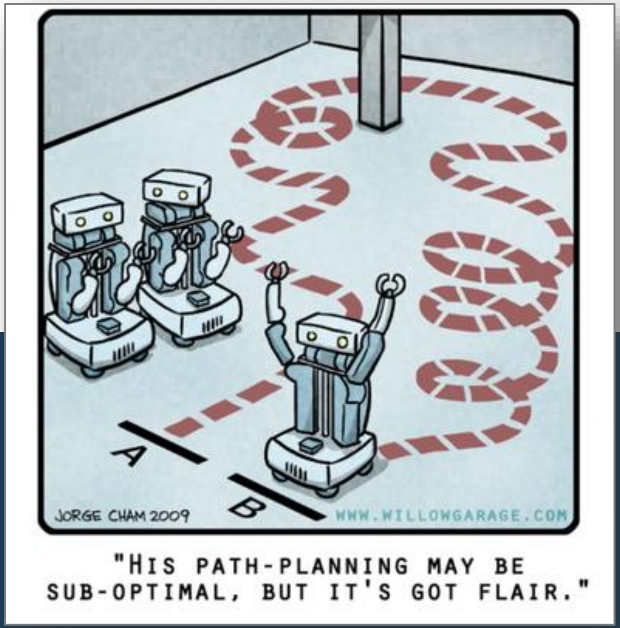


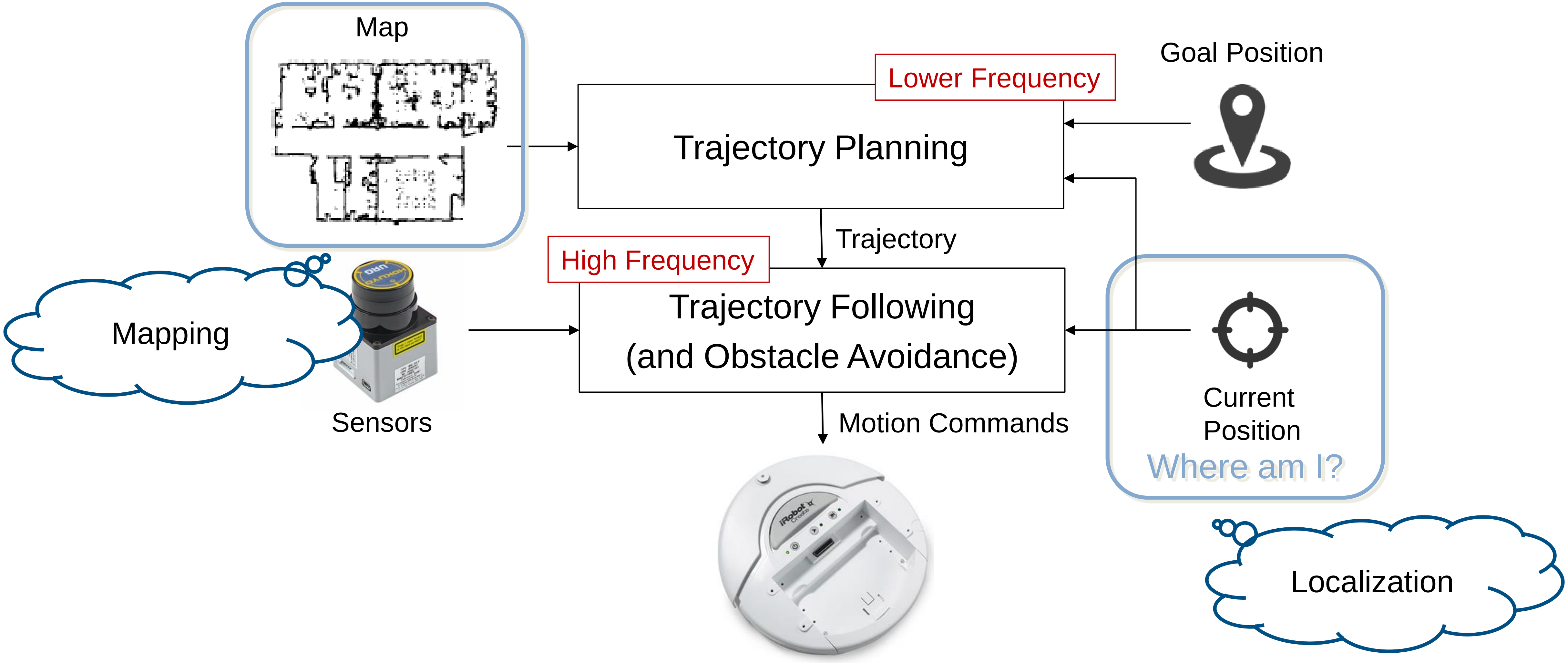
Robotics

Robot Motion Control

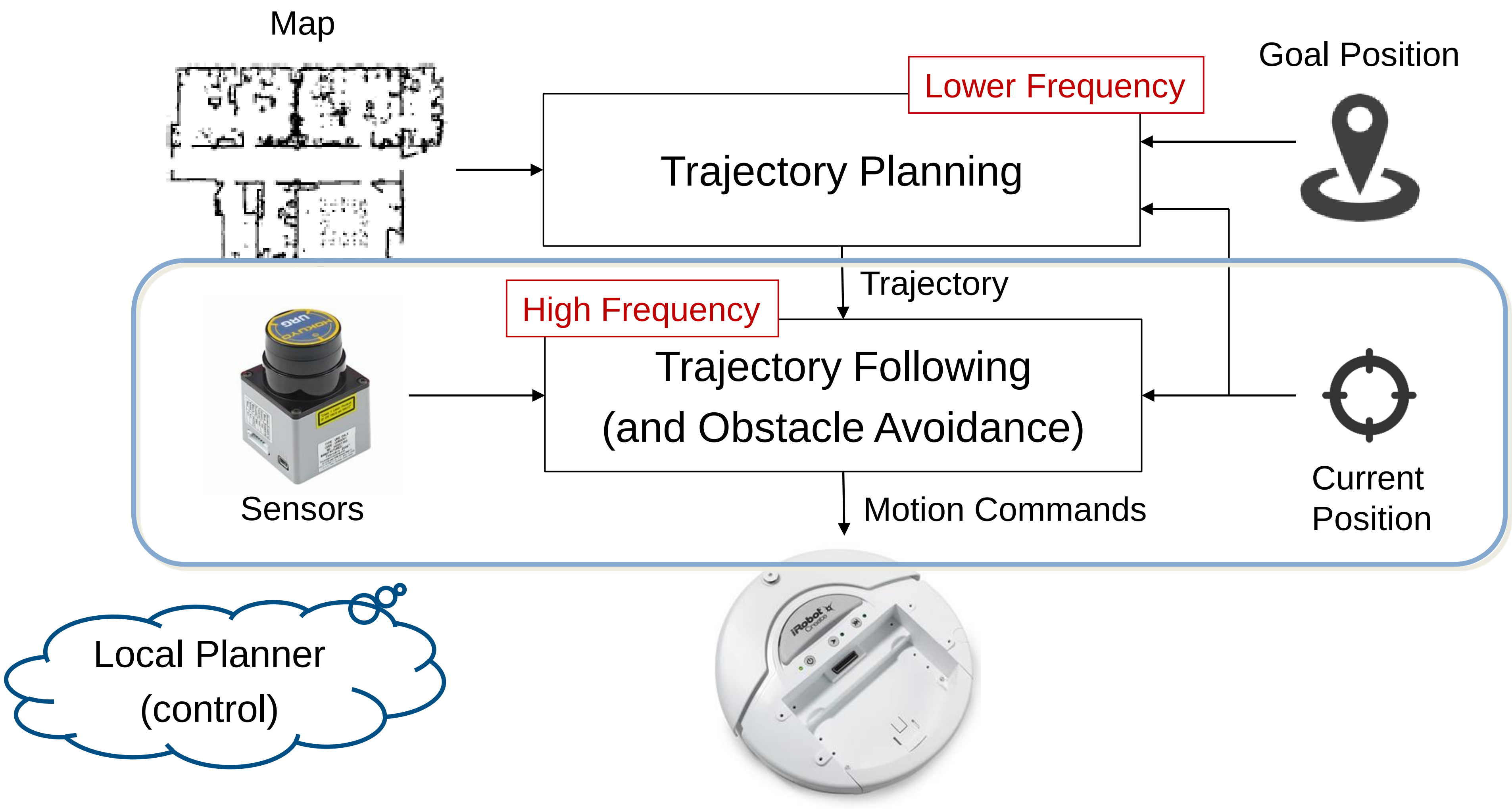
Matteo Matteucci
matteo.matteucci@polimi.it

Artificial Intelligence and Robotics Lab - Politecnico di Milano

A Simplified Sense-Plan-Act Architecture



A Simplified Sense-Plan-Act Architecture



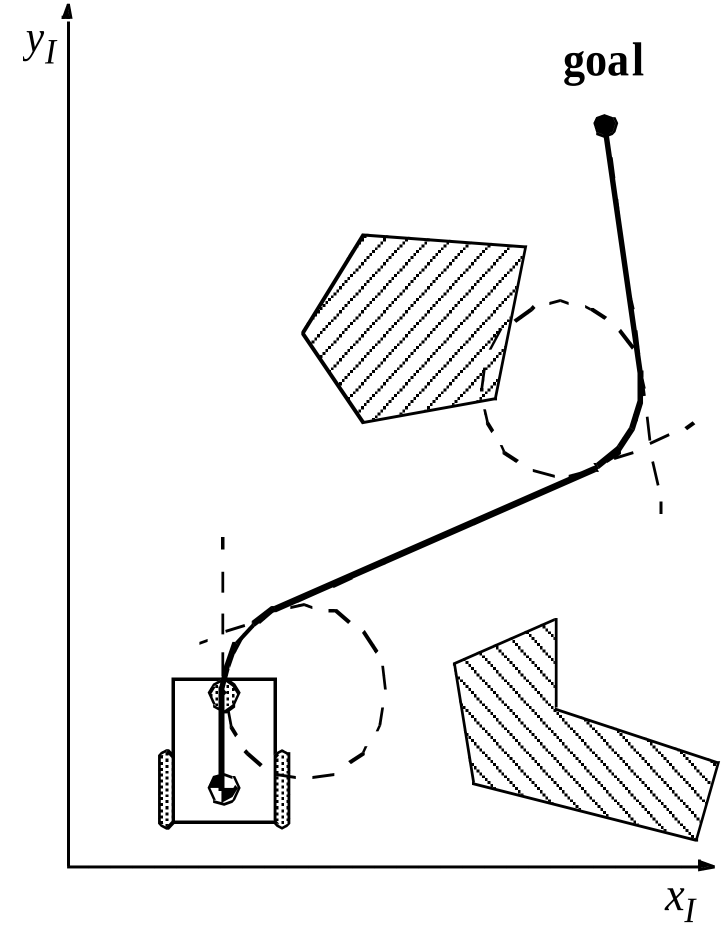
Open loop control

A mobile robot is meant to move from one place to another

- Pre-compute a smooth trajectory based on motion segments (e.g., line or circles) from start to goal
- Execute the planned trajectory till the goal

Disadvantages:

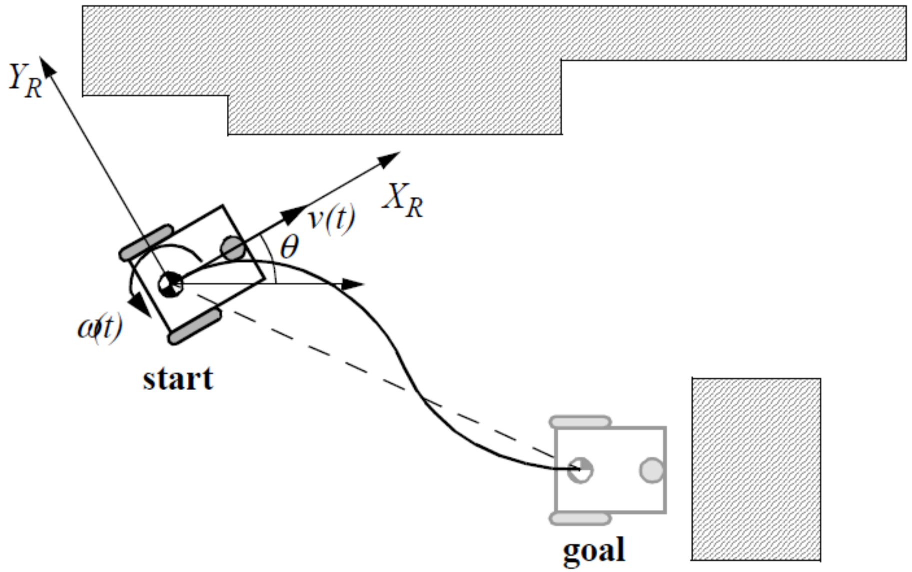
- Not easy to pre-compute a feasible trajectory
- Limitations and constraints of the robots velocities and accelerations
- Does not handle dynamical changes (obstacles)
- No recovery from errors

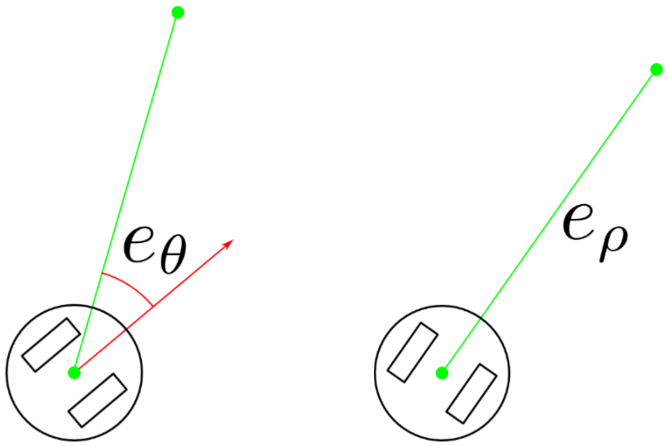


Feedback control (simple diff drive example)

The trajectory is recomputed / adapted online
via a simple control schema for path following

- Control orientation acting on angular velocity
- Control distance acting on linear velocity

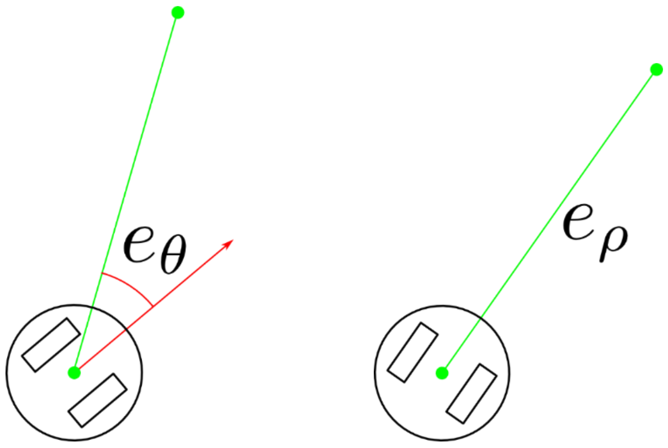


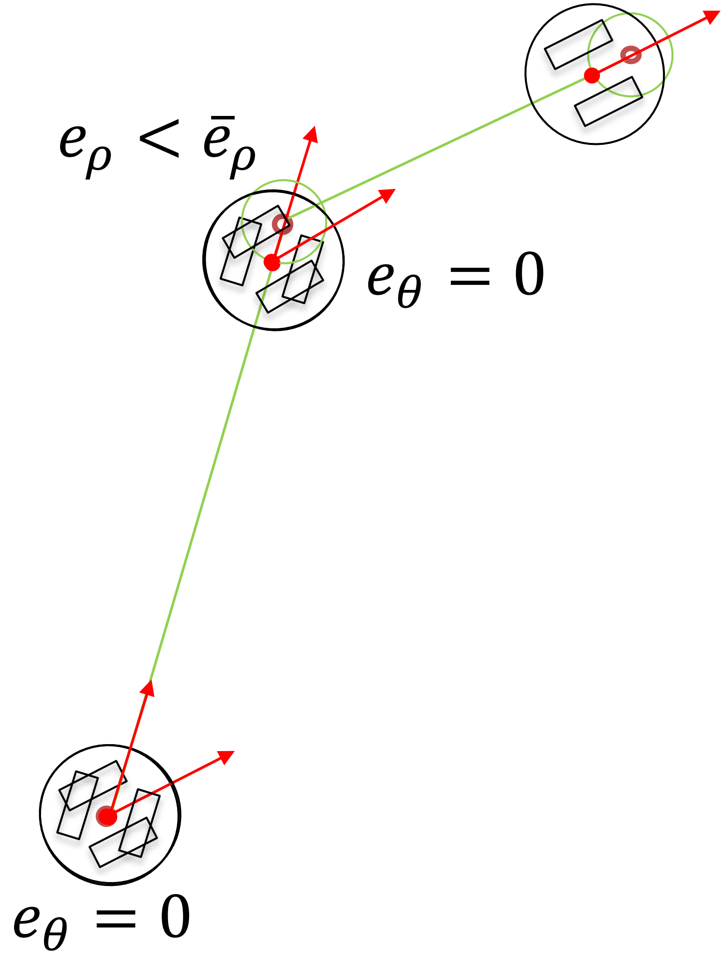


Feedback control (simple diff drive example)

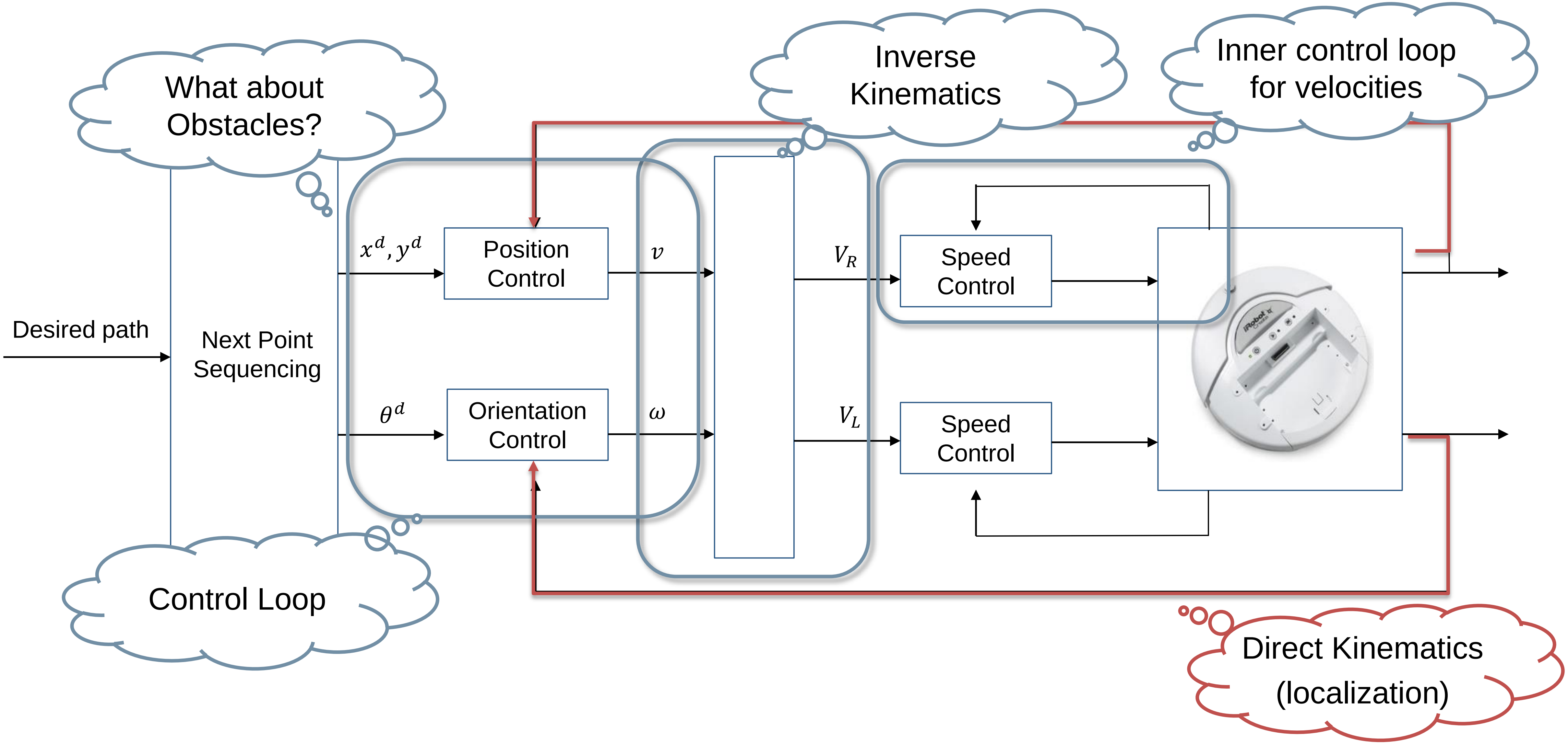
The trajectory is recomputed / adapted online
via a simple control schema for path following

- Control orientation acting on angular velocity
- Control distance acting on linear velocity





Feedback control (simple diff drive example)



Obstacle Avoidance (Local Path Planning)

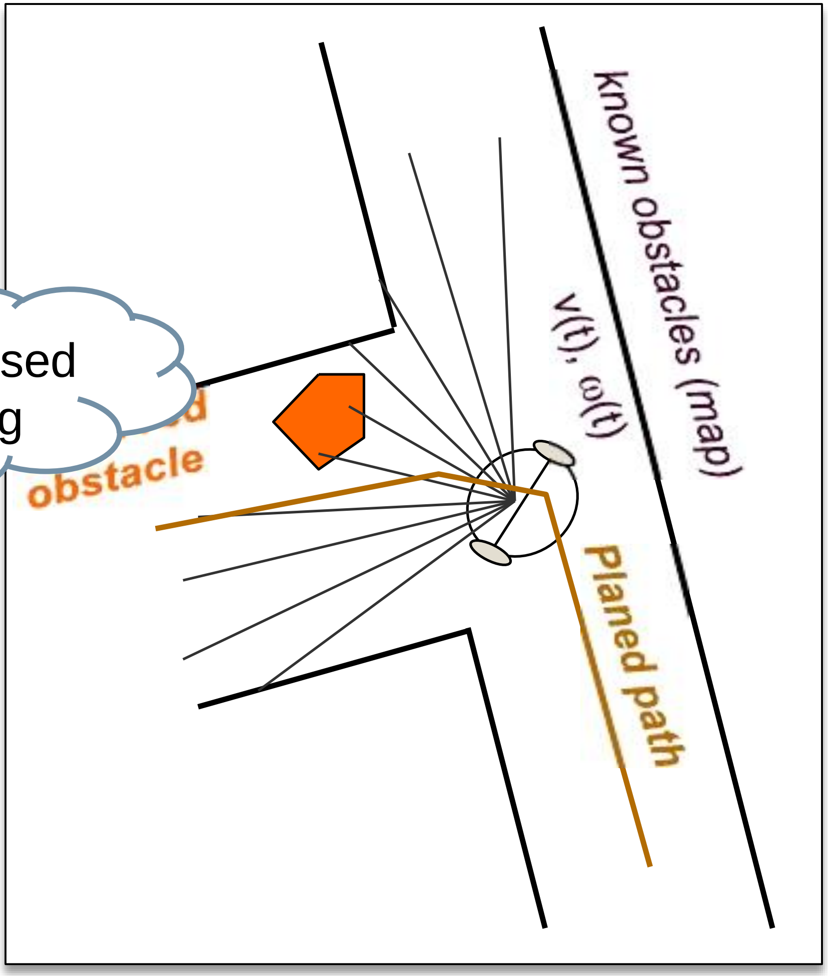
Obstacle avoidance should:

- Follow the planned path
- Avoid unexpected obstacle (i.e., not in the map)

Several proposed methods in the literature

- Potential field methods [Borenstein, 89]
- Vector field histogram [Borenstein, 91, 98, 00]
- Curvature-Velocity [Simmons, 96]
- Nearness diagram [Minguez & Montano, 00]
- Dynamic Window Approach [Fox, Burgard, Thrun, 97]
- …

Sometimes used for planning



The Simplest One …

"Bugs" have little if any knowledge …

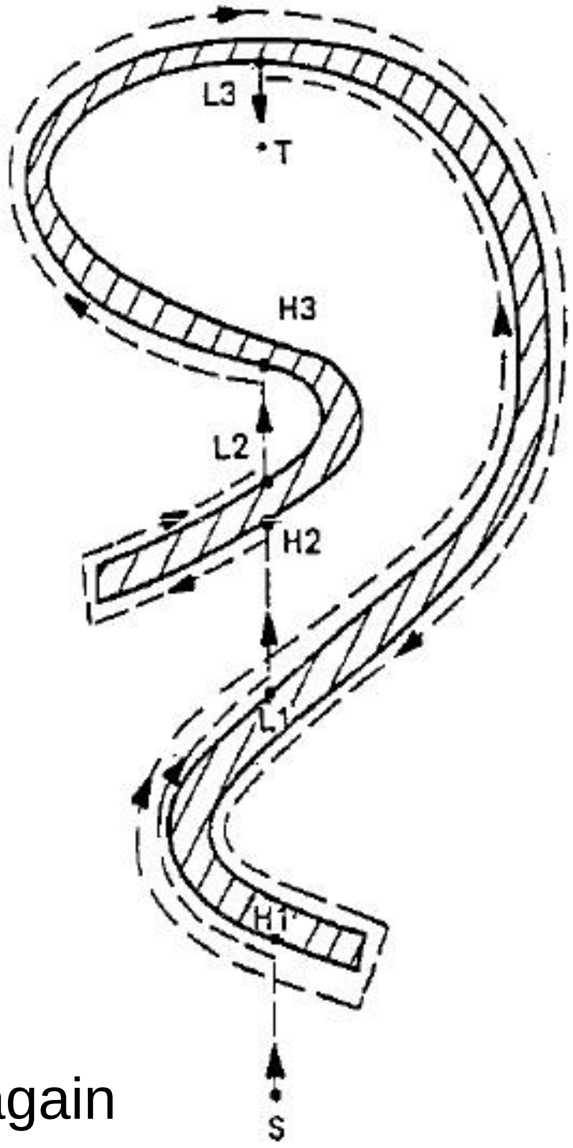
- They known the direction to the goal
- They have local sensing (obstacles + encoders)

… and their world is reasonable!

- Finite obstacles in any finite range
- A line intersects an obstacle finite times

Switch between two basic behaviors

1. Head toward goal
2. Follow obstacles until you can head toward the goal again

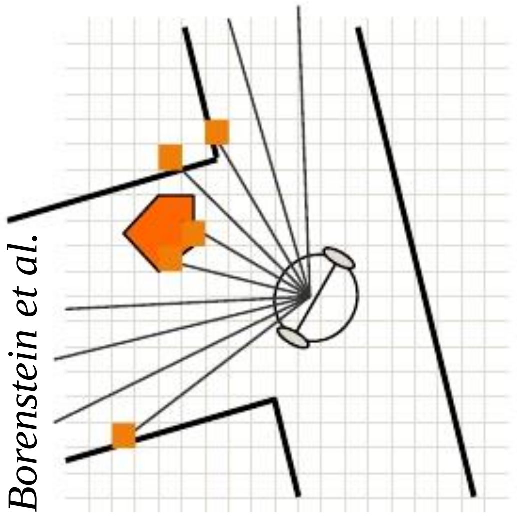


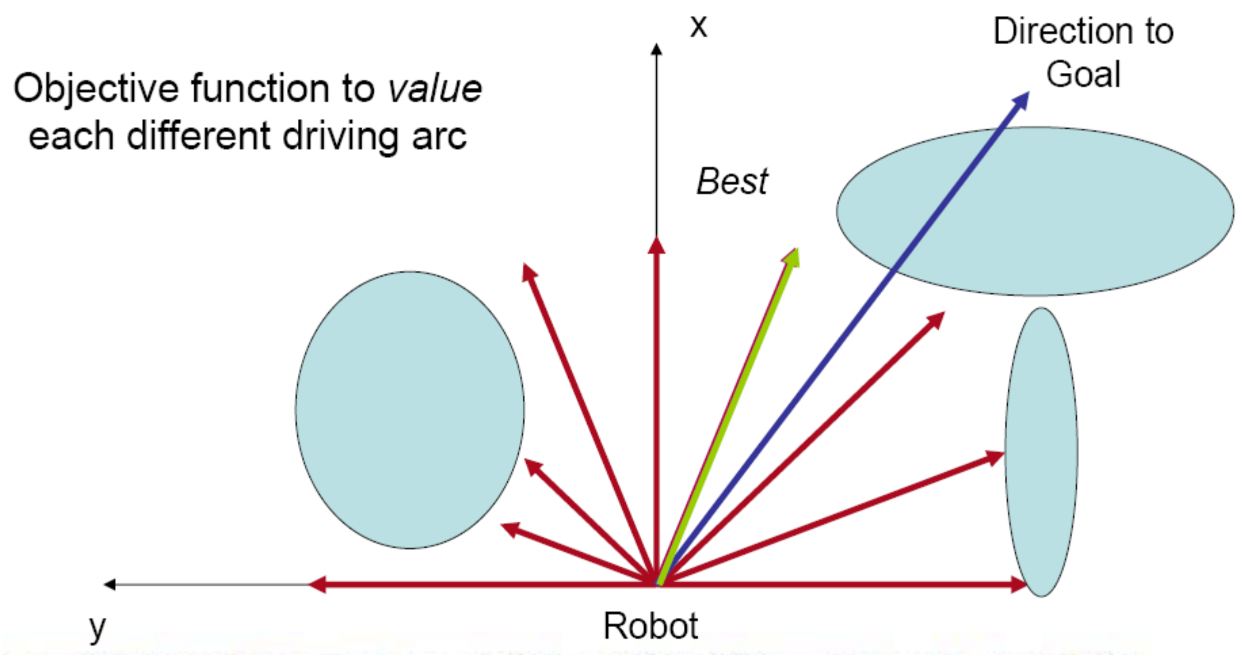
al

Vector Field Histograms (VHF) *[Borenstein et al. 1991]*

Use a local map of the environment and evaluate the angle to drive towards

- Environment represented in a grid (2 DOF) with local measurements
- All openings for the robot to pass are found

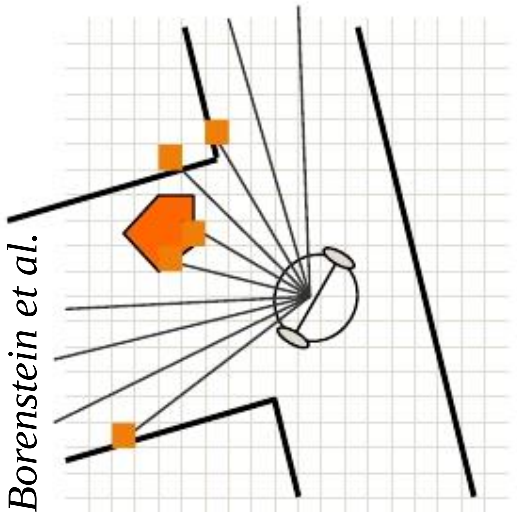




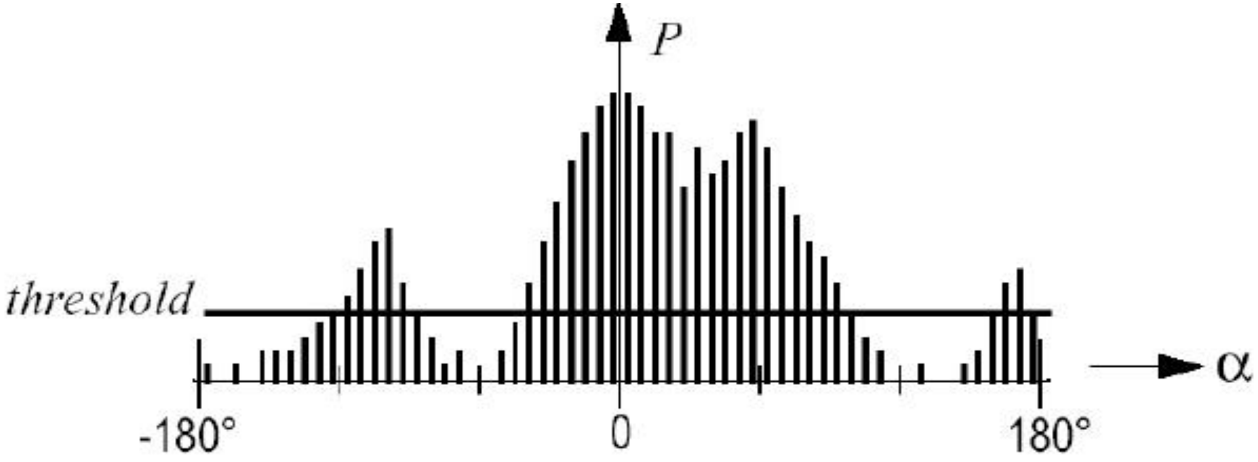
Vector Field Histograms (VHF) *[Borenstein et al. 1991]*

Use a local map of the environment and evaluate the angle to drive towards

- Environment represented in a grid (2 DOF) with local measurements
- All openings for the robot to pass are found
- The one with lowest cost is selected



$$G = a \cdot \text{target\_direction} + b \cdot \text{wheel\_orientation} + c \cdot \text{previous\_direction}$$



Vector Field Histograms (VHF) *[Borenstein et al. 1991]*

Use a local map of the environment and evaluate the angle to drive towards

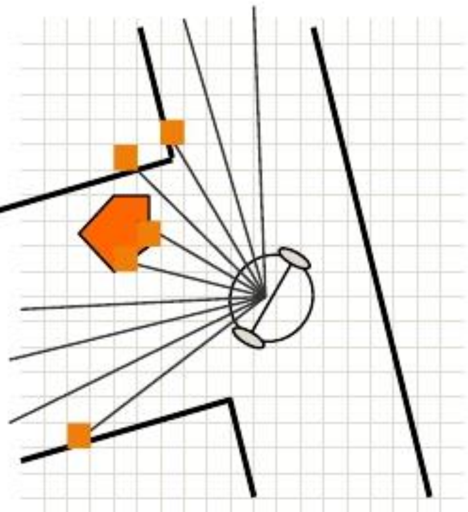
- Environment represented in a grid (2 DOF) with local measurements
- All openings for the robot to pass are found
- The one with

Alignment of the robot path with the goal

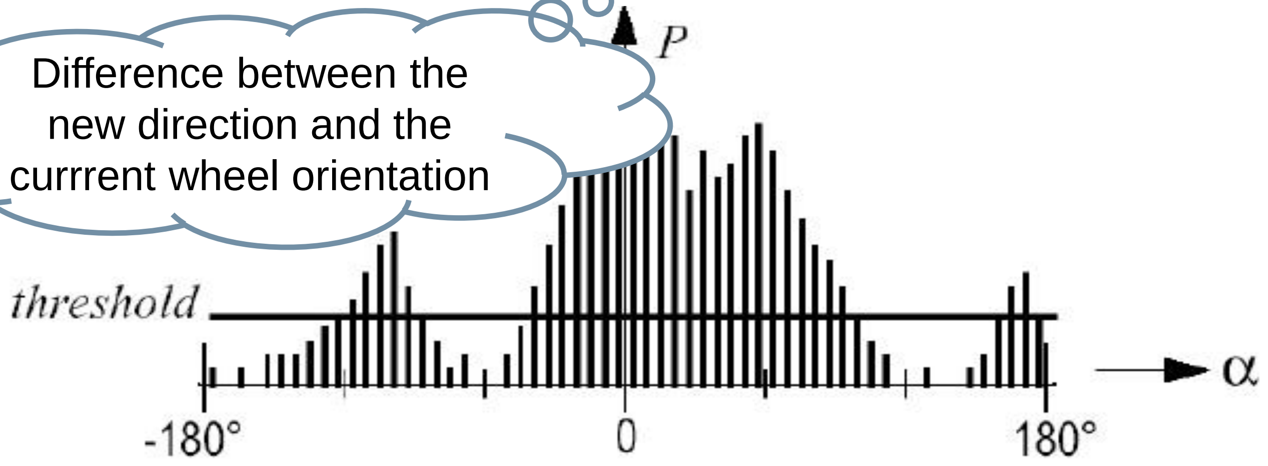
Difference between the previously selected direction and the new direction

$$G = a \cdot \text{target\_direction} + b \cdot \text{wheel\_orientation} + c \cdot \text{previous\_direction}$$

Difference between the new direction and the currrent wheel orientation



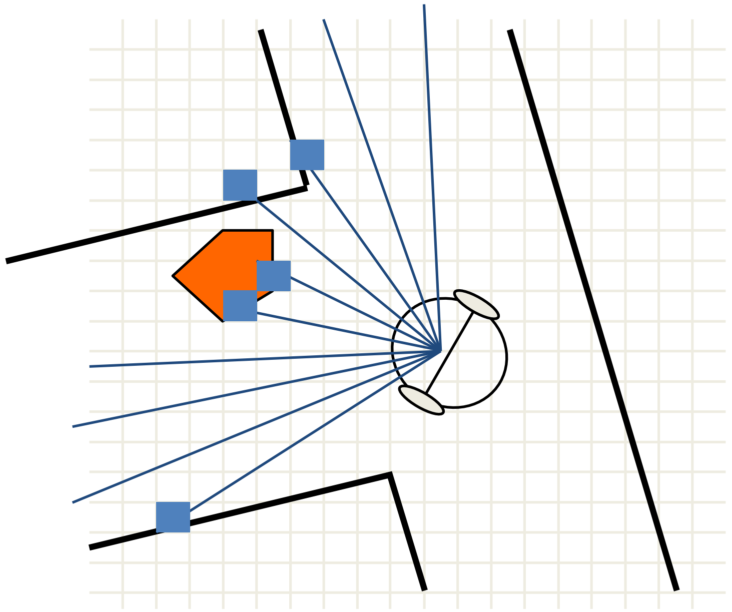
Borenstein et al.

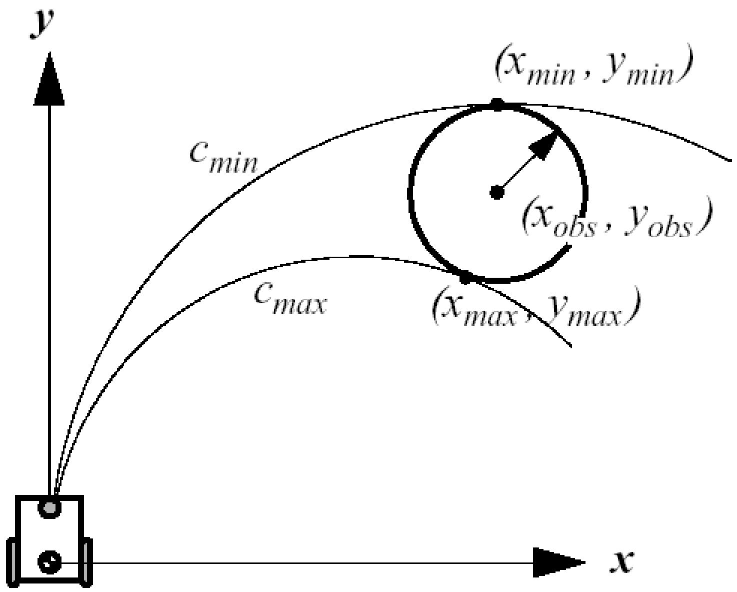


Curvature Velocity Methods (CVM) *[Simmons et al. 1996]*

CVMs add physical constraints from the robot and the environment on (v, w)

- Assumption that robot is traveling on arcs (c= w / v) with acceleration constraints
- Obstacles are transformed in velocity space
- An objective function to select the optimal speed



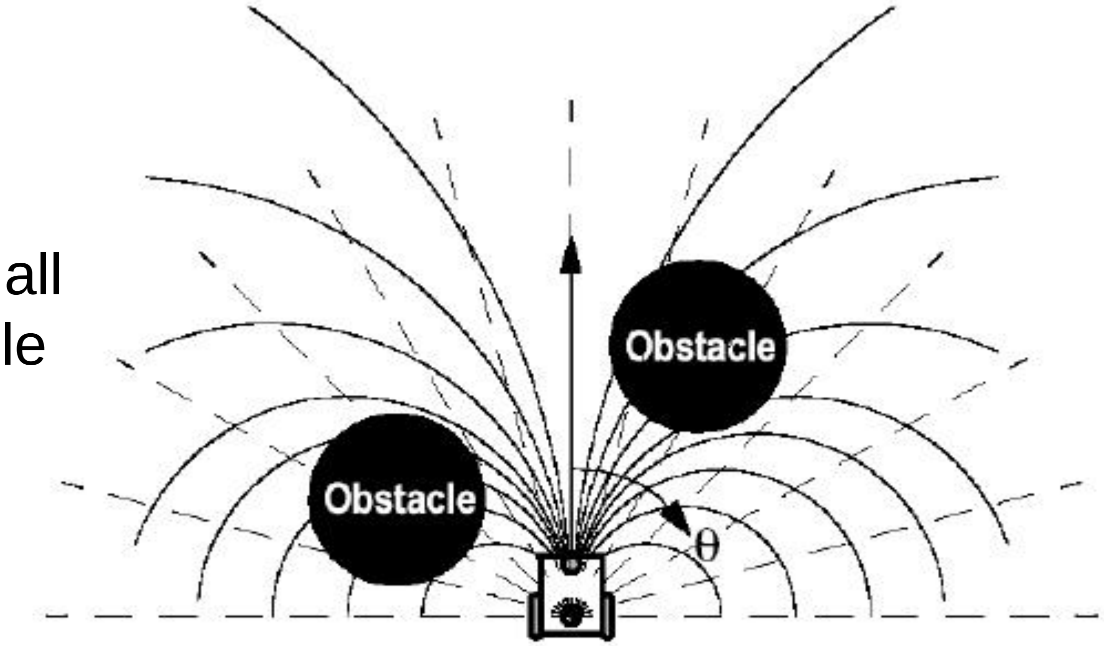


Simmons et al.

Vector Field Histogram+ (VFH+) *[Borenstein et al. 1998]*

VHF+ accounts also for vehicle kinematics

- Robot moving on arcs or straight lines
- Obstacles blocking a given direction blocks all trajectories (arcs) like in an Ackerman vehicle
- Obstacles are enlarged so to account for all kinematically blocked trajectories



Borenstein et al.

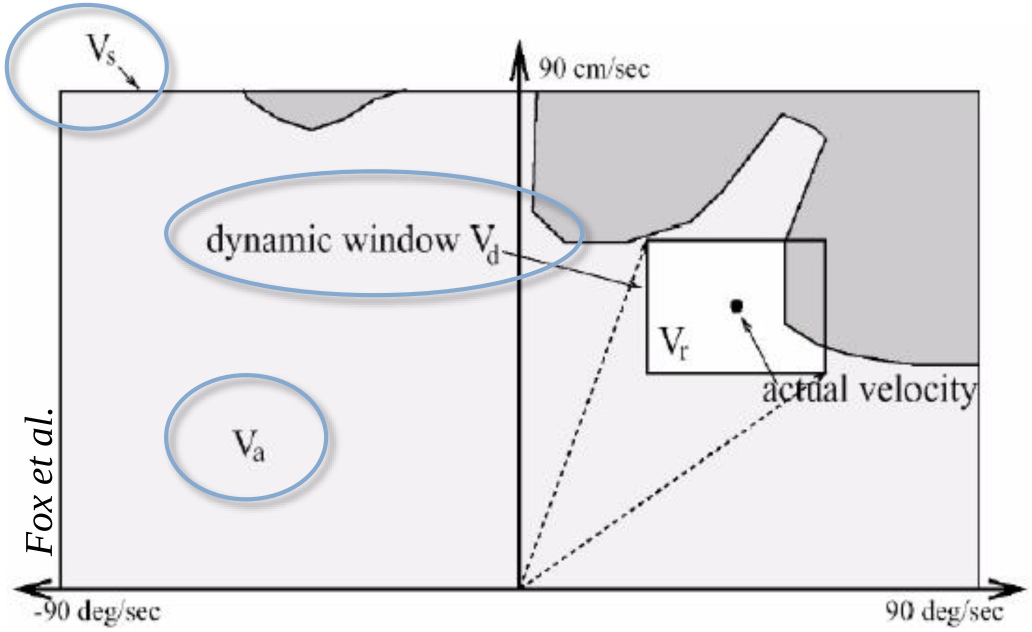
However VHF+ as VHF suffers

- Limitation if narrow areas (e.g. doors) have to be passed
- Local minima might not be avoided
- Reaching of the goal can not be guaranteed
- Dynamics of the robot not really considered

Dynamic Window Approach (DWA) *[Fox et al. 1997]*

The kinematics of the robot are considered via local search in velocity space:

- Consider only circular trajectories via pairs $V_s=(v,\omega)$ of linear and angular speeds
- $V_a=(v, \omega)$ is admissible, if the robot is able to stop before the closest obstacle
- A dynamic window restricts the reachable velocities V_d to those that can be reached within a short time given limited robot accelerations



$$V_d = \begin{cases} v \in [v - a_{tr} \cdot t, v + a_{tr} \cdot t] \\ \omega \in [\omega - a_{rot} \cdot t, \omega + a_{rot} \cdot t] \end{cases}$$

DWA Search Space

$$V_r = V_s \cap V_a \cap V_d$$

How to choose (v,ω)?

Steering commands are chosen maximizing a heuristic navigation function:

- Minimize the travel time by "driving fast in the right direction"
- Planning restricted to V_r space [Fox, Burgard, Thrun '97]

$$G(v,\omega) = \sigma\left(\underbrace{\alpha \cdot heading(v,\omega)}_{\text{Alignment with target direction}} + \underbrace{\beta \cdot dist(v,\omega)}_{\text{Distance to closest obstacle intersecting with curvature}} + \underbrace{\gamma \cdot velocity(v,\omega)}_{\text{Forward velocity of the robot}}\right)$$

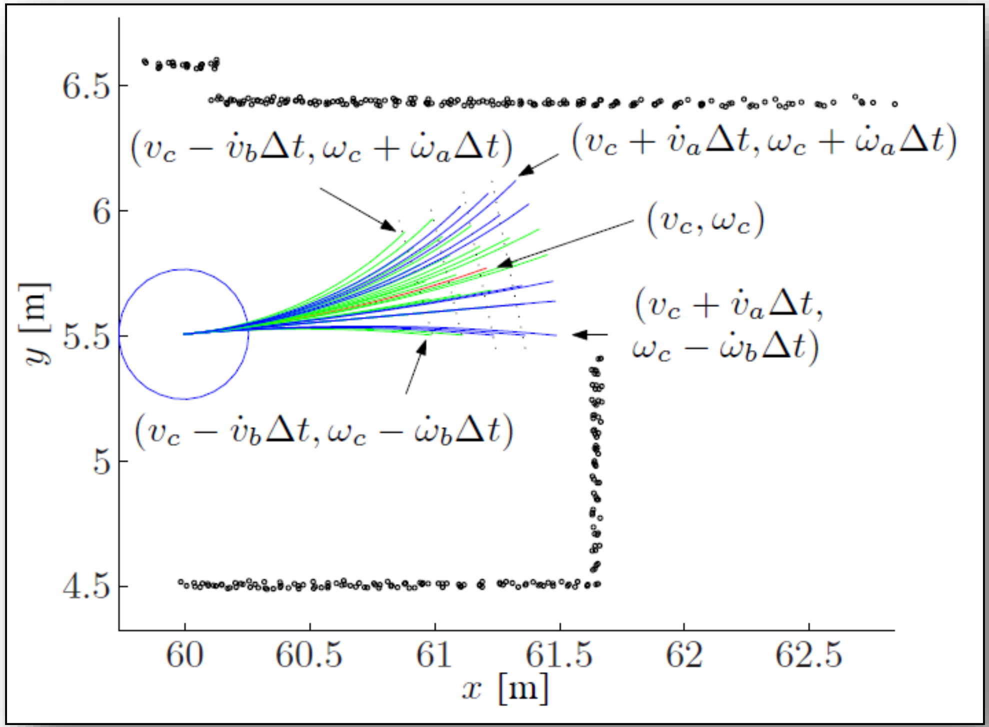
- Global approach [Brock & Khatib 99] in <x,y>-space uses

$$NF = \overbrace{\alpha \cdot vel}^{\text{Forward robot velocity}} + \underbrace{\beta \cdot nf}_{\text{Cost to reach the goal}} + \overbrace{\gamma \Delta nf}^{\text{Follows global path}} + \underbrace{\delta goal}_{\text{Goal nearness}}$$

DWA Algorithm (via trajectory rollout)

The basic idea of DWA … but with samples

1. Discretely sample robot control space
2. For each sampled velocity, perform forward simulation to predict what would happen if applied for some (short) time.
3. Evaluate (score) each trajectory resulting from the forward simulation
4. Discard illegal trajectories, i.e., those that collide with obstacles, and pick the highest-scoring trajectory



Can handle non circurar trajectories

Clothoid: $S(x) = \int_0^x \sin(t^2)\,dt, \quad C(x) = \int_0^x \cos(t^2)\,dt.$

A Simplified Sense-Plan-Act Architecture