Navigation

ST5 Autonomous robotics

Francis Colas

2022-10-07

Introduction

Path planning

- configuration space and planning algorithms
- known and static map

Introduction

Path planning

- configuration space and planning algorithms
- known and static map

Navigation

- mobile robot motion
  - path planning
  - path execution
  - obstacle avoidance
- exploration
  - unknown environment
  - decide commands to build map

Introduction

Path planning

- configuration space and planning algorithms
- known and static map

Navigation

- mobile robot motion
- exploration

Aim of the session

- trajectory following
- obstacle avoidance
- exploration

01

Trajectory following

Trajectory following

Trajectory following

- decide commands to execute planned trajectory
- using sensor values
- taking into account the robot constraints

Trajectory following

Trajectory following

- decide commands to execute planned trajectory
- using sensor values
- taking into account the robot constraints

General principle

- given a trajectory
- given the position/error
- given the robot motion model
- compute a command to follow the trajectory

Trajectory following

Trajectory following

- decide commands to execute planned trajectory
- using sensor values
- taking into account the robot constraints

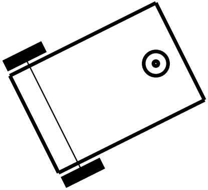
General principle

- given a trajectory
- given the position/error
- given the robot motion model
- compute a command to follow the trajectory

Kinematic models

Differential-drive robot

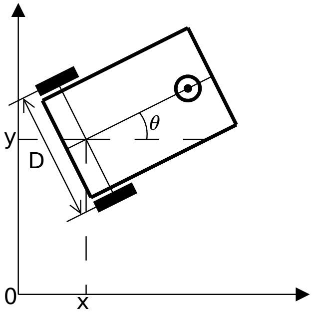
- left and right independent motor wheels
- caster wheel for stabilization



Kinematic models

Differential-drive robot

- left and right independent motor wheels
- caster wheel for stabilization
- configuration: 2D pose (x, y, θ)
- command: wheel velocities (v_l, v_r)

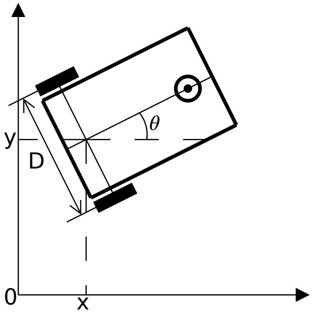


Kinematic models

Differential-drive robot

- left and right independent motor wheels
- caster wheel for stabilization
- configuration: 2D pose (x, y, θ)
- command: wheel velocities (v_l, v_r)
- kinematic model

$$\begin{cases} \dot{x} &= \frac{v_r+v_l}{2} \cos\theta \\ \dot{y} &= \frac{v_r+v_l}{2} \sin\theta \\ \dot{\theta} &= \frac{v_r-v_l}{D} \end{cases}$$

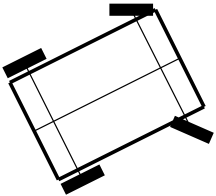


Inria informatics mathematics

Kinematic models

Car-like vehicles

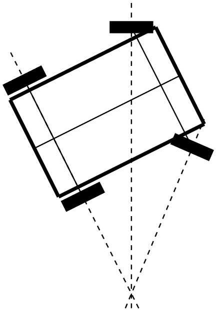
- front wheels can pivot
- rear wheels are fixed



Kinematic models

Car-like vehicles

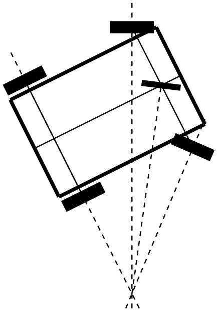
- front wheels can pivot
- rear wheels are fixed



Kinematic models

Car-like vehicles

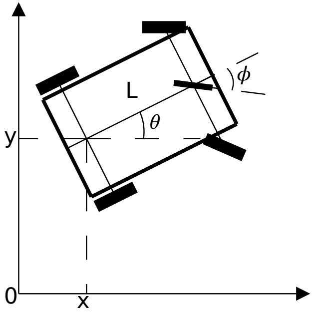
- front wheels can pivot
- rear wheels are fixed



Kinematic models

Car-like vehicles

- front wheels can pivot
- rear wheels are fixed
- configuration: 2D pose and steering angle (x, y, θ, ϕ)
- command: wheel speed and change in steering angle (v, u)

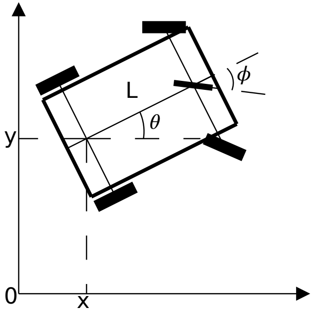


Kinematic models

Car-like vehicles

- front wheels can pivot
- rear wheels are fixed
- configuration: 2D pose and steering angle (x, y, θ, ϕ)
- command: wheel speed and change in steering angle (v, u)
- kinematic model

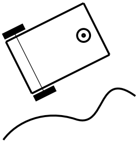
$$\begin{cases} \dot{x} &= v \cos \theta \\ \dot{y} &= v \sin \theta \\ \dot{\theta} &= \frac{v}{L} \tan \phi \\ \dot{\phi} &= u \end{cases}$$



Trajectory following

Principle

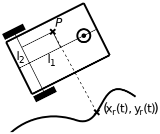
- define commands as a function of error
- differential equation of error



Trajectory following

Principle

- define commands as a function of error
- differential equation of error



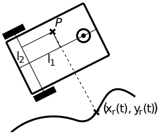
Differential-drive robot

- point motion

$$\begin{pmatrix} \dot{x}_P \\ \dot{y}_P \end{pmatrix} = \begin{pmatrix} \frac{v_r+v_l}{2} \cos\theta - \frac{v_r-v_l}{D}(l_1 \sin\theta + l_2 \cos\theta) \\ \frac{v_r+v_l}{2} \sin\theta - \frac{v_r-v_l}{D}(-l_1 \cos\theta + l_2 \sin\theta) \end{pmatrix} = \boldsymbol{M} \begin{pmatrix} v_r \\ v_l \end{pmatrix}$$

informatics mathematics Inria

Trajectory following



Principle

- define commands as a function of error
- differential equation of error

Differential-drive robot

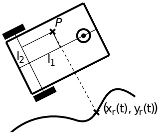
- point motion

$$\dot{\boldsymbol{x}}_P = \boldsymbol{M}\boldsymbol{u}$$

Trajectory following

Principle

- define commands as a function of error
- differential equation of error



Differential-drive robot

- point motion

$$\dot{\boldsymbol{x}}_P = \boldsymbol{M}\boldsymbol{u}$$

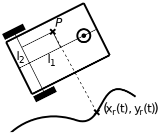
- error with respect to $\boldsymbol{e} = (x_P - x_r(t), y_P - y_r(t))$:

$$\dot{\boldsymbol{e}} = \dot{\boldsymbol{x}}_P - \dot{\boldsymbol{x}}_r = \boldsymbol{M}\boldsymbol{u} - \dot{\boldsymbol{x}}_r$$

Trajectory following

Principle

- define commands as a function of error
- differential equation of error



Differential-drive robot

- point motion

$$\dot{\boldsymbol{x}}_P = \boldsymbol{M}\boldsymbol{u}$$

- error with respect to $\boldsymbol{e} = (x_P - x_r(t), y_P - y_r(t))$:

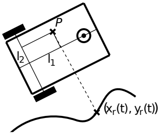
$$\dot{\boldsymbol{e}} = \dot{\boldsymbol{x}}_P - \dot{\boldsymbol{x}}_r = \boldsymbol{M}\boldsymbol{u} - \dot{\boldsymbol{x}}_r$$

- error reduction $\dot{\boldsymbol{e}} = -\boldsymbol{K}\boldsymbol{e}$

Trajectory following

Principle

- define commands as a function of error
- differential equation of error



Differential-drive robot

- point motion

$$\dot{\boldsymbol{x}}_P = \boldsymbol{M}\boldsymbol{u}$$

- error with respect to $\boldsymbol{e} = (x_P - x_r(t), y_P - y_r(t))$:

$$\dot{\boldsymbol{e}} = \dot{\boldsymbol{x}}_P - \dot{\boldsymbol{x}}_r = \boldsymbol{M}\boldsymbol{u} - \dot{\boldsymbol{x}}_r$$

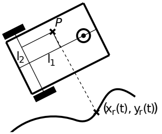
- error reduction $\dot{\boldsymbol{e}} = -\boldsymbol{K}\boldsymbol{e}$
- proportional correction with feed-forward

$$\boldsymbol{M}\boldsymbol{u} = \dot{\boldsymbol{x}}_r - \boldsymbol{K}\boldsymbol{e}$$

Trajectory following

Principle

- define commands as a function of error
- differential equation of error



Differential-drive robot

- point motion

$$\dot{\boldsymbol{x}}_P = \boldsymbol{M}\boldsymbol{u}$$

- error with respect to $\boldsymbol{e} = (x_P - x_r(t), y_P - y_r(t))$:

$$\dot{\boldsymbol{e}} = \dot{\boldsymbol{x}}_P - \dot{\boldsymbol{x}}_r = \boldsymbol{M}\boldsymbol{u} - \dot{\boldsymbol{x}}_r$$

- error reduction $\dot{\boldsymbol{e}} = -\boldsymbol{K}\boldsymbol{e}$
- proportional correction with feed-forward

$$\boldsymbol{u} = \boldsymbol{M}^{-1}\dot{\boldsymbol{x}}_r - \boldsymbol{M}^{-1}\boldsymbol{K}\boldsymbol{e}$$

informatics mathematics

Conclusion on trajectory following

Trajectory following

- automation
- several methods

Conclusion on trajectory following

Trajectory following

- automation
- several methods

Proportional with feed-forward

- simple error reduction
- based on the kinematic model
- can be generalized to cars
- limits:
  - $l_1 \neq 0$
  - no control of orientation

Conclusion on trajectory following

Trajectory following

- automation
- several methods

Proportional with feed-forward

- simple error reduction
- based on the kinematic model
- can be generalized to cars
- limits:
  - $l_1 \neq 0$
  - no control of orientation

Path following

- reference position
- projection in Frenet frame

02

Obstacle avoidance

Obstacle avoidance

Obstacle avoidance

- need exteroceptive sensors
- computation of new commands:
  - avoiding obstacles
  - while reaching target

Approaches

- potential fields
- vector field histogram
- dynamic window approach
- velocity obstacles

Vector field histogram

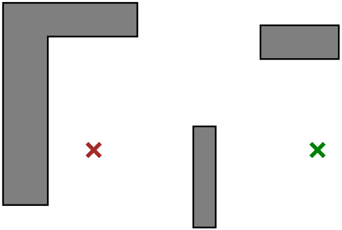
Vector field histogram

- histogram of density of obstacles
- according to direction
- identification of density valleys
- choice of the deepest valley

Vector field histogram

Vector field histogram

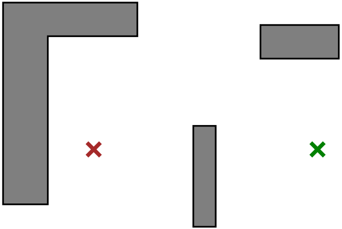
- histogram of density of obstacles
- according to direction
- identification of density valleys
- choice of the deepest valley

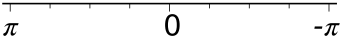


Vector field histogram

Vector field histogram

- histogram of density of obstacles
- according to direction
- identification of density valleys
- choice of the deepest valley

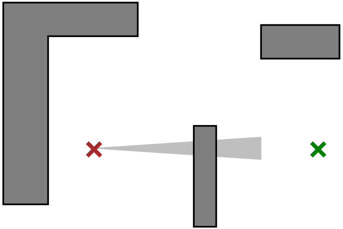


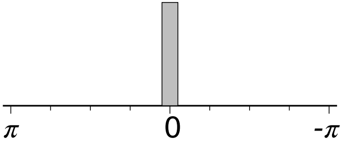


Vector field histogram

Vector field histogram

- histogram of density of obstacles
- according to direction
- identification of density valleys
- choice of the deepest valley

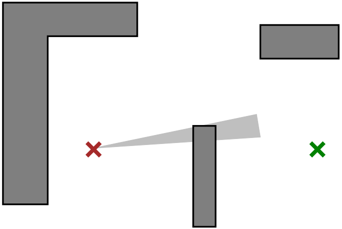


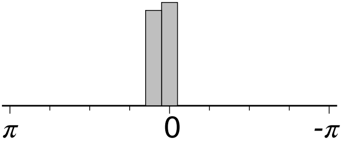


Vector field histogram

Vector field histogram

- histogram of density of obstacles
- according to direction
- identification of density valleys
- choice of the deepest valley

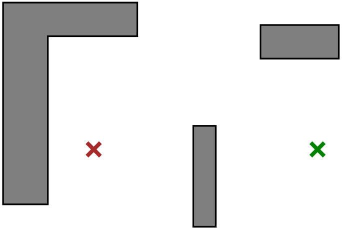


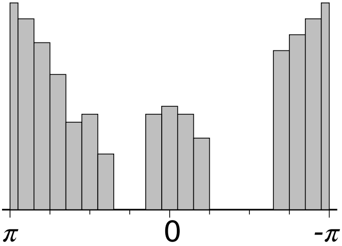


Vector field histogram

Vector field histogram

- histogram of density of obstacles
- according to direction
- identification of density valleys
- choice of the deepest valley

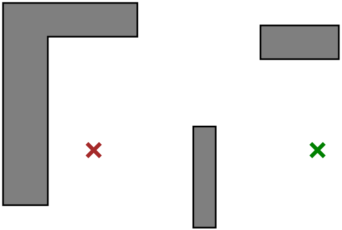


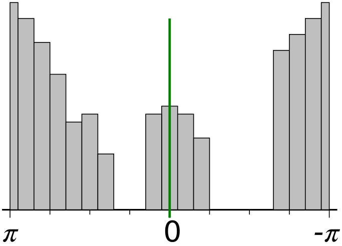


Vector field histogram

Vector field histogram

- histogram of density of obstacles
- according to direction
- identification of density valleys
- choice of the deepest valley

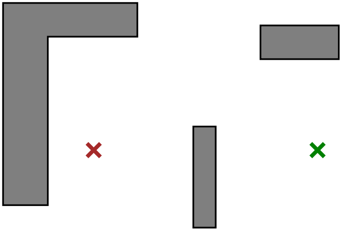


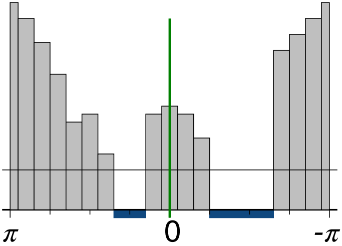


Vector field histogram

Vector field histogram

- histogram of density of obstacles
- according to direction
- identification of density valleys
- choice of the deepest valley

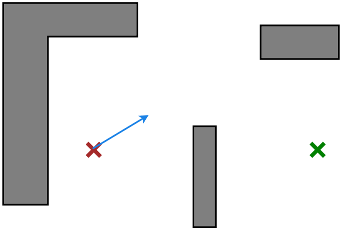


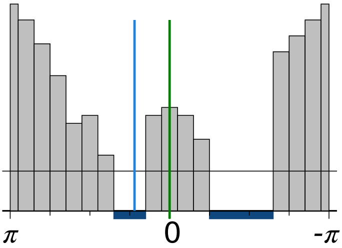


Vector field histogram

Vector field histogram

- histogram of density of obstacles
- according to direction
- identification of density valleys
- choice of the deepest valley





Dynamic window approach

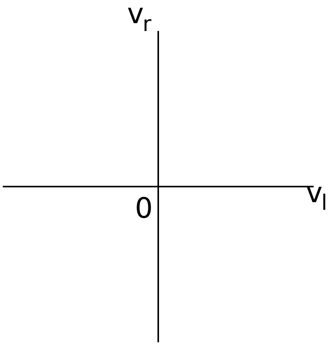
Dynamic window approach

- command space
- check commands leading to collision
- check feasible commands based on dynamics
- check difference with desire
- weighting

Dynamic window approach

Dynamic window approach

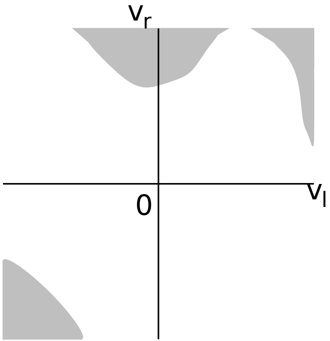
- command space
- check commands leading to collision
- check feasible commands based on dynamics
- check difference with desire
- weighting



Dynamic window approach

Dynamic window approach

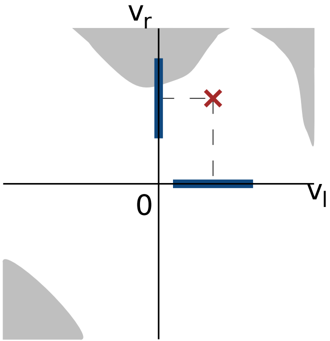
- command space
- check commands leading to collision
- check feasible commands based on dynamics
- check difference with desire
- weighting



Dynamic window approach

Dynamic window approach

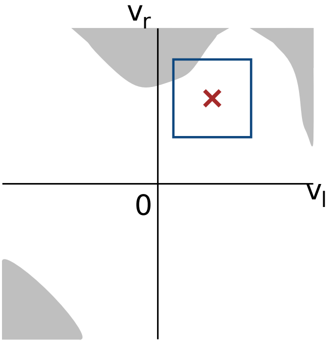
- command space
- check commands leading to collision
- check feasible commands based on dynamics
- check difference with desire
- weighting



Dynamic window approach

Dynamic window approach

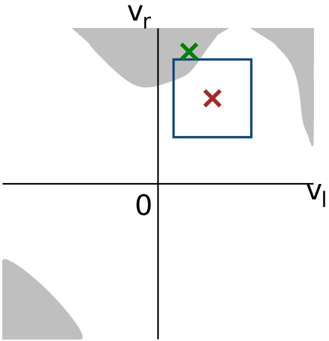
- command space
- check commands leading to collision
- check feasible commands based on dynamics
- check difference with desire
- weighting



Dynamic window approach

Dynamic window approach

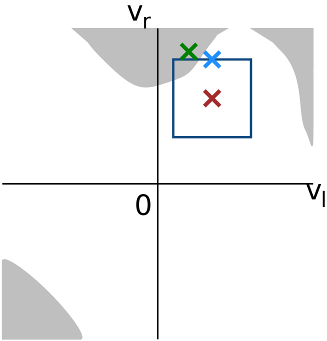
- command space
- check commands leading to collision
- check feasible commands based on dynamics
- check difference with desire
- weighting



Dynamic window approach

Dynamic window approach

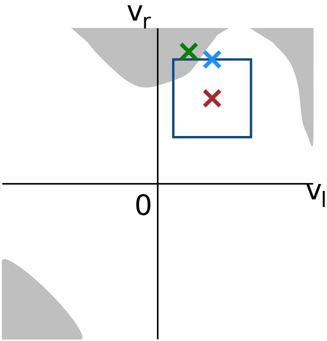
- command space
- check commands leading to collision
- check feasible commands based on dynamics
- check difference with desire
- weighting



Dynamic window approach

Dynamic window approach

- command space
- check commands leading to collision
- check feasible commands based on dynamics
- check difference with desire
- weighting
- check commands nearer to path



Velocity obstacles

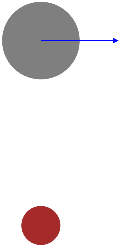
Velocity obstacles

- avoid dynamic obstacles
- assumption of known velocities
- planning in velocity space
- check velocities leading to collision

Velocity obstacles

Velocity obstacles

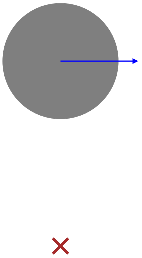
- avoid dynamic obstacles
- assumption of known velocities
- planning in velocity space
- check velocities leading to collision



Velocity obstacles

Velocity obstacles

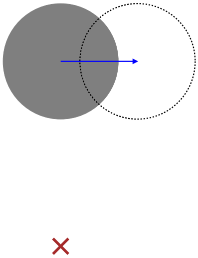
- avoid dynamic obstacles
- assumption of known velocities
- planning in velocity space
- check velocities leading to collision



Velocity obstacles

Velocity obstacles

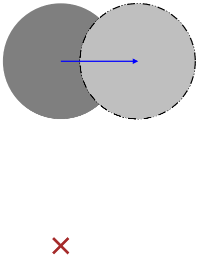
- avoid dynamic obstacles
- assumption of known velocities
- planning in velocity space
- check velocities leading to collision



Velocity obstacles

Velocity obstacles

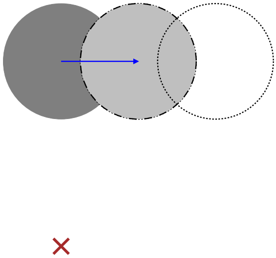
- avoid dynamic obstacles
- assumption of known velocities
- planning in velocity space
- check velocities leading to collision



Velocity obstacles

Velocity obstacles

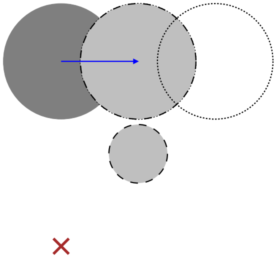
- avoid dynamic obstacles
- assumption of known velocities
- planning in velocity space
- check velocities leading to collision



Velocity obstacles

Velocity obstacles

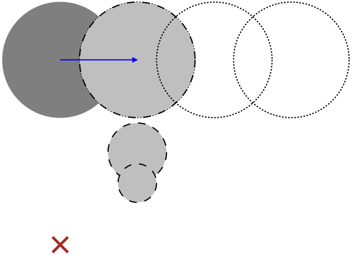
- avoid dynamic obstacles
- assumption of known velocities
- planning in velocity space
- check velocities leading to collision



Velocity obstacles

Velocity obstacles

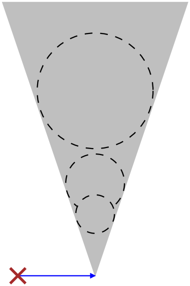
- avoid dynamic obstacles
- assumption of known velocities
- planning in velocity space
- check velocities leading to collision



Velocity obstacles

Velocity obstacles

- avoid dynamic obstacles
- assumption of known velocities
- planning in velocity space
- check velocities leading to collision



Conclusion on obstacle avoidance

Obstacle avoidance

- local modification of commands
- recognize acceptable commands
- fast reactions
- sometimes also trajectory following

Conclusion on obstacle avoidance

Obstacle avoidance

- local modification of commands
- recognize acceptable commands
- fast reactions
- sometimes also trajectory following

Limitations

- obstacle detection
- velocity estimation
- no general guarantee

03

Exploration

Autonomous motion decision

Exploration

- choose the actions of a mobile robot
- to discover an environment
- while building the map
- $\rightarrow$ information optimization

Autonomous motion decision

Exploration

- choose the actions of a mobile robot
- to discover an environment
- while building the map
- $\rightarrow$ information optimization

Active localization

- unknown localization
- motions to better localize
- known map

Autonomous motion decision

Exploration

- choose the actions of a mobile robot
- to discover an environment
- while building the map
- $\rightarrow$ information optimization

Active localization

- unknown localization
- motions to better localize
- known map

Pursuit evasion problem

- find and follow another mobile object
- known or unknown environment

Information optimization

Information quantity

- use of entropy

$$H_p = \begin{cases} -\int p(x)\log p(x)\,\mathrm{d}x \\ -\sum_x p(x)\log p(x) \end{cases}$$

- entropy: uncertainty measurement
- maximize information by minimizing entropy

Information optimization

Information quantity

- use of entropy

$$H_p = \begin{cases} -\int p(x) \log p(x) \, \mathrm{d}x \\ -\sum_x p(x) \log p(x) \end{cases}$$

- entropy: uncertainty measurement
- maximize information by minimizing entropy

Information gain

- comparison between current and expected information

$$I_p(\boldsymbol{u}) = H_p - \mathsf{E}[H_{p'} \mid \boldsymbol{u}]$$

- unknown next observation

Exploration heuristics

Entropy

- correlation between entropy and information gain in an occupancy grid
- greedy method: choose best immediate action

Exploration heuristics

Entropy

- correlation between entropy and information gain in an occupancy grid
- greedy method: choose best immediate action

Uncertainty in unexplored space

- long-term plans are invalid
- greedy methods at the borders

Exploration heuristics

Entropy

- correlation between entropy and information gain in an occupancy grid
- greedy method: choose best immediate action

Uncertainty in unexplored space

- long-term plans are invalid
- greedy methods at the borders

Frontier-based exploration

- list known borders
- go explore nearest

04

Conclusion

Conclusion

Navigation

- motion decision
- adapt to robot: trajectory following
- adapt to environment: obstacle avoidance
- adapt to our knowledge: exploration

Conclusion

Navigation

- motion decision
- adapt to robot: trajectory following
- adapt to environment: obstacle avoidance
- adapt to our knowledge: exploration

Limits

- articulation between path planning and execution
- obstacle identification
- heuristic exploration

Bibliography

Books

- Thrun *et al.*, *Probabilistic Robotics*, MIT Press, 2005.
- Siciliano *et al.*, *Springer Handbook of Robotics*, 2nd ed., Springer, 2016.

Vector Field Histogram

- Ulrich et Borenstein, *VFH+: reliable obstacle avoidance for fast mobile robots*, RA, 1998.

Velocity obstacles

- Fiorini et Shiller, *Motion planning in dynamic environments using velocity obstacles*, IJRR 1998.

Exploration

- Holz *et al.*, *Evaluating the efficiency of frontier-based exploration*, ISR/Robotik, 2010.

informatics mathematics

Thanks for your attention

Questions?