Robotika: Plánování pohybu

Vladimír Petrík

vladimir.petrik@cvut.cz

06.11.2023

Motivace

- Představte si úlohu, kde máte pomocí robota přenést plný hrnek z jednoho místa na druhé
- Jak byste to řešili pomocí metod, které znáte?
  - Spojit úsečkou počáteční a koncový bod - interpolace polohy
  - Zafixovat orientaci hrnku
  - Výpočet inverzní kinematiky

Motivace

- Představte si úlohu, kde máte pomocí robota přenést plný hrnek z jednoho místa na druhé
- Jak byste to řešili pomocí metod, které znáte?
  - Spojit úsečkou počáteční a koncový bod - interpolace polohy
  - Zafixovat orientaci hrnku
  - Výpočet inverzní kinematiky
- Možné problémy
  - Inverzní kinematika spočítá několik řešení
  - Inverzní kinematika nemusí najít žádné řešení
  - Překážka v přímé cestě

Motivace

- Představte si úlohu, kde máte pomocí robota přenést plný hrnek z jednoho místa na druhé
- Jak byste to řešili pomocí metod, které znáte?
  - Spojit úsečkou počáteční a koncový bod - interpolace polohy
  - Zafixovat orientaci hrnku
  - Výpočet inverzní kinematiky
- Možné problémy
  - Inverzní kinematika spočítá několik řešení
  - Inverzní kinematika nemusí najít žádné řešení
  - Překážka v přímé cestě
- Definovat přípustné natočení a použít plánování

Cíl dnešní přednášky

- Plánování pohybu
  - Co je to plánování?
  - Plánování v mobilní robotice
  - Plánování pro 2D manipulátor
  - Plánování pro reálný 7D manipulátor
- Založeno na 10. kapitole Lynch&Park: Modern Robotics

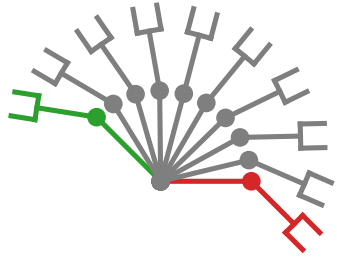
Formulace problému

- Plánování pohybu pro robota z daného startu do daného cíle
  - Vyhýbání překážkám
  - Respektování omezení kloubů
  - Respektování maximální rychlosti, sil/momentů
  - Ostatní omezení daného prostředí (např. nevylít vodu z hrnku $\rightarrow$ omezení orientace)

[1] Existují *task-space* plánovače, ale nebudeme se jimi dnes zabývat.

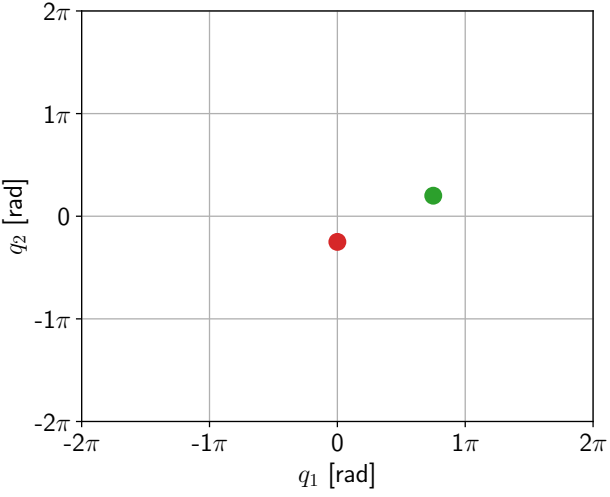
Formulace problému

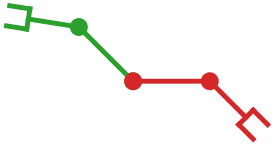
- Plánování pohybu pro robota z daného startu do daného cíle
  - Vyhýbání překážkám
  - Respektování omezení kloubů
  - Respektování maximální rychlosti, sil/momentů
  - Ostatní omezení daného prostředí (např. nevylít vodu z hrnku $\rightarrow$ omezení orientace)
- Plánování probíha v konfiguračním prostoru robota[1]
  - Konfigurační prostor $\mathcal{C}$ (Configuration space, C-space)
  - Konfigurace robota $\boldsymbol{q} \in \mathcal{C}$
  - Start i Cíl je definován v konfiguračním prostoru $(\boldsymbol{q}_s, \boldsymbol{q}_g) \in \mathcal{C} \times \mathcal{C}$



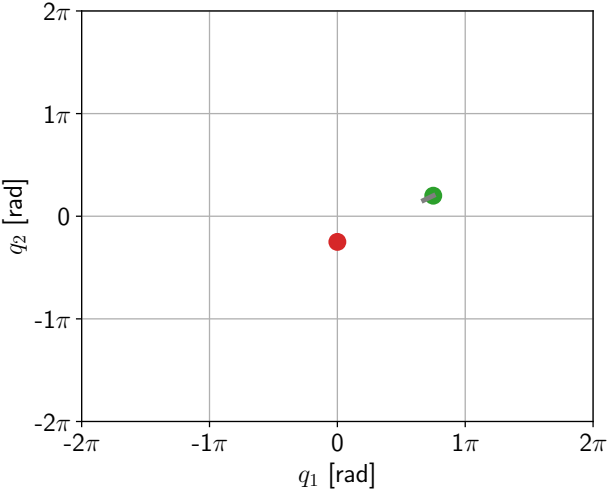
[1] Existují *task-space* plánovače, ale nebudeme se jimi dnes zabývat.

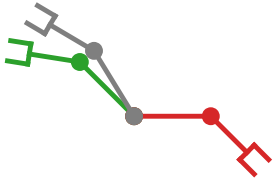
Konfigurační prostor - interpolace



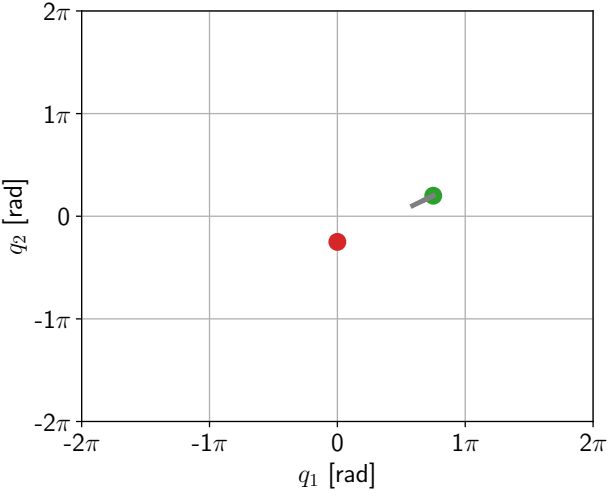


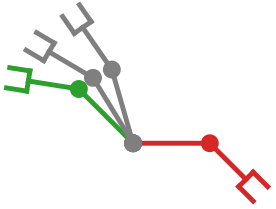
Konfigurační prostor - interpolace



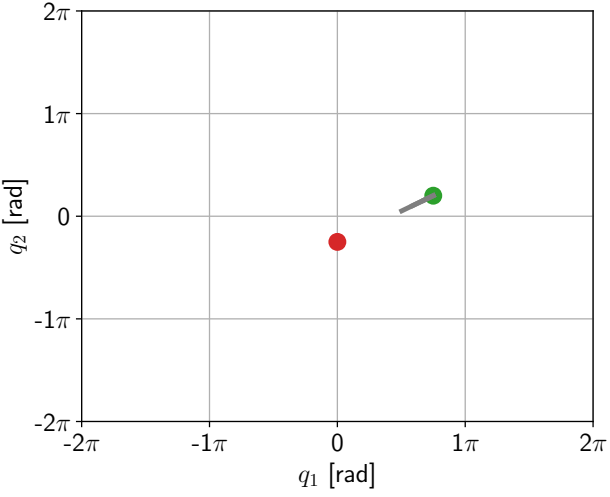


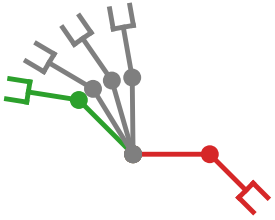
Konfigurační prostor - interpolace



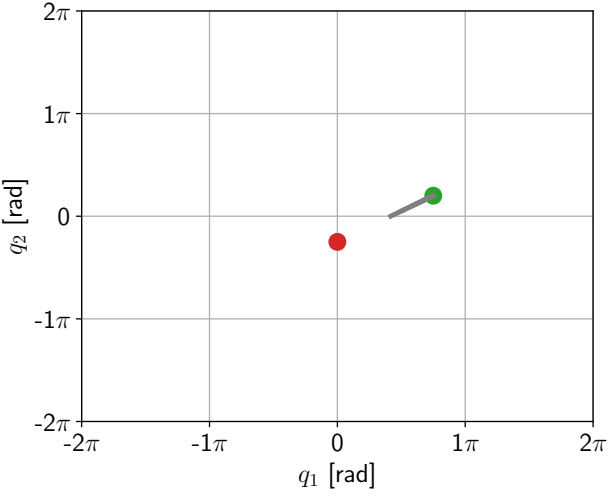


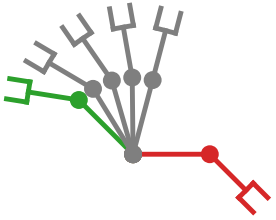
Konfigurační prostor - interpolace



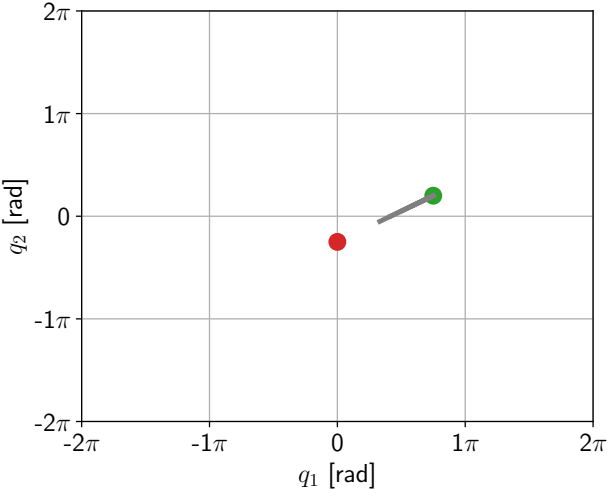


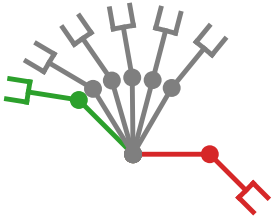
Konfigurační prostor - interpolace



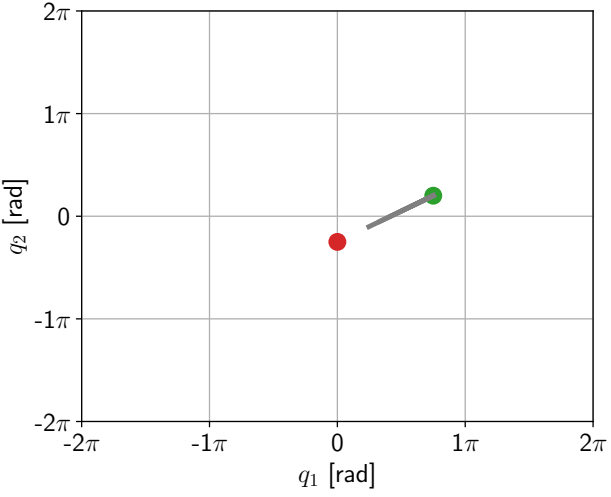


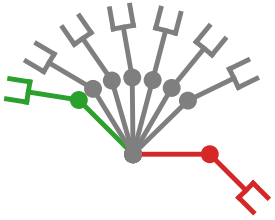
Konfigurační prostor - interpolace



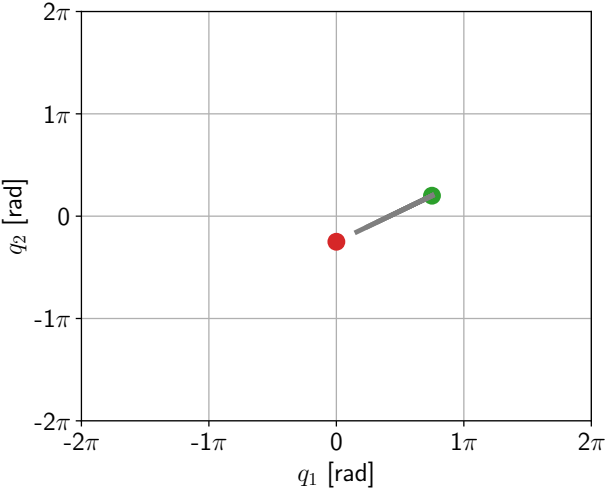


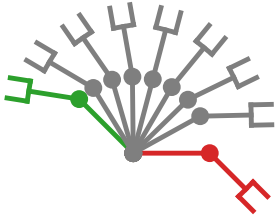
Konfigurační prostor - interpolace



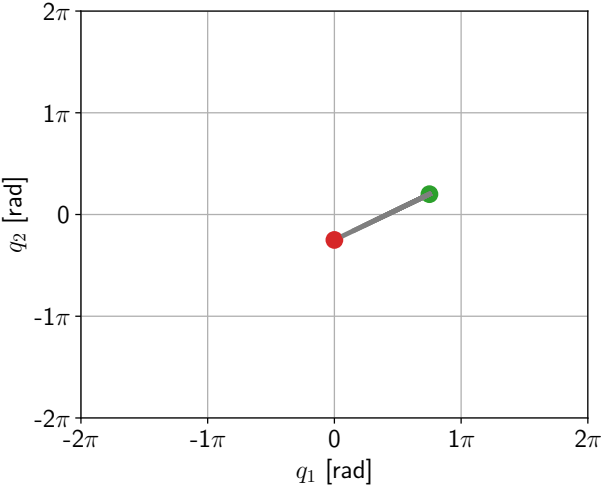


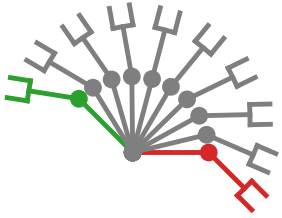
Konfigurační prostor - interpolace



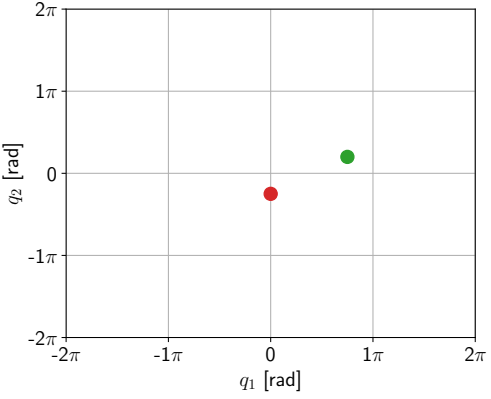


Konfigurační prostor - interpolace



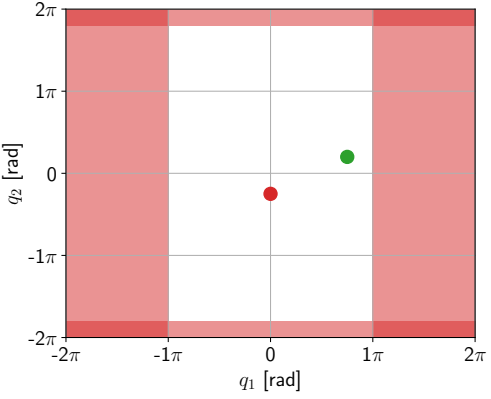


Vliv omezení na konfigurační prostor



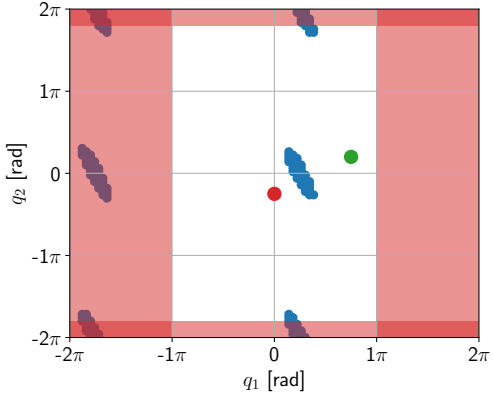
Vliv omezení na konfigurační prostor

- Limity kloubů



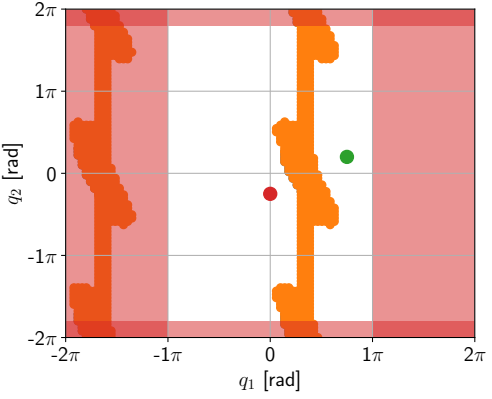
Vliv omezení na konfigurační prostor

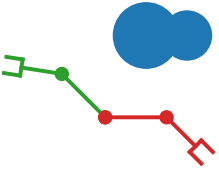
- Limity kloubů
- Překážky prostředí (pozn. tvar chapadla ignorován)



Vliv omezení na konfigurační prostor

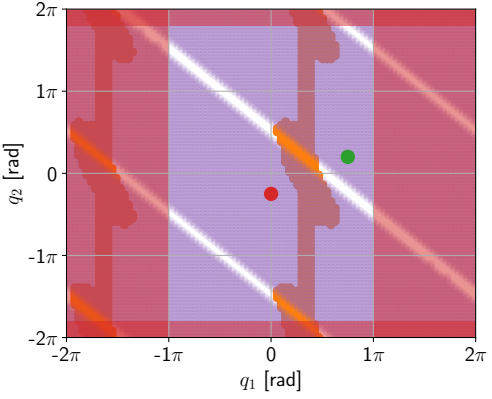
- Limity kloubů
- Překážky prostředí (pozn. tvar chapadla ignorován)

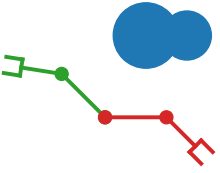




Vliv omezení na konfigurační prostor

- Limity kloubů
- Překážky prostředí (pozn. tvar chapadla ignorován)
- Omezení orientace chapadla ($80° < \theta < 100°$)





Konfigurační prostor - závěr

- Konfigurační prostor rozdělujeme na:
  - Prostor přípustných konfigurací $\mathcal{C}_{\text{free}}$
  - Prostor nepřípustných konfigurací $\mathcal{C}_{\text{occ}}$
  - $\mathcal{C} = \mathcal{C}_{\text{free}} \cup \mathcal{C}_{\text{occ}}$
- Konfigurační prostor je (většinou) težké popsat explicitně
- I když Start i Cíl leží v $\mathcal{C}_{\text{free}}$, nemusí existovat pohyb, který je spojuje

Co je to pohyb? Cesta × Trajektorie

- Cesta (path)
  - Geometrický popis (sekvence konfigurací)
  - Bez časových značek, dynamiky a omezení na kontrolní signál
  - Najdi $\boldsymbol{q}(s) \in \mathcal{C}_{\text{free}}, s \in [0,1]$ tak, aby:
  - $\boldsymbol{q}(0) = \boldsymbol{q}_s\,, \quad \boldsymbol{q}(1) = \boldsymbol{q}_g$
  - Hlavním předpokladem je, že následným zpracovaním je možné získat trajektorii
- Trajektorie (trajectory)
  - Popisuje pozici robota v čase
  - $\boldsymbol{q}(t) \in \mathcal{C}_{\text{free}}, t \in [0,T]$

Můžeme plánovat rovnou trajektorii?

- Definice problému zahrnuje rovnici pohybu systému
- Najdi kontrolní signál v čase $\boldsymbol{u}(t)$ tak, aby:
- $\boldsymbol{q}(0) = \boldsymbol{q}_s\,, \quad \boldsymbol{q}(T) = \boldsymbol{q}_g\,, \quad \dot{\boldsymbol{q}} = f(\boldsymbol{q}(t), \boldsymbol{u}(t))$
- Komplikovanější úloha než hledání cesty
- Kdy je preferovaná?
  - jsou poháněné jenom některé klouby - nemusí být možné sledovat naplánovanou cestu
- V robotickém plánování pohybu se typicky naplánuje cesta a z ní se spočítá trajektorie

Plánování cesty v mobilní robotice

- Jak plánování funguje v ROS: MoveBase
- Robot se pohybuje v rovině $(v_x, v_\theta)^\top$
- Známe:
  - rozměry robota (půdorys)
  - mapu prostředí
  - pozici robota v mapě $(x, y, \theta)^\top$
  - cílovou pozici v mapě

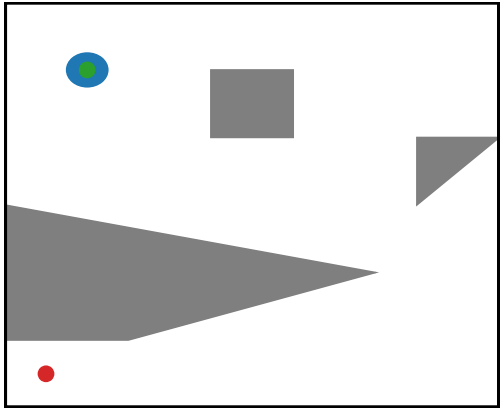
Plánování cesty v mobilní robotice

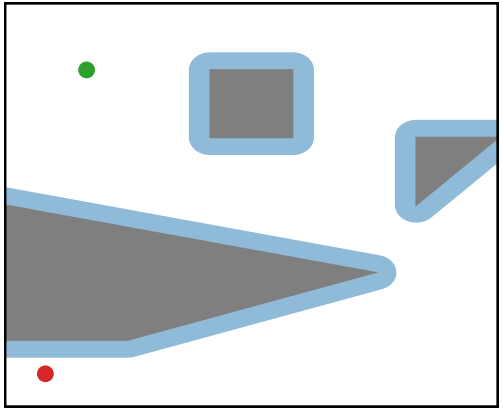
- Jak plánování funguje v ROS: MoveBase
- Robot se pohybuje v rovině $(v_x, v_\theta)^\top$
- Známe:
  - rozměry robota (půdorys)
  - mapu prostředí
  - pozici robota v mapě $(x, y, \theta)^\top$
  - cílovou pozici v mapě
- Řešení navigace do zadaného cíle v ROS: MoveBase:
  - Globální plánovač cesty ve 2D
  - Dopočítání orientace (robot směřuje dopředu podél cesty - proč?)
  - Lokální plánovač (vyhýbání dynamickým překážkám, používá vstup z lidaru)
  - Poslání příkazu robotovi

Plánování cesty v mobilní robotice

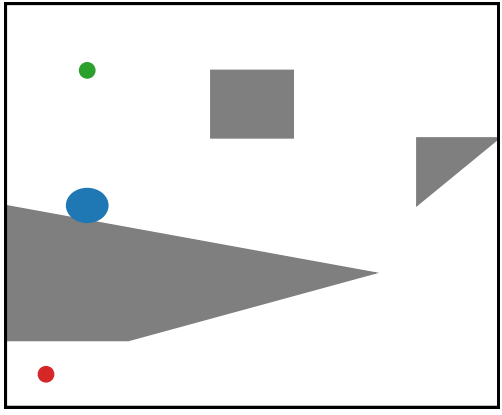
- Jak plánování funguje v ROS: MoveBase
- Robot se pohybuje v rovině $(v_x, v_\theta)^\top$
- Známe:
  - rozměry robota (půdorys)
  - mapu prostředí
  - pozici robota v mapě $(x, y, \theta)^\top$
  - cílovou pozici v mapě
- Řešení navigace do zadaného cíle v ROS: MoveBase:
  - **Globální plánovač cesty ve 2D**
  - Dopočítání orientace (robot směřuje dopředu podél cesty - proč?)
  - Lokální plánovač (vyhýbání dynamickým překážkám, používá vstup z lidaru)
  - Poslání příkazu robotovi

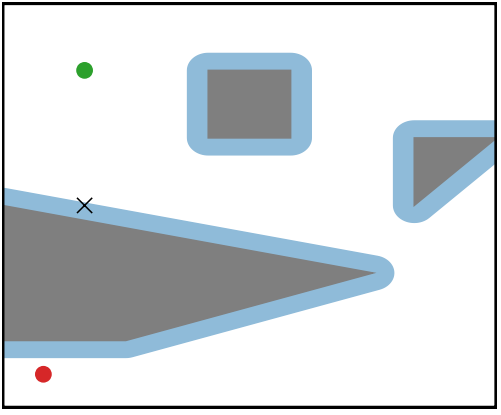
Jak vypadá mapa a konfigurační prostor





Jak vypadá mapa a konfigurační prostor

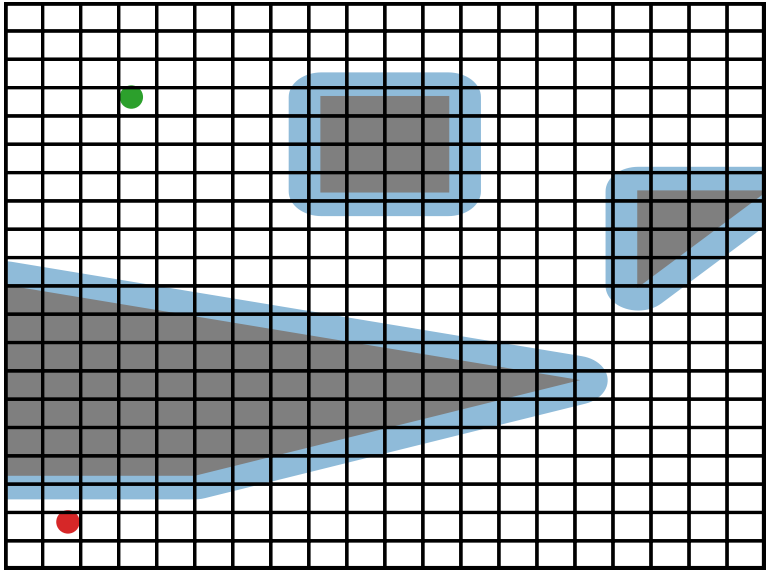




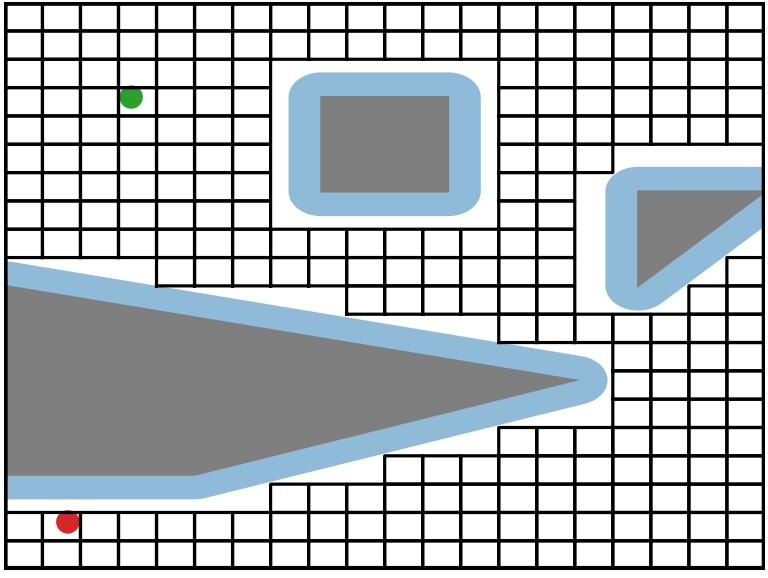
Plánovače #1: Plánování v diskrétní mřížce

- Diskretizovat konfigurační prostor (mapa je často diskretizovaná na vstupu)
- Nejjednodušší způsob diskretizace je použít mřížku
- Cílem je reprezentovat volný konfigurační prostor pomocí grafu

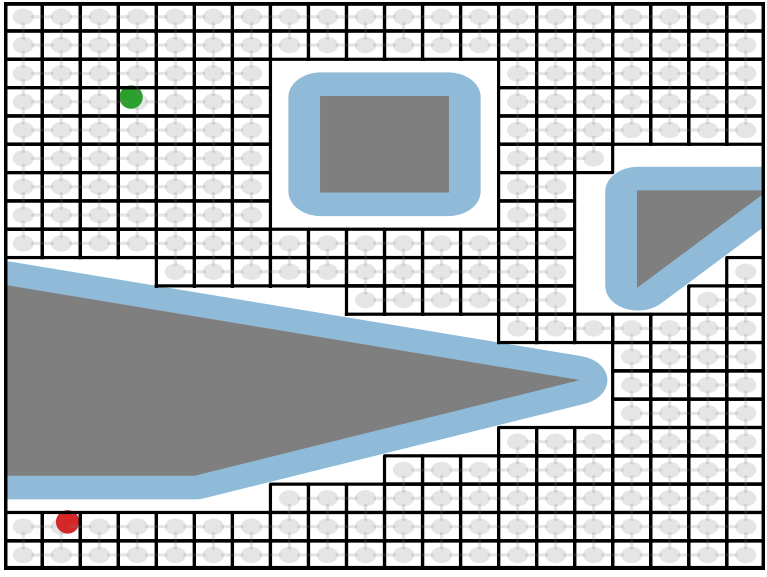
Plánovače #1: Plánování v diskrétní mřížce



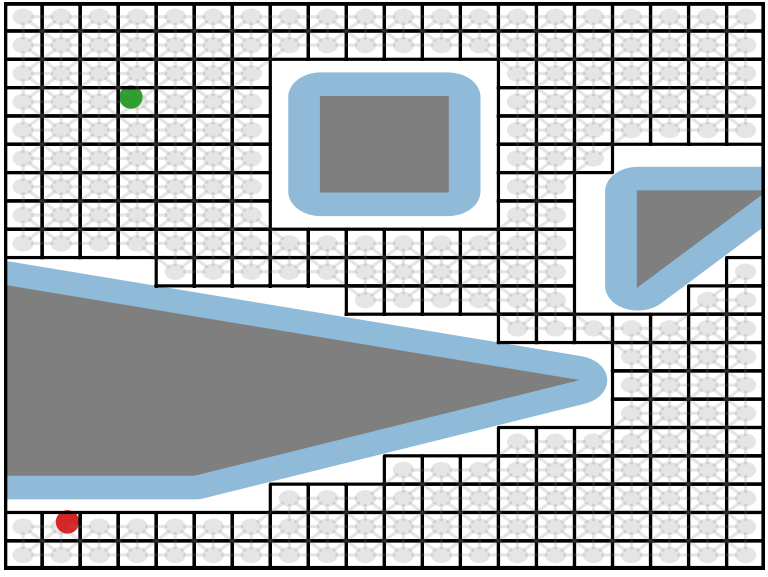
Plánovače #1: Plánování v diskrétní mřížce



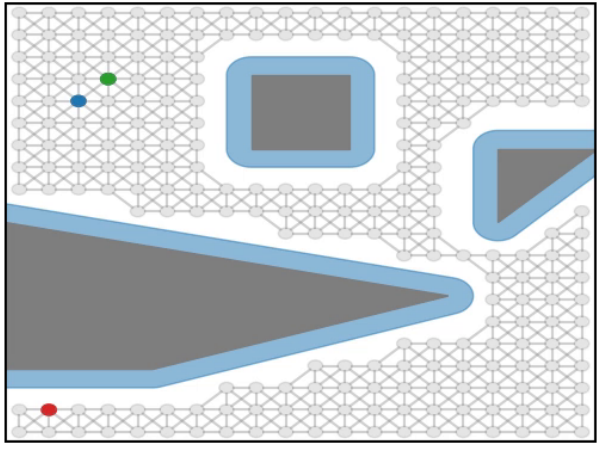
Plánovače #1: Plánování v diskrétní mřížce



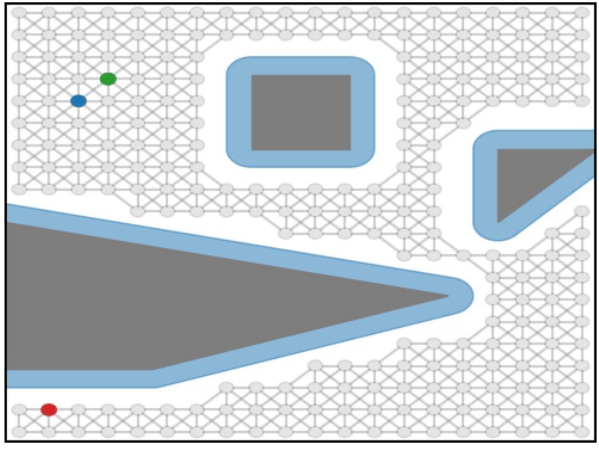
Plánovače #1: Plánování v diskrétní mřížce



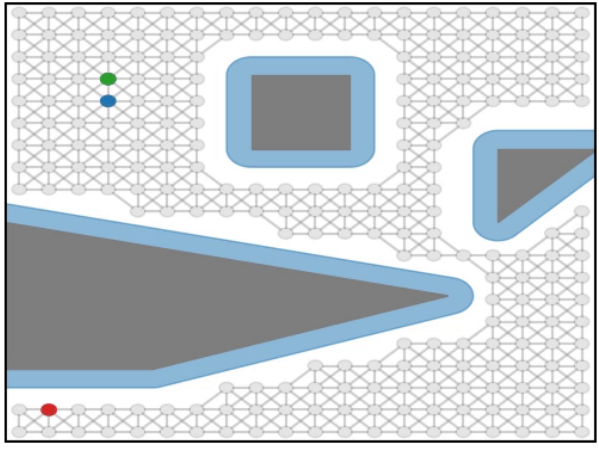
Prohledávání grafu: do šířky



Prohledávání grafu: do hloubky



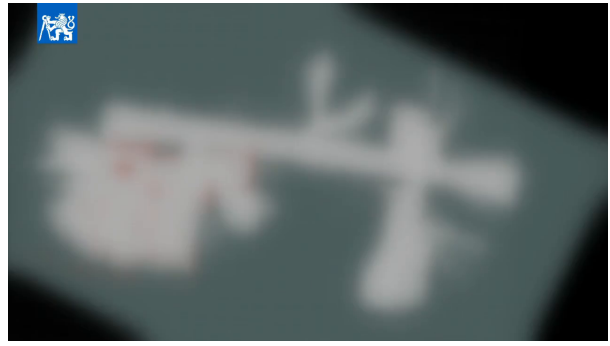
Prohledávání grafu: Informované prohledávání A*



Prohledávání grafu

- Neinformované do šířky
  - úplné - najde řešení pokud existuje
  - optimální
- Neinformované do hloubky
  - úplné jenom v konečných grafech
  - neoptimální
- Informované A*
  - používá heuristiku pro vedení prohledávání a tím zrychluje prohledávání
  - úplné, optimální

Prohledávání grafu

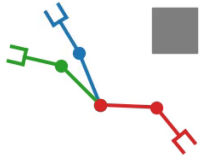


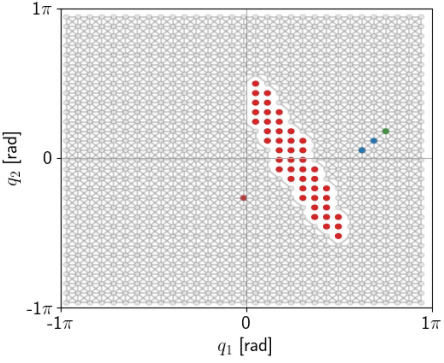
- Neinformované do šířky
  - úplné - najde řešení pokud existuje
  - optimální
- Neinformované do hloubky
  - úplné jenom v konečných grafech
  - neoptimální
- Informované A*
  - používá heuristiku pro vedení prohledávání a tím zrychluje prohledávání
  - úplné, optimální

Prohledávání grafu

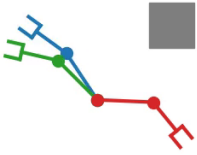
- Neinformované do šířky
  - úplné - najde řešení pokud existuje
  - optimální
- Neinformované do hloubky
  - úplné jenom v konečných grafech
  - neoptimální
- Informované A*
  - používá heuristiku pro vedení prohledávání a tím zrychluje prohledávání
  - úplné, optimální
  - můžeme použít A* pro robotický manipulátor?

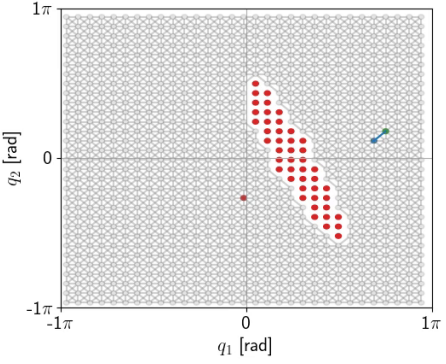
A* pro 2D manipulátor





A* pro 2D manipulátor



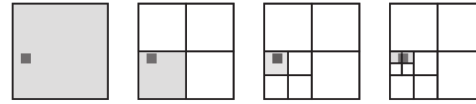


Jak velká mřížka je potřeba?

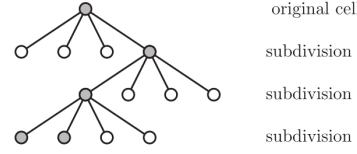
- Příklad mobilního robota $20^2 = 400$
- Příklad manipulátora $30^2 = 900$
- Komplexnější manipulátor: $30^6 = 729.000.000$
- Humanoid: $30^{20} = 3,5 \cdot 10^{29}$
- Úplnost algoritmů závisí na rozlišení mřížky - - *resolution complete*

[2] Obrázek z Lynch&Park: Modern Robotics

Jak velká mřížka je potřeba?



- Příklad mobilního robota $20^2 = 400$
- Příklad manipulátora $30^2 = 900$
- Komplexnější manipulátor: $30^6 = 729.000.000$
- Humanoid: $30^{20} = 3,5 \cdot 10^{29}$
- Úplnost algoritmů závisí na rozlišení mřížky - - *resolution complete*
- Existují mřížky s variabilním rozlišením[2]
- Diskretizovat vysoce dimenzionální prostor ale není snadné ani efektivní - používají se jiné alternativy plánování.



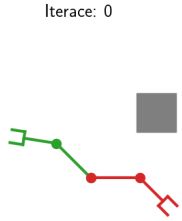
[2]Obrázek z Lynch&Park: Modern Robotics

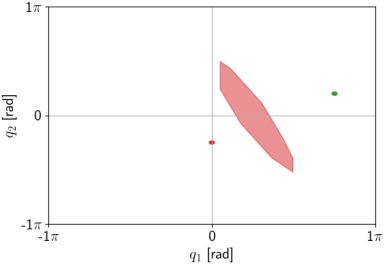
Plánovače #2: Metody založené na náhodném vzorkování Sampling Methods

- Založena na
  - náhodném vybírání vzorků z $\mathcal{C}$
  - funkci, která určí, jestli je vzorek bezkolizní
  - lokální plánovač, který nalezne cestu k náhodnému vzorku
- Vytváříme graf náhodně $\times$ vytvoření grafu pomocí diskretizace
- Méně výpočetně náročné než mřížkové metody
- Pravděpodobnostně úplné - pravděpodobnost, že najdeme řešení, se blíží 1 s počtem vzorků blížícím se nekonečnu
- Náhodné stromy (random trees - RT) a pravděpodobnostní cestovní mapy (probabilistic roadmap - PRM)

Náhodné stromy - naivní přístup

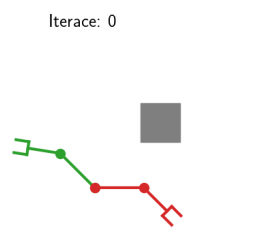
1: Vstup: start $\boldsymbol{q}_s$, cíl $\boldsymbol{q}_g$, maximální délka hrany Δq

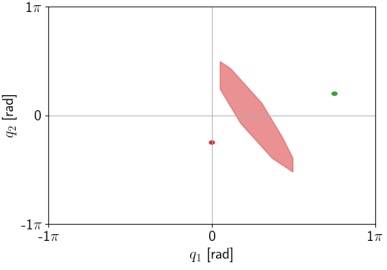




Náhodné stromy - naivní přístup

1: Vstup: start $\boldsymbol{q}_s$, cíl $\boldsymbol{q}_g$, maximální délka hrany Δq
2: $\mathcal{V} \leftarrow \{\boldsymbol{q}_s\} \quad \mathcal{E} = \{\}$

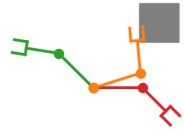


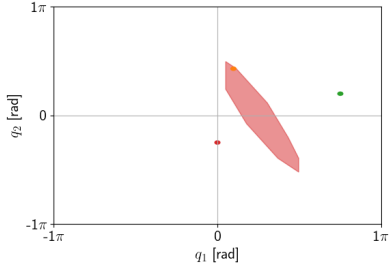


Náhodné stromy - naivní přístup

1: Vstup: start $\boldsymbol{q}_s$, cíl $\boldsymbol{q}_g$, maximální délka hrany Δq
2: $\mathcal{V} \leftarrow \{\boldsymbol{q}_s\} \quad \mathcal{E} = \{\}$
3: **for** $k = 1$ to M **do**
4: $\quad \boldsymbol{q}_{\text{rand}} \sim \mathcal{C}$

14: **end for**

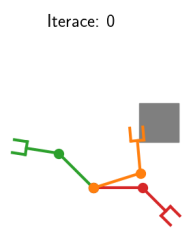


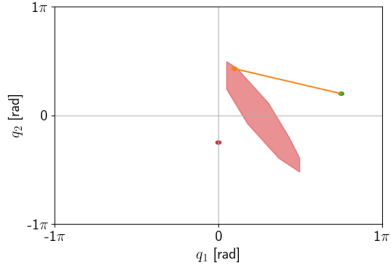


Náhodné stromy - naivní přístup

1: Vstup: start $\boldsymbol{q}_s$, cíl $\boldsymbol{q}_g$, maximální délka hrany Δq
2: $\mathcal{V} \leftarrow \{\boldsymbol{q}_s\} \quad \mathcal{E} = \{\}$
3: **for** $k = 1$ to M **do**
4: $\quad \boldsymbol{q}_{\text{rand}} \sim \mathcal{C}$
5: $\quad \boldsymbol{q}_{\text{tree}} \sim \mathcal{V}$

14: **end for**

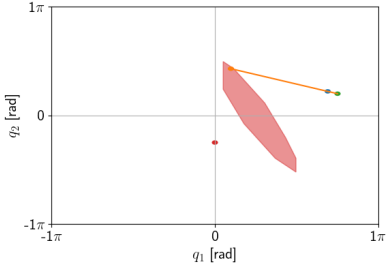




Náhodné stromy - naivní přístup

1: Vstup: start $\boldsymbol{q}_s$, cíl $\boldsymbol{q}_g$, maximální délka hrany Δq
2: $\mathcal{V} \leftarrow \{\boldsymbol{q}_s\} \quad \mathcal{E} = \{\}$
3: **for** $k = 1$ to M **do**
4: $\quad \boldsymbol{q}_{\text{rand}} \sim \mathcal{C}$
5: $\quad \boldsymbol{q}_{\text{tree}} \sim \mathcal{V}$
6: $\quad \boldsymbol{q}_{\text{new}} \leftarrow f_{\text{plan}}(\boldsymbol{q}_{\text{tree}}, \boldsymbol{q}_{\text{rand}}, \Delta q)$

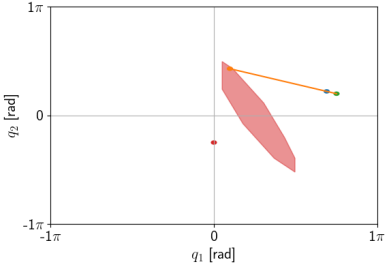
14: **end for**



Náhodné stromy - naivní přístup

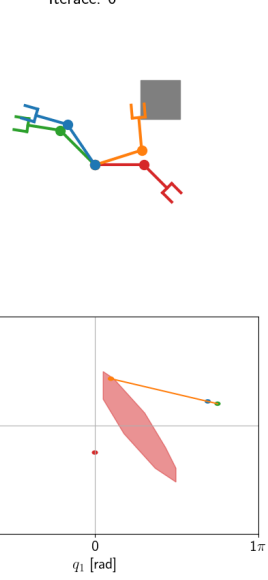
1: Vstup: start $\boldsymbol{q}_s$, cíl $\boldsymbol{q}_g$, maximální délka hrany Δq
2: $\mathcal{V} \leftarrow \{\boldsymbol{q}_s\} \quad \mathcal{E} = \{\}$
3: **for** $k = 1$ to M **do**
4: $\quad \boldsymbol{q}_{\text{rand}} \sim \mathcal{C}$
5: $\quad \boldsymbol{q}_{\text{tree}} \sim \mathcal{V}$
6: $\quad \boldsymbol{q}_{\text{new}} \leftarrow f_{\text{plan}}(\boldsymbol{q}_{\text{tree}}, \boldsymbol{q}_{\text{rand}}, \Delta q)$
7: $\quad$ **if** $\boldsymbol{q}_{\text{new}} \in \mathcal{C}_{\text{free}}$ **then**
8: $\quad\quad \mathcal{V} \leftarrow \mathcal{V} \cup \{\boldsymbol{q}_{\text{new}}\} \quad \mathcal{E} \leftarrow \mathcal{E} \cup \{(\boldsymbol{q}_{\text{tree}}, \boldsymbol{q}_{\text{new}})\}$
9: $\quad$ **end if**

14: **end for**

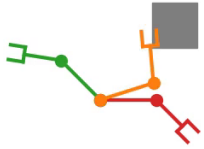


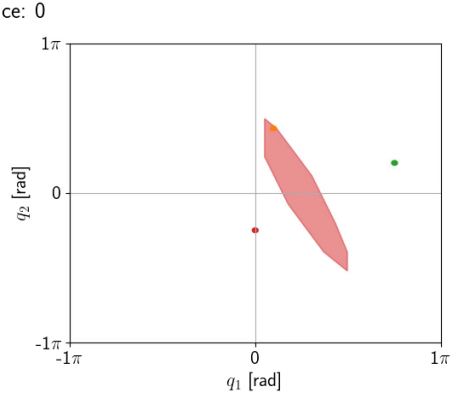
Náhodné stromy - naivní přístup

1: Vstup: start $\boldsymbol{q}_s$, cíl $\boldsymbol{q}_g$, maximální délka hrany Δq
2: $\mathcal{V} \leftarrow \{\boldsymbol{q}_s\} \quad \mathcal{E} = \{\}$
3: **for** $k = 1$ to M **do**
4: $\quad \boldsymbol{q}_{\text{rand}} \sim \mathcal{C}$
5: $\quad \boldsymbol{q}_{\text{tree}} \sim \mathcal{V}$
6: $\quad \boldsymbol{q}_{\text{new}} \leftarrow f_{\text{plan}}(\boldsymbol{q}_{\text{tree}}, \boldsymbol{q}_{\text{rand}}, \Delta q)$
7: $\quad$ **if** $\boldsymbol{q}_{\text{new}} \in \mathcal{C}_{\text{free}}$ **then**
8: $\quad\quad \mathcal{V} \leftarrow \mathcal{V} \cup \{\boldsymbol{q}_{\text{new}}\} \quad \mathcal{E} \leftarrow \mathcal{E} \cup \{(\boldsymbol{q}_{\text{tree}}, \boldsymbol{q}_{\text{new}})\})$
9: $\quad$ **end if**
10: $\quad$ **if** $\|\boldsymbol{q}_{\text{new}} - \boldsymbol{q}_g\| < \epsilon$ **then**
11: $\quad\quad \mathcal{V} \leftarrow \mathcal{V} \cup \{\boldsymbol{q}_{\text{g}}\} \quad \mathcal{E} \leftarrow \mathcal{E} \cup \{(\boldsymbol{q}_{\text{new}}, \boldsymbol{q}_{\text{g}})\})$
12: $\quad\quad$ **break**
13: $\quad$ **end if**
14: **end for**



Náhodné stromy





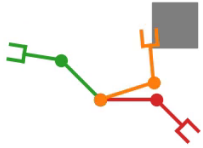
Rychle rostoucí stromy - Rapidly-exploring random tree - RRT

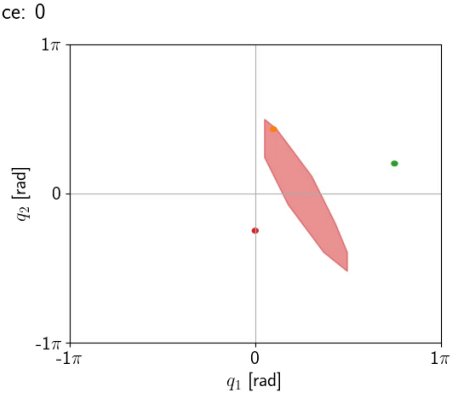
- Náhodné vybírání vrcholu stromu nepodporuje exploraci
- RRT[3] navrženy pro exploraci
- Pro výpočet $\boldsymbol{q}_{\text{new}}$ je náhodně vybraná konfigurace "spojena" s nejbližším vrcholem stromu

$$\boldsymbol{q}_{\text{tree}} = \arg\min_{\boldsymbol{q} \in \mathcal{V}} \|\boldsymbol{q} - \boldsymbol{q}_{\text{rand}}\| \quad \cancel{\boldsymbol{q}_{\text{tree}} \sim \mathcal{V}}$$

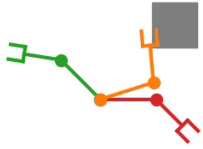
[3] LaValle et. al., Randomized Kinodynamic Planning, IJRR 2001

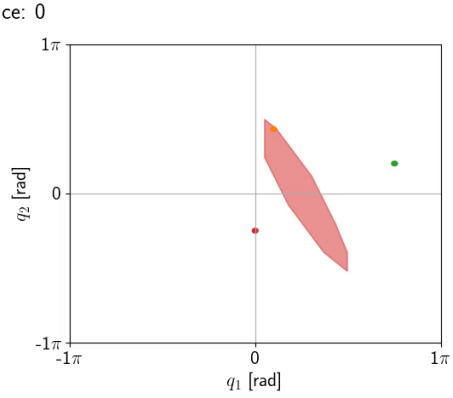
RRT





RRT - vzorkovací bias



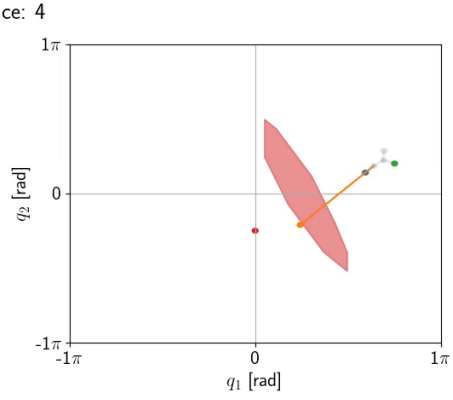


RRT

- Výsledná cesta není optimální - často nám ale stačí alespoň nějaké řešení
- RRT je efektivní i pro velké dimenze
- Vhodným vzorkováním lze zvýšit efektivita prohledávání (304×257)
- Existuje spousta modifikací RRT algoritmu
  - RRT*, BiDirectional RRT, RRG, PDRRTs, RRT*-Smart, A*-RRT, RRT*FN, RRT*-AR, RT-RRT*, RRT#, Theta*-RRT, RRT* FND, RRT-GPU, RRdT*

RRT*

- Cestu vylepšuje pomocí dalších vzorků
- Je nepatrně pomalejší než RRT (dodatečné operace pro správu stromu)
- Je potřeba definovat ukončovací kritérium
  - maximální počet vzorků
  - čas plánování

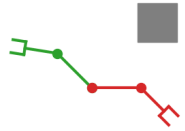

Iterace: 4
1π
0
-1π
q_2 [rad]
-1π
0
1π
q_1 [rad]

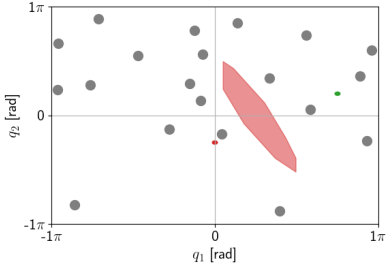
Pravděpodobnostní cestovní mapy - PRM

- Vytvoří neorientovaný graf pro pokrytí $\mathcal{C}_{\text{free}}$ před hledáním plánu
- Graf se pak prohledá pomocí A*
- PRM vs RRT:
  - Graf nemusí být strom
  - Vytvořený graf se dá použít opakovaně
  - Multi-query vs Single-query

PRM vytvoření grafu

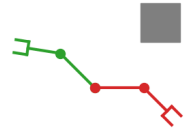
1: $\mathcal{V} \leftarrow \{\boldsymbol{q}_i | \boldsymbol{q}_i \sim \mathcal{C}_{\text{free}}\}_{i=0}^{M}$

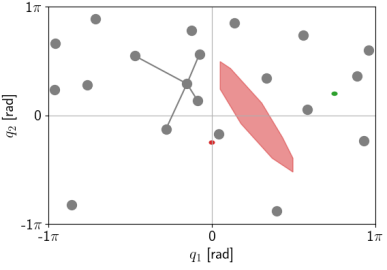




PRM vytvoření grafu

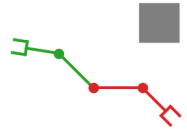
1: $\mathcal{V} \leftarrow \{\boldsymbol{q}_i | \boldsymbol{q}_i \sim \mathcal{C}_{\text{free}}\}_{i=0}^{M}$
2: $\mathcal{E} \leftarrow \{\}$
3: **for** $\boldsymbol{q}_i \in \mathcal{V}$ **do**
4: $\quad \mathcal{N}(\boldsymbol{q}_i) \leftarrow k$ nejbližších sousedů vrcholu $\boldsymbol{q}_i$
5: $\quad$ **for** $\boldsymbol{q}_j \in \mathcal{N}(\boldsymbol{q}_i)$ **do**
6: $\quad\quad$ **if** $\text{plan}(\boldsymbol{q}_i, \boldsymbol{q}_j) \in \mathcal{C}_{\text{free}} \wedge (\boldsymbol{q}_i, \boldsymbol{q}_j) \notin \mathcal{E}$ **then**
7: $\quad\quad\quad \mathcal{E} \leftarrow \mathcal{E} \cup \{(\boldsymbol{q}_i, \boldsymbol{q}_j)\}$
8: $\quad\quad$ **end if**
9: $\quad$ **end for**
10: **end for**

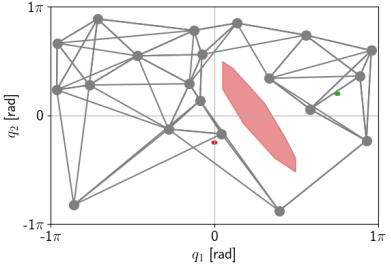




PRM vytvoření grafu

1: $\mathcal{V} \leftarrow \{\boldsymbol{q}_i | \boldsymbol{q}_i \sim \mathcal{C}_{\text{free}}\}_{i=0}^{M}$
2: $\mathcal{E} \leftarrow \{\}$
3: **for** $\boldsymbol{q}_i \in \mathcal{V}$ **do**
4: $\quad \mathcal{N}(\boldsymbol{q}_i) \leftarrow k$ nejbližších sousedů vrcholu $\boldsymbol{q}_i$
5: $\quad$ **for** $\boldsymbol{q}_j \in \mathcal{N}(\boldsymbol{q}_i)$ **do**
6: $\quad\quad$ **if** $\text{plan}(\boldsymbol{q}_i, \boldsymbol{q}_j) \in \mathcal{C}_{\text{free}} \wedge (\boldsymbol{q}_i, \boldsymbol{q}_j) \notin \mathcal{E}$ **then**
7: $\quad\quad\quad \mathcal{E} \leftarrow \mathcal{E} \cup \{(\boldsymbol{q}_i, \boldsymbol{q}_j)\}$
8: $\quad\quad$ **end if**
9: $\quad$ **end for**
10: **end for**

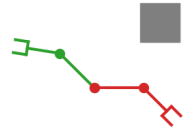


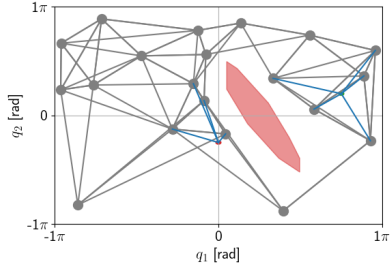


PRM vytvoření grafu

1: $\mathcal{V} \leftarrow \{\boldsymbol{q}_i | \boldsymbol{q}_i \sim \mathcal{C}_{\text{free}}\}_{i=0}^{M}$
2: $\mathcal{E} \leftarrow \{\}$
3: **for** $\boldsymbol{q}_i \in \mathcal{V}$ **do**
4: $\quad \mathcal{N}(\boldsymbol{q}_i) \leftarrow k$ nejbližších sousedů vrcholu $\boldsymbol{q}_i$
5: $\quad$ **for** $\boldsymbol{q}_j \in \mathcal{N}(\boldsymbol{q}_i)$ **do**
6: $\quad\quad$ **if** $\text{plan}(\boldsymbol{q}_i, \boldsymbol{q}_j) \in \mathcal{C}_{\text{free}} \wedge (\boldsymbol{q}_i, \boldsymbol{q}_j) \notin \mathcal{E}$ **then**
7: $\quad\quad\quad \mathcal{E} \leftarrow \mathcal{E} \cup \{(\boldsymbol{q}_i, \boldsymbol{q}_j)\}$
8: $\quad\quad$ **end if**
9: $\quad$ **end for**
10: **end for**

- Při plánování se do grafu vloží $\boldsymbol{q}_s$ a $\boldsymbol{q}_g$

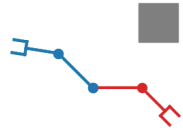


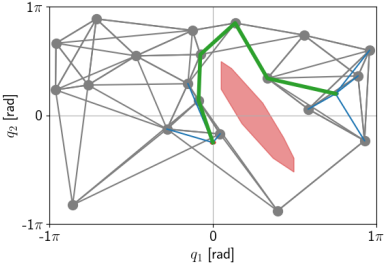


PRM vytvoření grafu

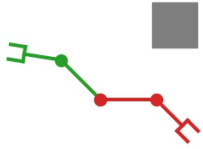
1: $\mathcal{V} \leftarrow \{\boldsymbol{q}_i | \boldsymbol{q}_i \sim \mathcal{C}_{\text{free}}\}_{i=0}^{M}$
2: $\mathcal{E} \leftarrow \{\}$
3: **for** $\boldsymbol{q}_i \in \mathcal{V}$ **do**
4: $\quad \mathcal{N}(\boldsymbol{q}_i) \leftarrow k$ nejbližších sousedů vrcholu $\boldsymbol{q}_i$
5: $\quad$ **for** $\boldsymbol{q}_j \in \mathcal{N}(\boldsymbol{q}_i)$ **do**
6: $\quad\quad$ **if** $\text{plan}(\boldsymbol{q}_i, \boldsymbol{q}_j) \in \mathcal{C}_{\text{free}} \wedge (\boldsymbol{q}_i, \boldsymbol{q}_j) \notin \mathcal{E}$ **then**
7: $\quad\quad\quad \mathcal{E} \leftarrow \mathcal{E} \cup \{(\boldsymbol{q}_i, \boldsymbol{q}_j)\}$
8: $\quad\quad$ **end if**
9: $\quad$ **end for**
10: **end for**

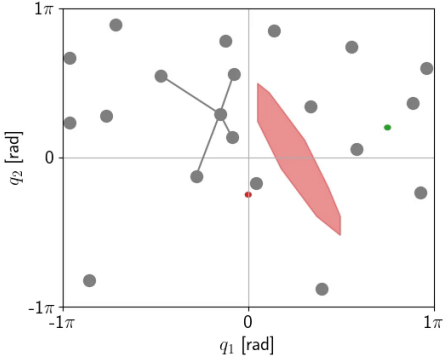
- Při plánování se do grafu vloží $\boldsymbol{q}_s$ a $\boldsymbol{q}_g$
- Pomocí prohledávání grafu se nalezne cesta





PRM





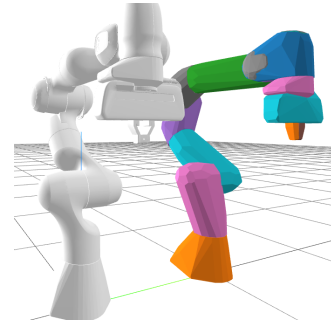
Jak by plánování vypadalo pro reálný manipulátor?

- Franka Emika Panda Robot
- Výpočet kolize je komplikovanější ve 3D
  - Vstup: $\boldsymbol{q}$ a kinematický model včetně 3D modelů jednotlivých ramen
  - Výstup: Je konfigurace v kolizi?
- Jinak lze RRT/PRM aplikovat beze změny
- Prohledávaný prostor je vícedimenzionální (7 DoF + 1 DoF)

Výpočet kolize

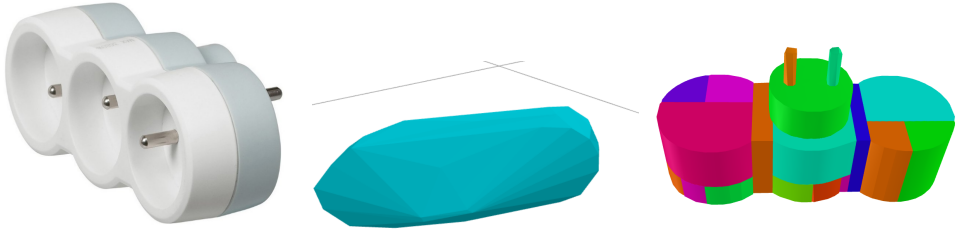
- Hledáme polohu a orientaci každého ramena
  - Výpočet přímé kinematické úlohy
  - Počítáme kolizi každého ramena zvlášť

Výpočet kolize

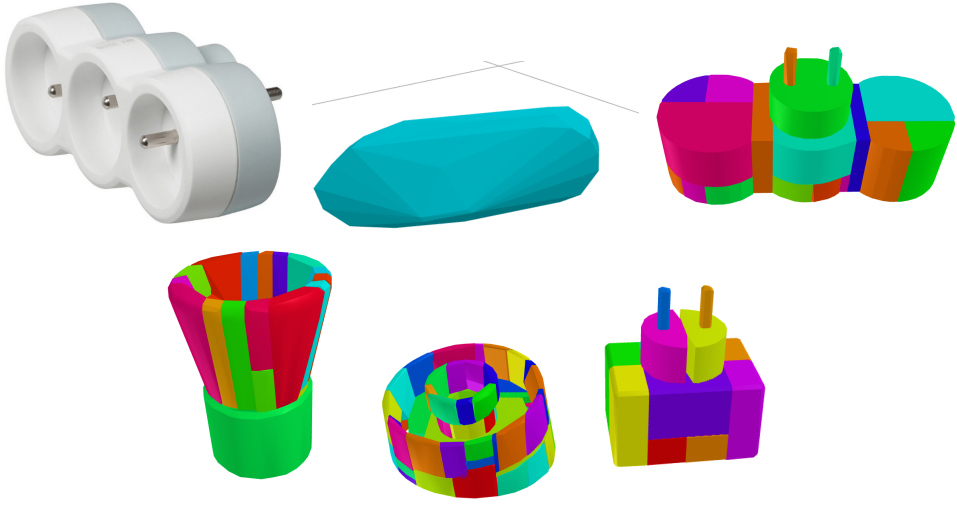


- Hledáme polohu a orientaci každého ramena
  - Výpočet přímé kinematické úlohy
  - Počítáme kolizi každého ramena zvlášť
- 3D modely bývají velmi detailní (a nekonvexní)
  - Pro urychlení výpočtu se spočítá konvexní aproximace
  - Konvexní obálka nebo konvexní dekompozice
  - Konvexní dekompozice se spočítá i pro objekty scény
- Bezkoliznost dvou konvexních objektů lze ověřit např. pomocí algoritmu GJK

Ukázka konvexní dekompozice objektů



Ukázka konvexní dekompozice objektů



Kolize mezi rameny

- Jsou ramena na obrázku v kolizi?

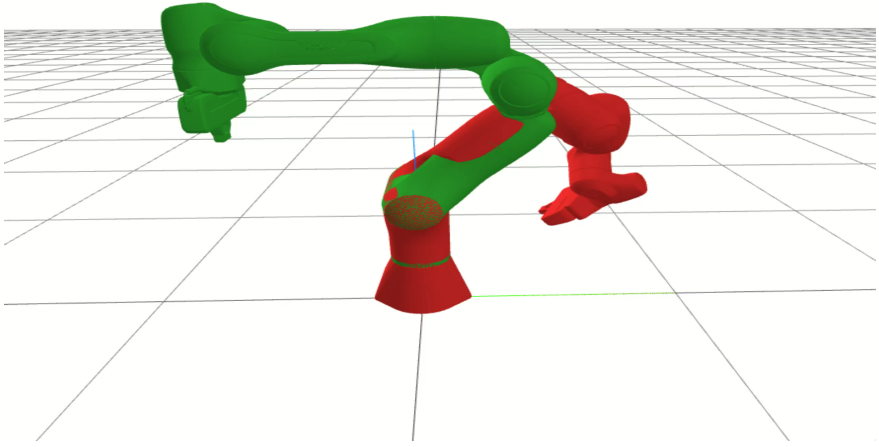
Kolize mezi rameny

- Jsou ramena na obrázku v kolizi?
- Výpočet kolize je časově náročný (zvlášť pro objekty, které jsou téměř v kontaktu)
- Snažíme se počet kolizních párů zredukovat
- V praxi vypínáme kolize mezi objekty, které:
  - jsou pořád v kolizi, např. sousední ramena
  - nemohou být nikdy v kolizi, např. oddělené statické kolizní objekty
  - ROS: MoveIt! - detekuje tyto kolizní páry při vytváření konfiguračních souborů

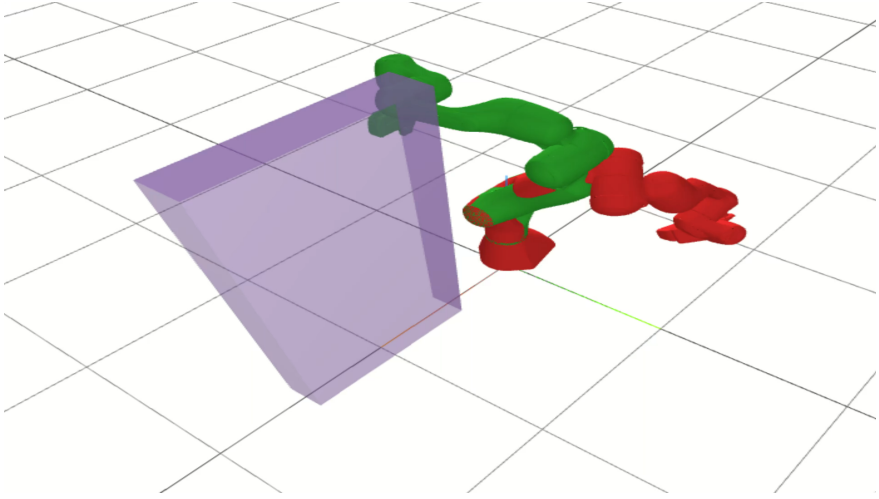
Detekce kolizí pro robota - závěr

- Přípravná fáze
  - Spočítat konvexní obaly
  - Definovat kolizní páry
- Výpočetní fáze
  - Pomocí přímé kinematiky spočítáme polohu a orientaci každého ramena
  - Pro každý kolizní pár spočítáme bezkoliznost
  - Pokud je kterýkoliv pár v kolizi, konfigurace je nepřípustná

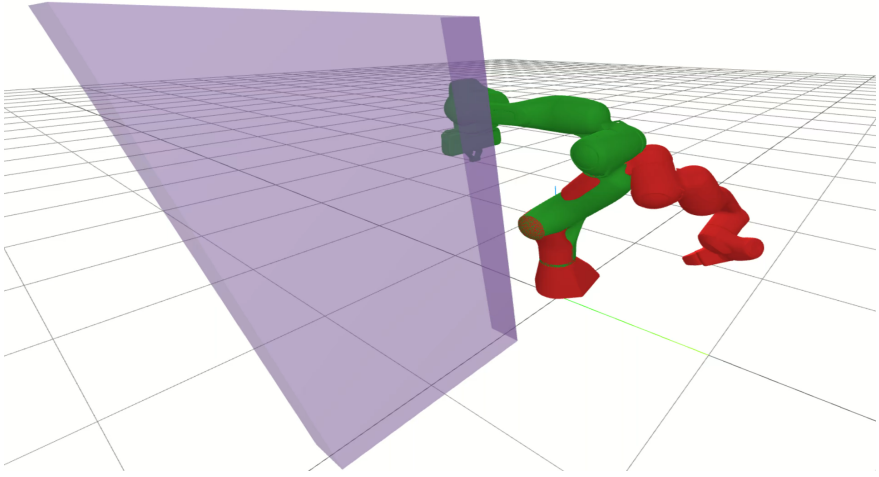
Naplánovaný pohyb



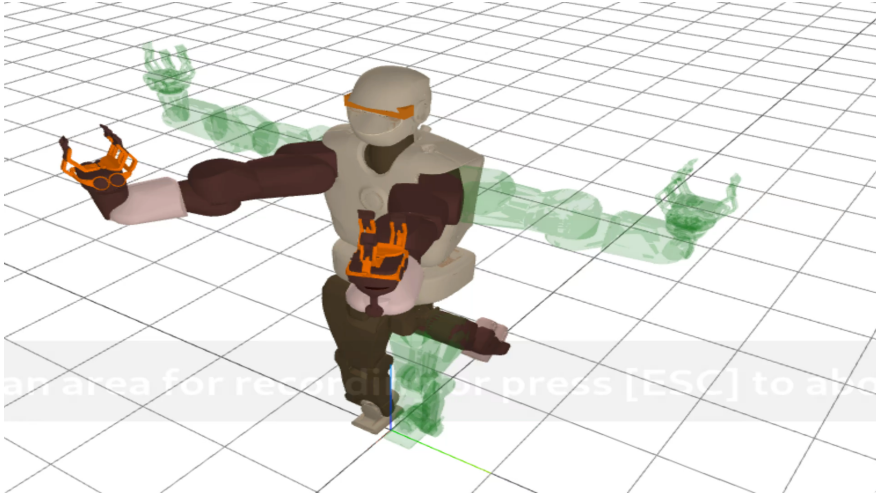
Naplánovaný pohyb RRT - 3132 iterací



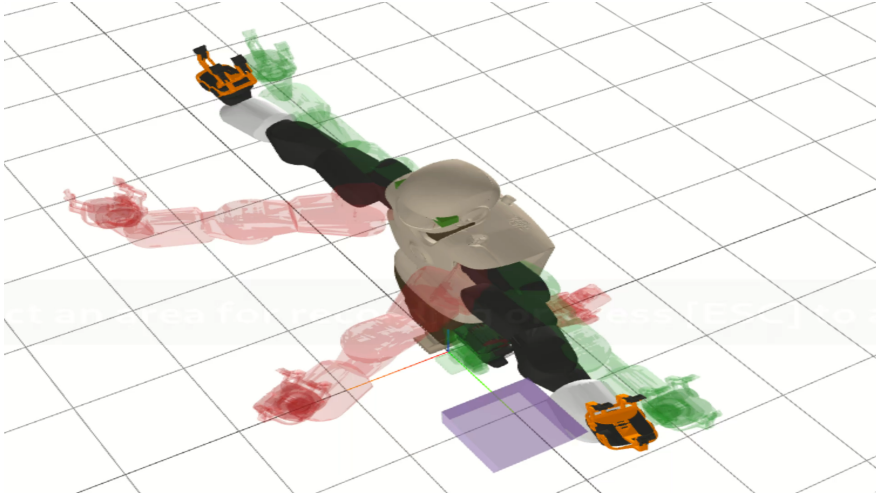
Naplánovaný pohyb RRT* - 10000 iterací



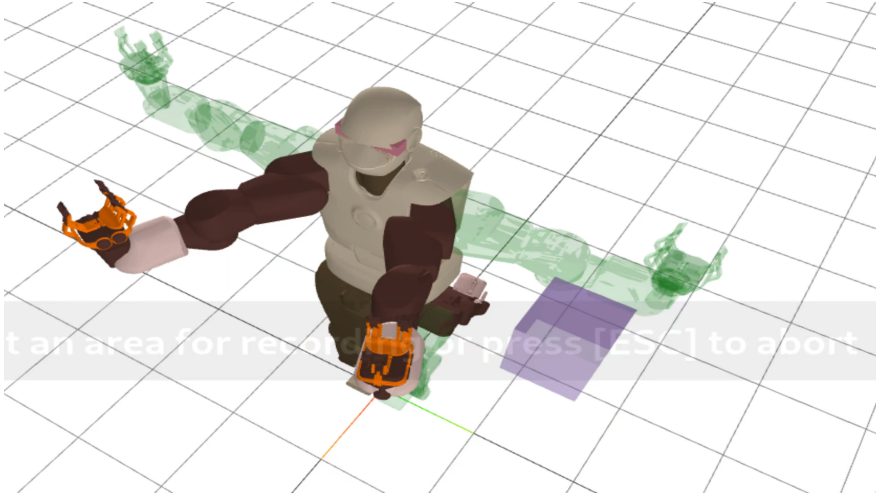
Naplánovaný pohyb Talos



Naplánovaný pohyb



Naplánovaný pohyb



Zkracování cesty: metoda náhodného zkracování

- Random shortcut
- Zkracování cesty pomocí náhodných vzorků a výpočtu nejkratší cesty
- Algortimus:
  - Vyber náhodný bod $\boldsymbol{q}_1$ na cestě
  - Vyber náhodný bod $\boldsymbol{q}_2$ na cestě
  - Pokud je interpolace mezi $\boldsymbol{q}_1$ a $\boldsymbol{q}_2$ bezkolizní, nahraď cestu mezi $\boldsymbol{q}_1$ a $\boldsymbol{q}_2$ přímkou v konfiguračním prostoru

Shrnutí

- Co je to konfigurační prostor
- Rozdíl mezi cestou a trajektorií
- Co je to plánování cesty
- Rozdíl mezi plánovacími metodami:
  - Diskretizovaná mřížka
  - RRT
  - RRT*
  - PRM
- Co chybělo:
  - Plánovačů existuje spousta
  - Optimalizační metody plánování

Cvičení

- Implementace PRM
- Domácí úkol: implementace RRT a Random shortcut