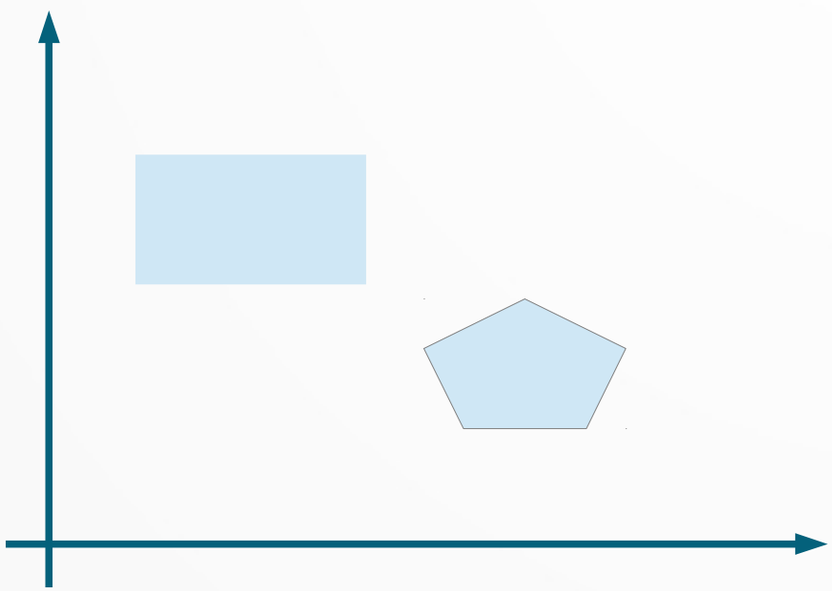
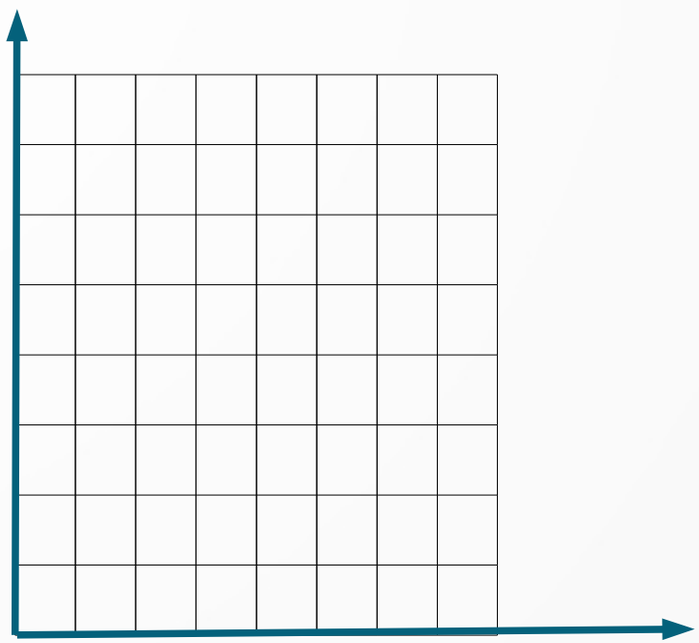
Plánování cesty 2

Jiří Pech

Reprezentace prostoru



Polygonální



Mřížkové

Převod mapy objektů na graf

- Graf – množina vrcholů a hran
- Zvětšit překážky o poloměr robota
- Nyní propojit start, cíl a všechny vrcholy, kde je cesta legální
- Odstraníme překážky a hledáme nejkratší cestu po zbylém grafu

Dijkstrův algoritmus

- Pro jakýkoliv uzel n nalezne nejkratší cestu ze startu – g(n)
- Vstup:
  - Graf
  - Vzdálenosti na grafu
  - Start
  - Cíl
- Výstup
  - Cesta ze startu do cíle

Dijksrův algoritmus

- Nastav všechny uzly jako nenavštívené
- Nastav pro start g(S)=0
- Dokud existují nenavštívené uzly:
  - Vezmi nenavštívený uzel s nejmenší g(n)
  - Nastav n jako navštívený
  - Pro všechny r sousedy n
    - g(r)=min[g(r), g(n)+d(n,m)]

Dijkstrův algoritmus

- Pokud si pamatujeme u každého uzlu, ze kterého uzlu jsme přišli nejkratší cestou je na konci jednoduché zobrazení cesty

Plánování na mřížce

- Převod mřížky na graf
  - Všechny volné mřížky bereme jako vrchol
  - Ke všem volným sousedním buňkám uděláme hranu
    - Sousedství „vedle“ - délka hrany 1
    - Sousedství „přes vrchol“ – délka hrany 1,4

A* algoritmus

- Pro všechny uzly n nastavíme funkci h(n) – jak je uzel vzdálen od cíle. Při nastavování ignorujeme překážky.
- Vstup: graf, start, cíl, hodnoty h(n) v bodech
- Výstup: cesta od startu do cíle

A* algoritmus

- Označ všechny uzly jako nenavštívené
- Osnač Start jako navštívený, nastav funkci g(S)=0, dále funkci f(S)=h(S)+g(S)
- Dokud existují nenavštívené uzly:
  - Vezmi uzel n s nejnižší f(n)
  - Označ jej za navštívený
  - Pro všechny jeho sousedy r:
    - g(r)=min(g(r), g(n)+d(n,r)
    - f(r)=g(r)+h(r)

Plánování pohybu robotů v praxi

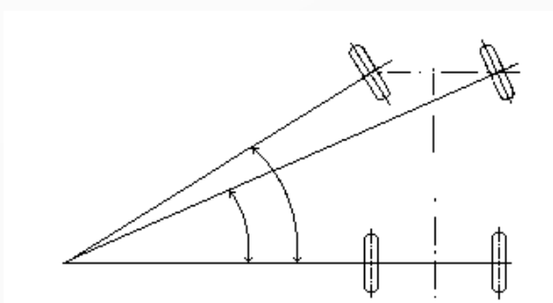
- Rozdělení:
  - Holonomní – mohou se vydat kdykoliv kterýmkoliv směrem (dron)
  - Neholonomní
    - Tank – umí se otočit na místě
    - Auto – je třeba počítat s trajektorií pro zatočení

Diferenciální robot (tank)

- Pouze pro vnitřní použití
- Pro
  - Pouze dvě ovládaná kola
  - Není třeba speciální řídící algoritmus
- Proti
  - Nemůže jet přímo do strany, musí se otočit
  - Je nutné pasivní kolečko s možností vlivu na pohyb

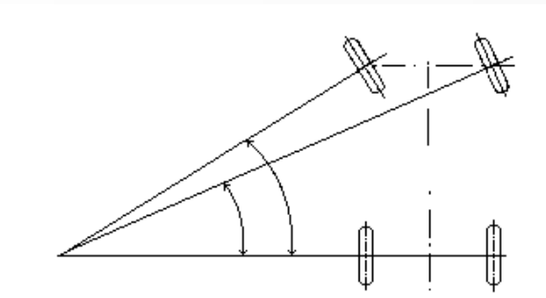
Neholonomický robot

- Závisí na orientaci (předek, zadek)
- Pohyb je možné složit z několika základních (vpřed, vzad, vlevo, vpravo)
- C-prostor je nutné rozšířit o orientaci (x,y,Θ)
- Ackermannovo řízení - pro venkovní roboty



Ackermanovo řízení

- Pro venkovní roboty
- Pro:
  - Dvě ovládaná kola
  - Jeden řídící vstup
  - Žádný smyk
- Proti:
  - Nelze točit pod velkým úhlem
  - Nelze se otočit na místě
  - Ne-holonomické



Konec

Děkuji za pozornost