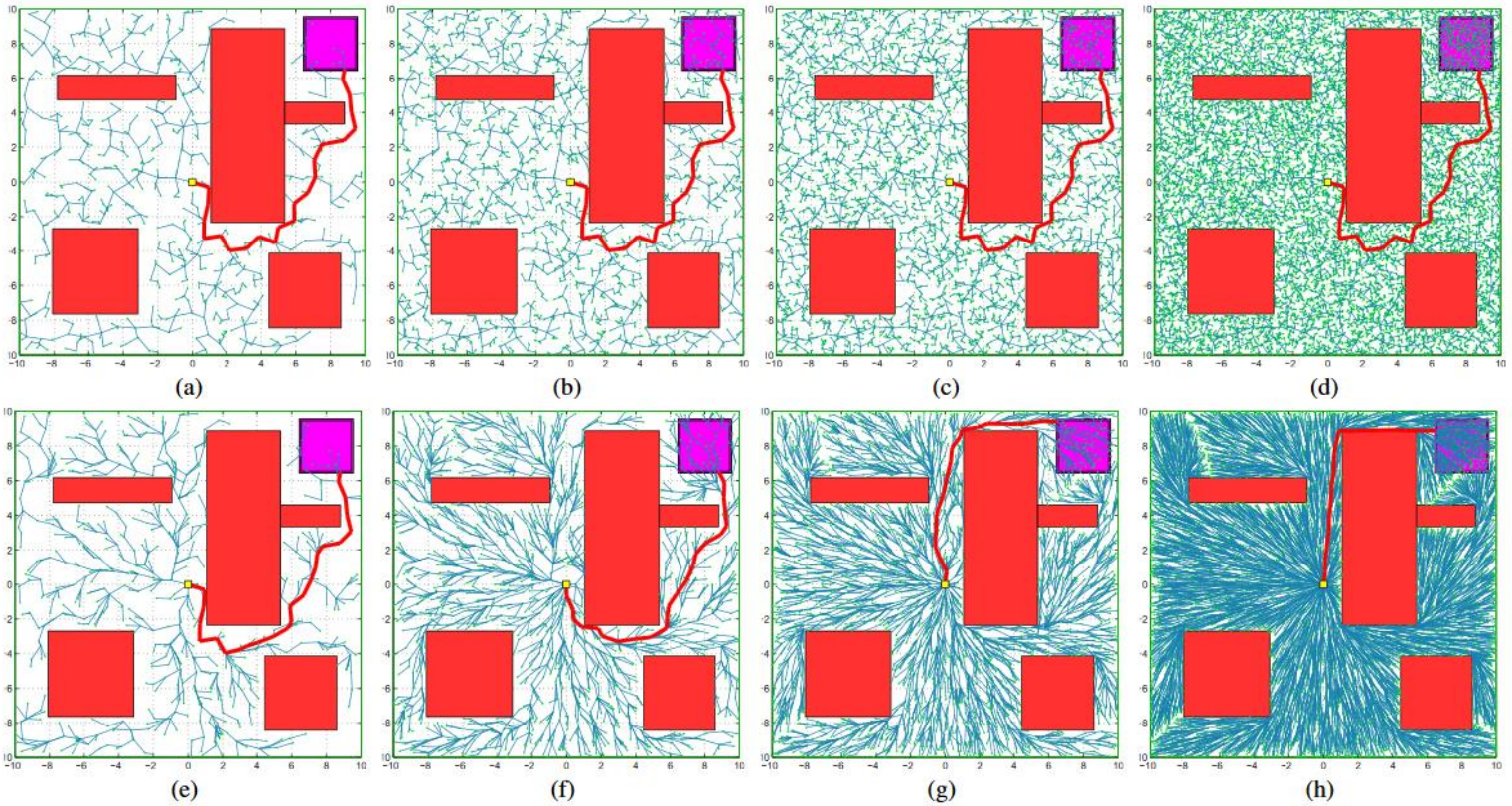
RRT*: Rewiring

- $Rewire(T, \boldsymbol{q}_{new}, Q_{near})$
  - Überprüfe für alle $\boldsymbol{q}_{near} \in Q_{near}$ ob
    $$Cost(\boldsymbol{q}_{new}) + Cost(\boldsymbol{q}_{new}, \boldsymbol{q}_{near}) < Cost(\boldsymbol{q}_{near})$$
    - Ersetze ggf. Verbindung zu $\boldsymbol{q}_{near}$ (falls günstiger und kollisionsfrei)



$$Cost(q_5) = 12$$
$$Cost(q_6) = 13$$
$$Cost(q_{new}) = 5$$
$$Cost(q_{new}) + Cost(q_{new}, q_5) = 13$$
$$\boldsymbol{Cost(q_{new}) + Cost(q_{new}, q_6) = 6}$$

Vergleich RRT und RRT*

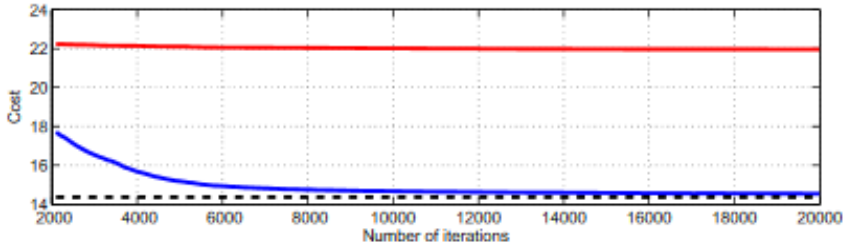


Iteration	1000	2500	5000	15000

S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, 30(7):846–894, Jan. 2011.

Vergleich RRT und RRT* (II)

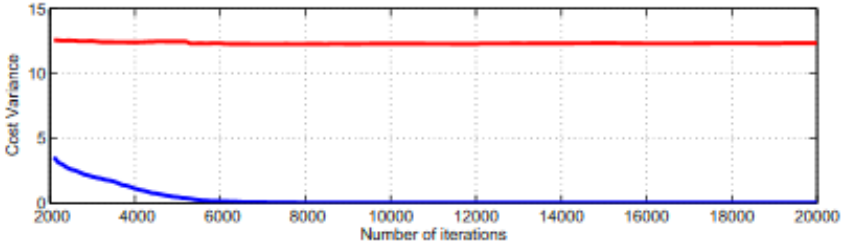
Pfadkosten der gefundenen Lösung
(Optimale Lösung: schwarz)



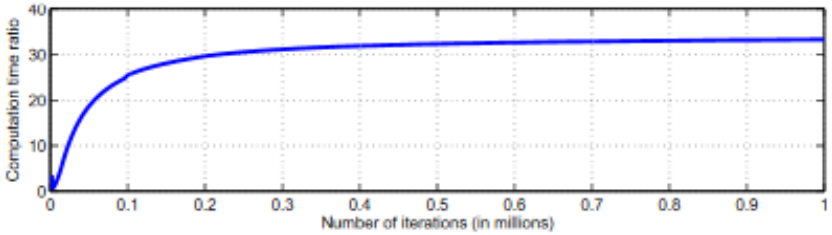
RRT

RRT*

Varianz der Pfadkosten



Verhältnis der Laufzeit
(angegeben für eine Iteration)



S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, 30(7):846–894, Jan. 2011.

Inhalt

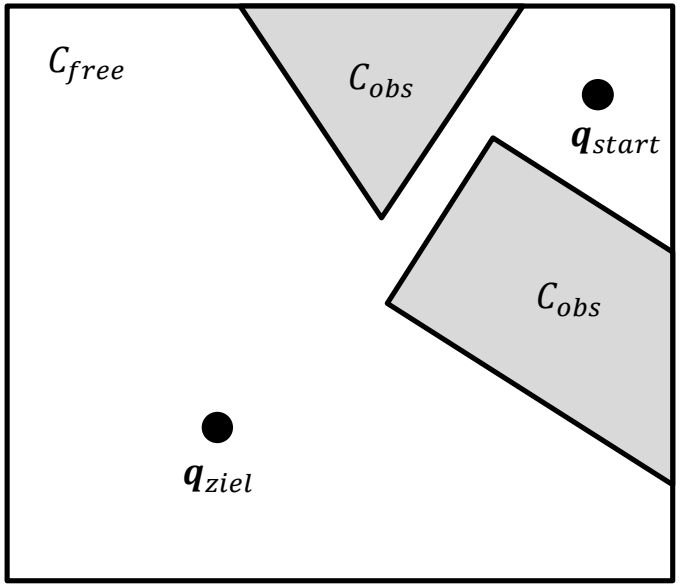
- Motivation
- Grundlagen der Bewegungsplanung
- Pfadplanung für mobile Roboter
- Bewegungsplanung für Manipulatoren
  - Probabilistic Roadmaps (PRM)
  - Rapidly-exploring Random Trees (RRT)
  - Erweiterungen
    - Constrained RRT
    - RRT*
    - Enge Passagen
      - Dynamic Domain RRT
      - Bridge Sampling

Enge Passagen: Motivation

- Klassiche RRTs bestimmen neue Punkte $\boldsymbol{q}_s$ durch gleichverteilte Zufallswahl im Konfigurationsraum C
- Ergebnis gleichverteilter Zufallswahl:
  - Viele eher uninteressante Stichproben
    z.B. „mitten in C_{free}“, weit entfernt von Hindernissen
  - Wenige interessante Stichproben
    z.B. nahe bei Hindernissen, insbesondere in engen Passagen zwischen zwei Hindernissen
- Klassische RRTs können viel Zeit benötigen, bis eine Lösung für einen Durchgang durch eine enge Passage gefunden wurde
- **Hauptidee der Verfahren:**
  Sampling ist deutlich günstiger als Kollisionsprüfung von Pfaden im Baum

Enge Passagen: Beispiel

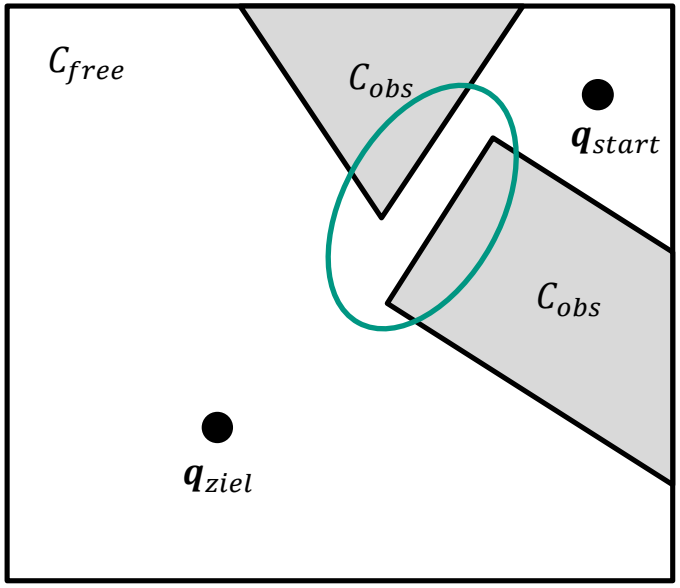
- Beispiel für enge Passagen in einem 2D Konfigurationsraum



Enge Passagen: Beispiel

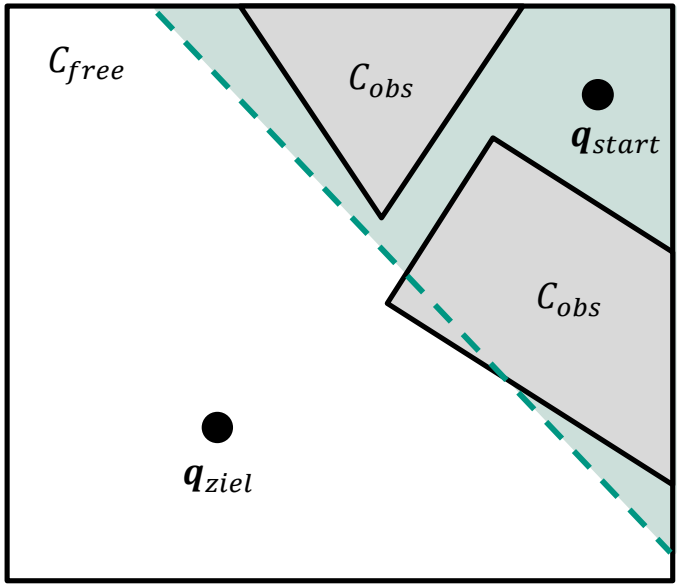
- Beispiel für enge Passagen in einem 2D Konfigurationsraum

- Geringe Wahrscheinlichkeit, dass neue Stichproben im Bereich der engen Passage liegen



Enge Passagen: Dynamic Domain RRT I

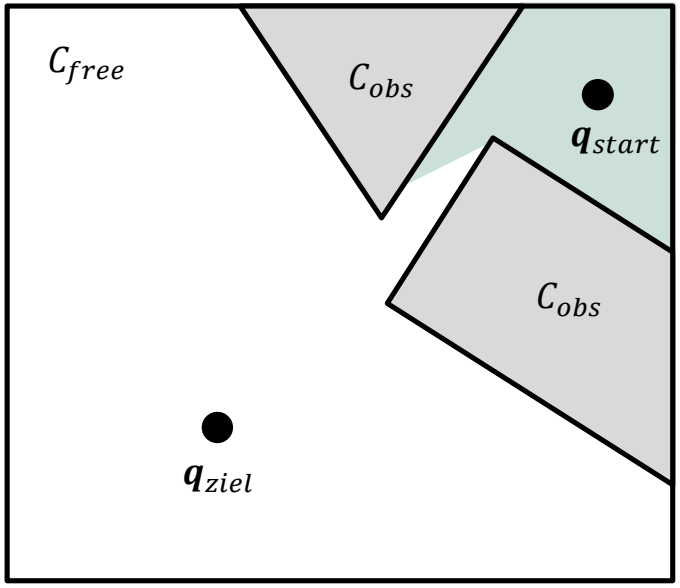
- **Problem:** RRTs erkennen enge Passagen nicht und können nicht zielgerichtet sampeln
- **Ideal:** Nur in sichtbarer Voronoi Region eines Knotens sampeln



Voronoi-Region zu q_{start}

Enge Passagen: Dynamic Domain RRT (II)

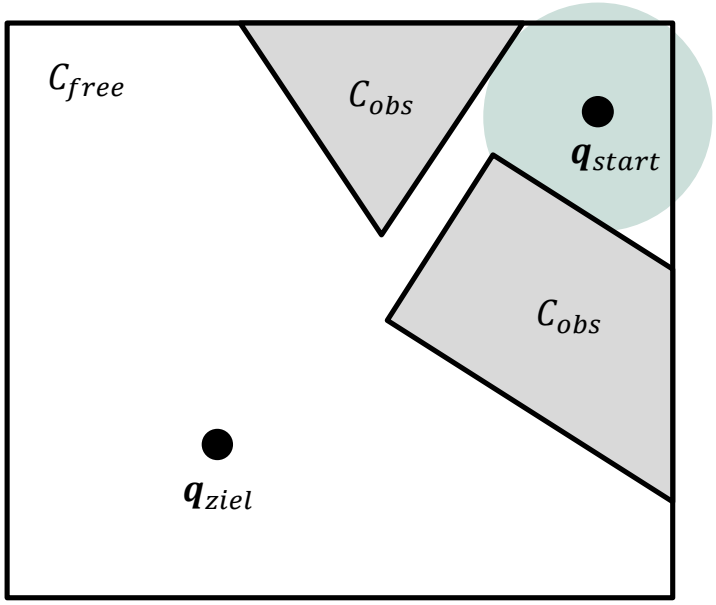
- **Problem:** RRTs erkennen enge Passagen nicht und können nicht zielgerichtet sampeln
- **Ideal:** Nur in sichtbarer Voronoi Region eines Knotens sampeln



Sichtbare Voronoi-Region zu q_{start}

Enge Passagen: Dynamic Domain RRT (III)

- **Aber:** Berechnung sichtbarer Voronoi Regionen ist aufwendig (keine explizite Darstellung der Hindernissregionen im Konfigurationsraum)
- **Stattdessen:** Approximation sichtbarer Voronoi Regionen durch Kugeln mit Radius r (**Dynamic Domain**)



Approximierte sichtbare Voronoi-Region zu q_{start} (Dynamic Domain)

Dynamic Domain RRT (IV)

- **Dynamic Domain RRT** beschränkt in der Nähe von Hindernissen die Sampling Domäne eines Knotens auf dessen Dynamic Domain (DD).
  - Initial wird der DD-Radius r jedes Knotens auf ∞ gesetzt.
    Sampling findet in gesamter Voronoi Region des Knotens statt.
  - Wenn während des RRT-Erweiterungsschritts keine Verbindung zu einem Knoten hergestellt werden kann, wird dessen DD-Radius auf einen festgelegten Wert R reduziert.
    Knoten dieser Art werden **Grenzknoten** genannt da sie an der Grenze von C_{free} und C_{obs} liegen.
- **Sampling**
  - Ein Sample $\boldsymbol{q_s}$ wird verworfen, falls $\boldsymbol{q_s}$ außerhalb des DD-Radius seines nächsten Nachbarn liegt ($dist(\boldsymbol{q_s}, \boldsymbol{q_{nn}}) > \boldsymbol{q_{nn}}.r$)

A. Yershova, L. Jaillet, T. Simeon, and S. M. LaValle. Dynamic-Domain RRTs: Efficient Exploration by Controlling the Sampling Domain. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 3856–3861, Apr. 2005.

Dynamic Domain RRT (V)

- Bei engen Passagen werden häufige Kollisionsprüfungen vermieden und keine Expansionsversuche zu weit entfernten und unerreichbaren Knoten unternommen.

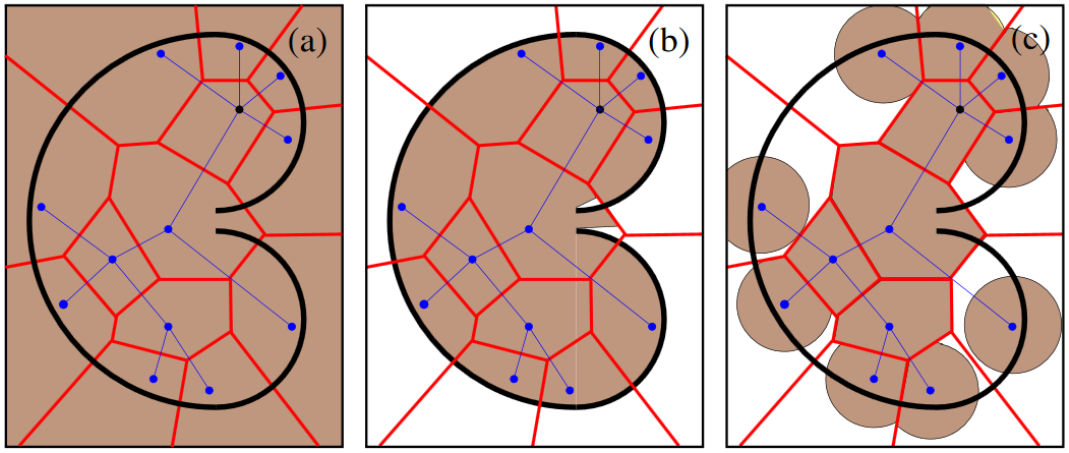
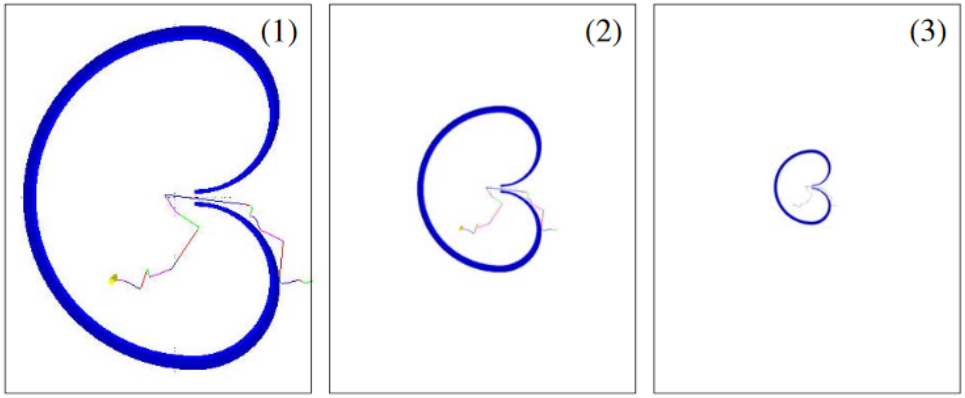


Fig. 3. For a set of points inside a bug trap different sampling domains are shown: (a) regular RRTs sampling domain, (b) visibility Voronoi region, (c) dynamic domain.

A. Yershova, L. Jaillet, T. Simeon, and S. M. LaValle. Dynamic-Domain RRTs: Efficient Exploration by Controlling the Sampling Domain. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 3856–3861, Apr. 2005.

Dynamic Domain RRT – Vergleich zu RRT

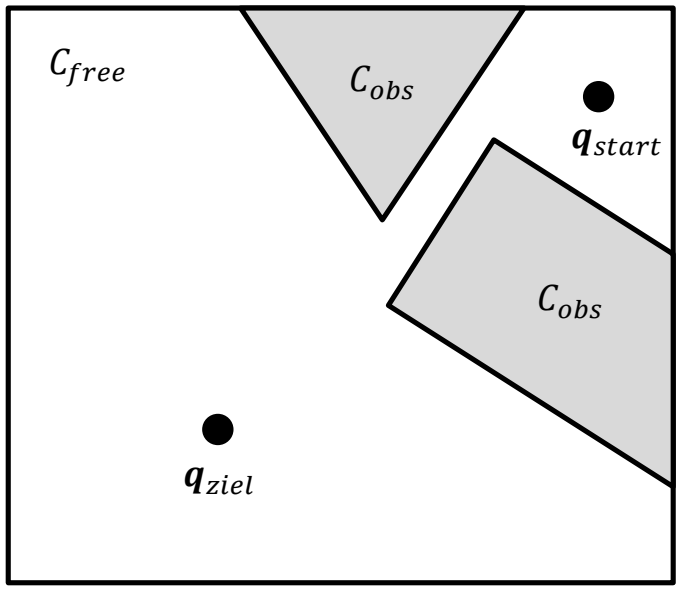


	Dynamic-Domain bi-RRT	bi-RRT
time (1)	0.4 sec	0.1 sec
no. nodes (1)	253	37
CD calls (1)	618	54
time (2)	2.5 sec	379 sec
no. nodes (2)	1607	6924
CD calls (2)	3751	781530
time (3)	1.6 sec	$>$ 80000 sec
no. nodes (3)	1301	–
CD calls (3)	3022	–

A. Yershova, L. Jaillet, T. Simeon, and S. M. LaValle. Dynamic-Domain RRTs: Efficient Exploration by Controlling the Sampling Domain. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 3856–3861, Apr. 2005.

Enge Passagen: Bridge Sampling I

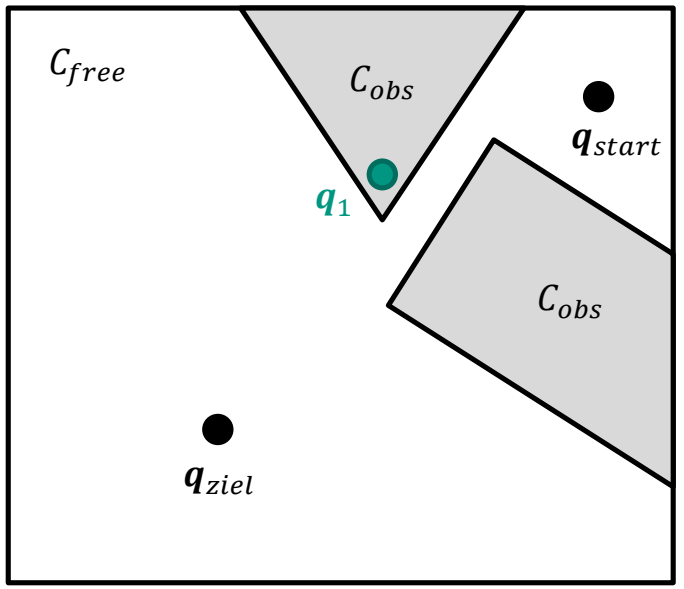
- Idee: Wähle zielgerichtet Punkte in engen Passagen für die nächste Stichprobe



Enge Passagen: Bridge Sampling I

- Idee: Wähle zielgerichtet Punkte in engen Passagen für die nächste Stichprobe

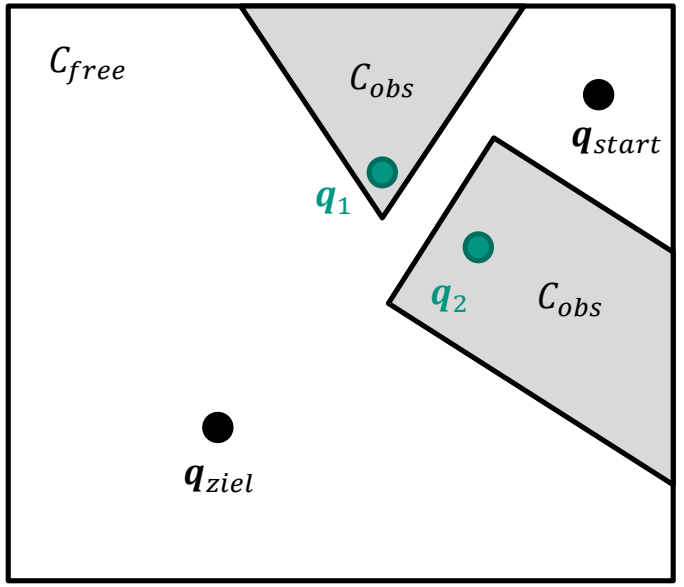
- Vorgehen:
  1. Wähle gleichverteilt einen zufälligen Punkt $\boldsymbol{q}_1 \in C_{obs}$



Enge Passagen: Bridge Sampling I

- Idee: Wähle zielgerichtet Punkte in engen Passagen für die nächste Stichprobe

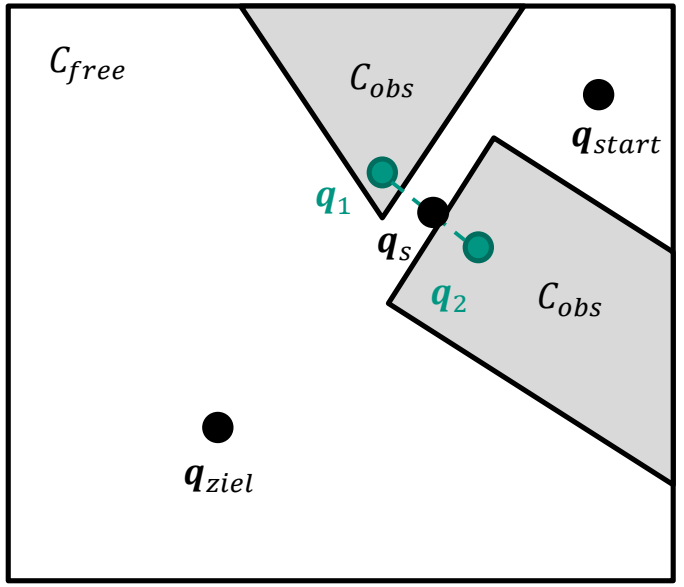
- Vorgehen:
  1. Wähle gleichverteilt einen zufälligen Punkt $\boldsymbol{q}_1 \in C_{obs}$
  2. Wähle nach einer geeigneten Wahrscheinlichkeitsverteilung einen zweiten Punkt $\boldsymbol{q}_2 \in C_{obs}$ in der Nähe von $\boldsymbol{q}_1$



Enge Passagen: Bridge Sampling I

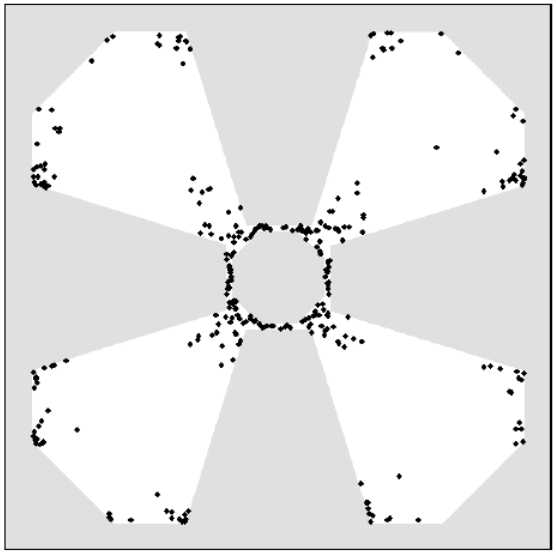
- Idee: Wähle zielgerichtet Punkte in engen Passagen für die nächste Stichprobe

- Vorgehen:
  1. Wähle gleichverteilt einen zufälligen Punkt $\boldsymbol{q}_1 \in C_{obs}$
  2. Wähle nach einer geeigneten Wahrscheinlichkeitsverteilung einen zweiten Punkt $\boldsymbol{q}_2 \in C_{obs}$ in der Nähe von $\boldsymbol{q}_1$
  3. Wenn der Mittelpunkt $\boldsymbol{q}_s$ zwischen $\boldsymbol{q}_1$ und $\boldsymbol{q}_2$ in C_{free} liegt, dann verwende ihn als neue Stichprobe für den RRT (oder die PRM)
  4. Wiederhole



Enge Passagen: Bridge Sampling II

- Bridge Sampling erhöht die Stichprobendichte in interessanten Bereichen des Konfigurationsraumes C
  - Interessant sind Bereiche in der Nähe von Hindernissen, besonders in engen Passagen
- Bridge Sampling kann für RRTs und PRMs verwendet werden
- Das zentrale Element des Verfahrens, der Bridge Test, ist auch in hochdimensionalen Räumen effizient berechenbar



Stichproben beim Bridge Sampling (Sun et al., 2005)

Z. Sun, D. Hsu, T. Jiang, H. Kurniawati, and J. Reif, "*Narrow passage sampling for probabilistic roadmap planning*,"IEEE Transactions on Robotics, vol. 21, no. 6, pp. 1105–1115, 2005

Englische Begriffe

Deutsch	Englisch
Bestensuche	Best-first search
Enge Passagen	Narrow passages
Kern	Kernel
Nebenbedingung	Constraint
Projektion	Projection
Stichprobe	Sampling
Vorgängerknoten	Predecessor
Zulässig	Admissible