



# 4SC020 Mobile Robot Control 2024: Local navigation

MAY 1<sup>ST</sup> 2024

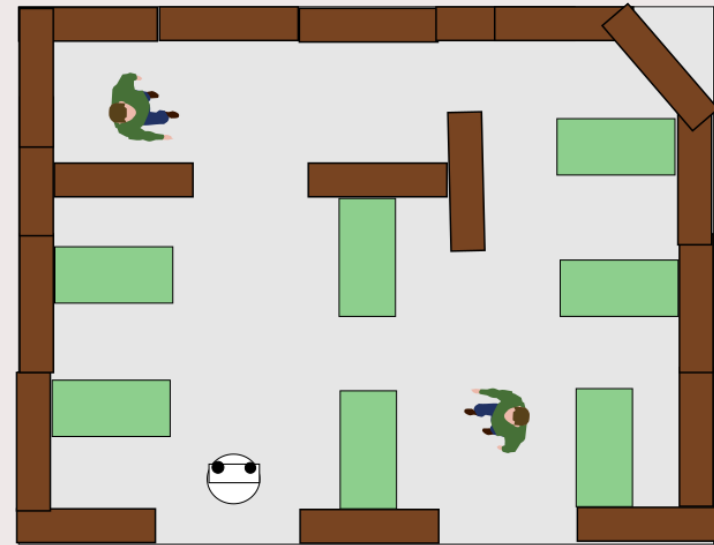
Aron Aertssen

# Outline

- Robot navigation problem
- Local navigation algorithms: properties
- Local navigation algorithms: examples
- Recap
- Assignment

# Robot navigation problem / introduction

- What is the robot navigation problem?



# Robot navigation problem / introduction

- What is the robot navigation problem?
  - Find a feasible path or trajectory from a given initial pose (A) to the desired final pose (B)



# Robot navigation problem / introduction

- What is the robot navigation problem?
  - Find a feasible path or trajectory from a given initial pose (A) to the desired final pose (B)
- This raises a question: where is A and where is B?



# Robot navigation problem / local vs. global

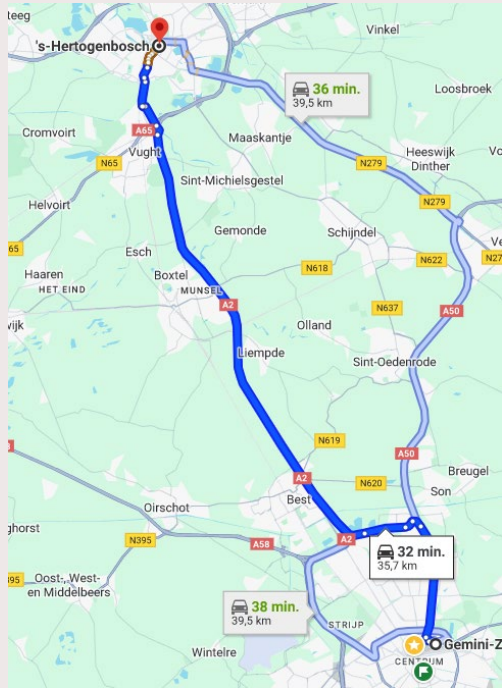
- Division into global and local navigation. Why?



# Robot navigation problem / local vs. global

- Division into global and local navigation. Why?
  1. Reduce complexity
    - Global: compute path from start to goal
    - Local: move towards the goal using the global path as a guide

# Robot navigation problem / local vs. global



Global



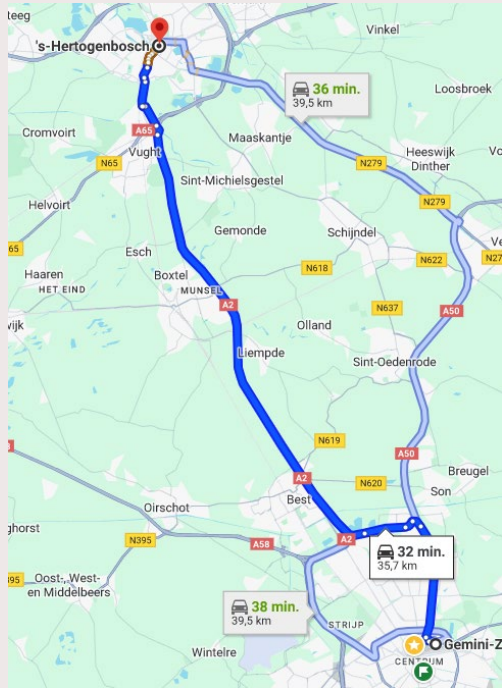
Local



# Robot navigation problem / local vs. global

- Division into global and local navigation. Why?
  1. Reduce complexity
  2. Static vs. dynamic environment
    - Global: static environment
    - Local: uncertain, dynamic environment

# Robot navigation problem / local vs. global



Global



Local

# Robot navigation problem / local vs. global

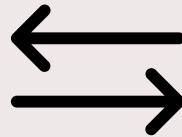
- Division into global and local navigation. Why?
  1. Reduce complexity
  2. Static vs. dynamic environment
  3. Global world model often incomplete
    - More information might come with time

# Robot navigation problem / local vs. global

- Division into global and local navigation. Why?
- Where is local and global?

# Robot navigation problem / local vs. global

- Division into global and local navigation. Why?
- Where is local and global?
  - Problem-dependent, but in general:
    - Local: sensor-range
    - Global: map
  - **Note:** explicitly define local and global to avoid confusion!



# Robot navigation problem / local vs. global

- Division into global and local navigation. Why?
- Where is local and global?
- This week: **local navigation**
  - How can we solve this?
- Next week: global navigation

# Local navigation algorithms / properties


- Goal of local navigation: go from A to B, using the **global path** as a guide




# Local navigation algorithms / properties


- Goal of local navigation: go from A to B, using the global path as a guide
- Properties of local navigation algorithms


 Completeness: finding a path if one exists

 Optimality: finding the optimal path (time, energy, distance, ...)

 Computational complexity: scalability

 Robustness against a dynamic environment


 Robustness against uncertainty


 Kinematic and dynamic constraints





# Local navigation algorithms / properties


- Last week's exercise: the art of nothing crashing
  - Let the robot drive forward and let it stop before it hits anything
  - **How to go to a certain goal?**
  - We want to balance not crashing and reaching the goal
- Several approach exist, we will discuss three today


 Completeness: finding a path if one exists

 Optimality: finding the optimal path (time, energy, distance, ...)

 Computational complexity: scalability

 Robustness against a dynamic environment

 Robustness against uncertainty

 Kinematic and dynamic constraints

# Local navigation algorithms / examples

- Three examples
  - Artificial potential fields
  - Dynamic window approach
  - Vector field histograms

# Local navigation algorithms / examples

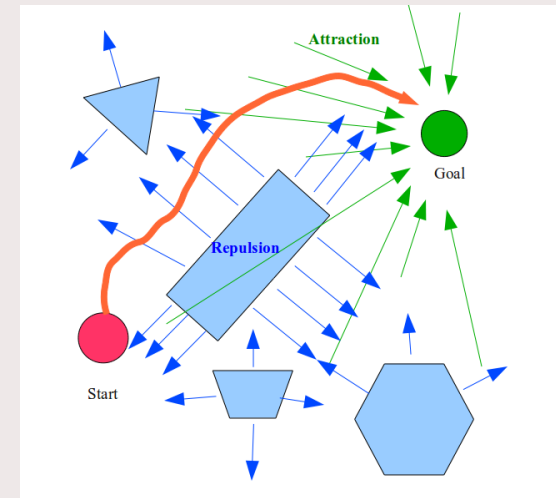
- Three examples
- Assumptions
  - A global path is available
  - Robot position is known
  - Obstacle positions are known

# Local navigation algorithms / examples

- Three examples
- Assumptions
- Note that the explained algorithms directly provide **control outputs**
  - Often, a **path** is the output of a local navigation algorithm with requires a path following controller to obtain control outputs

# Local navigation algorithms / artificial potential fields

- Artificial potentials
  - Attraction towards goal
  - Repulsion from obstacles
  - Think about marbles



<https://sudonull.com/post/62343-What-robotics-can-teach-gaming-AI>

# Local navigation algorithms / artificial potential fields

- Artificial potentials
- Amplitude based on distance to object  $\mathbf{q}_j^o$  and goal  $\mathbf{q}_{goal}$  (See Chap 12.6 of [1])

- Note
  - $\mathbf{q}$  is the robot configuration, in example:  $\mathbf{q} = [x, y]$
  - $k, \rho_o > 0$
  - 'Goal' is next point of global path
  - $\|\mathbf{q} - \mathbf{q}_j^o\|$  is to the closest point of the object

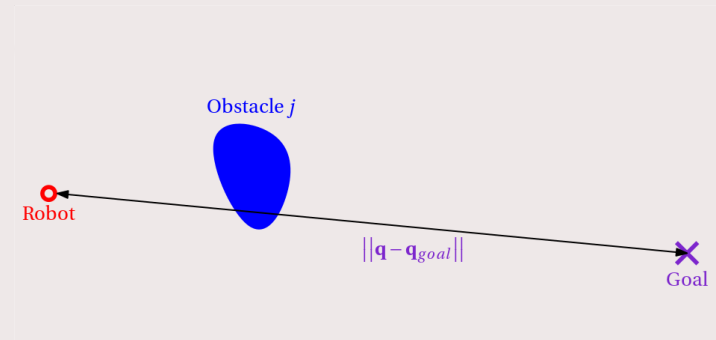


[1] B. Siciliano, L. Sciacivco, L. Villani, and G. Oriolo, "Robotics: Modelling, Planning and Control," Springer Publishing Company, Incorporated, 2010

# Local navigation algorithms / artificial potential fields

- Artificial potentials
- Amplitude based on distance to object  $\mathbf{q}_j^o$  and goal  $\mathbf{q}_{goal}$  (See Chap 12.6 of [1])
- $U_{att}(\mathbf{q}) = \frac{1}{2}k_a(\|\mathbf{q} - \mathbf{q}_{goal}\|)^2$

- Note
  - $\mathbf{q}$  is the robot configuration, in example:  $\mathbf{q} = [x, y]$
  - $k, \rho_o > 0$
  - 'Goal' is next point of global path
  - $\|\mathbf{q} - \mathbf{q}_j^o\|$  is to the closest point of the object



[1] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, "Robotics: Modelling, Planning and Control," Springer Publishing Company, Incorporated, 2010

# Local navigation algorithms / artificial potential fields

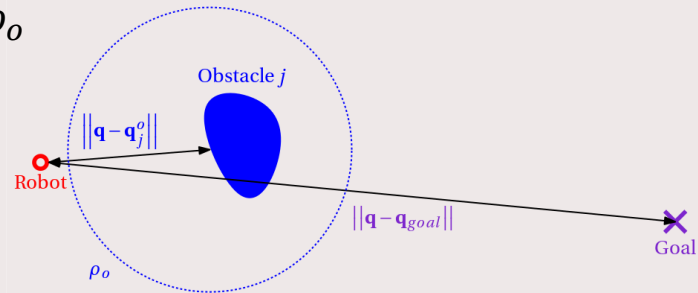
- Artificial potentials
- Amplitude based on distance to object  $\mathbf{q}_j^o$  and goal  $\mathbf{q}_{goal}$  (See Chap 12.6 of [1])

$$U_{att}(\mathbf{q}) = \frac{1}{2} k_a (\|\mathbf{q} - \mathbf{q}_{goal}\|)^2$$

$$U_{rep,j}(\mathbf{q}) = \begin{cases} \frac{1}{2} k_{rep,j} \left( \frac{1}{\|\mathbf{q} - \mathbf{q}_j^o\|} - \frac{1}{\rho_o} \right)^2 & \text{if } \|\mathbf{q} - \mathbf{q}_j^o\| \leq \rho_o \\ 0 & \text{if } \|\mathbf{q} - \mathbf{q}_j^o\| > \rho_o \end{cases}$$

- Note

- $\mathbf{q}$  is the robot configuration, in example:  $\mathbf{q} = [x, y]$
- $k, \rho_o > 0$
- 'Goal' is next point of global path
- $\|\mathbf{q} - \mathbf{q}_j^o\|$  is to the closest point of the object

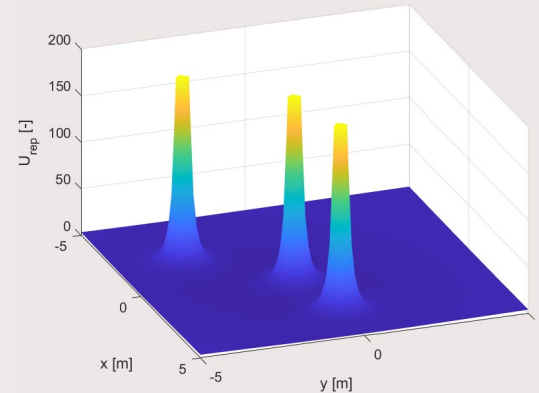
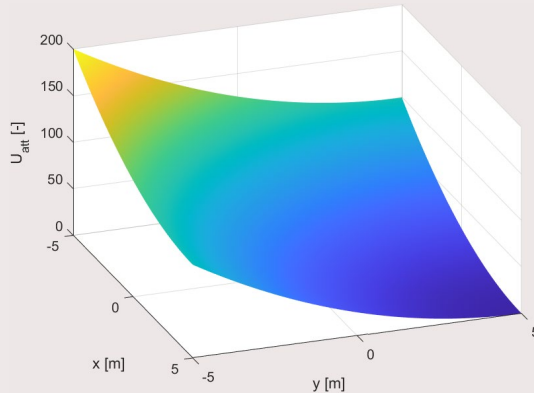


[1] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, "Robotics: Modelling, Planning and Control," Springer Publishing Company, Incorporated, 2010



# Local navigation algorithms / artificial potential fields

- Artificial potentials
- Amplitude based on distance to object  $q_j^0$  and goal  $q_{goal}$  (See Chap 12.6 of [1])



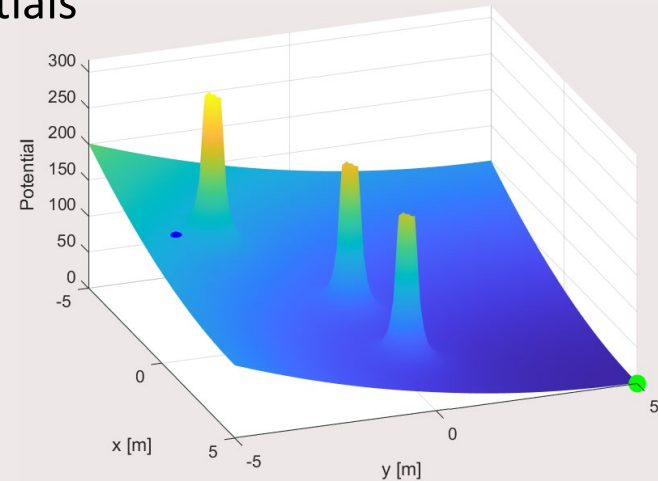
- Starting point:  $[-2, -4]$
- Goal point:  $[5, 5]$
- 3 obstacles:  $[-2, -3], [0.5, -0.5], [3, 0]$

[1] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, "Robotics: Modelling, Planning and Control," Springer Publishing Company, Incorporated, 2010

# Local navigation algorithms / artificial potential fields

- Artificial potentials
- Amplitude based on distance to object  $\mathbf{q}_j^0$  and goal  $\mathbf{q}_{goal}$
- Total potential field is the sum of individual potentials

$$U(\mathbf{q}) = U_{att}(\mathbf{q}) + \sum_{j=1}^n U_{rep,j}(\mathbf{q})$$



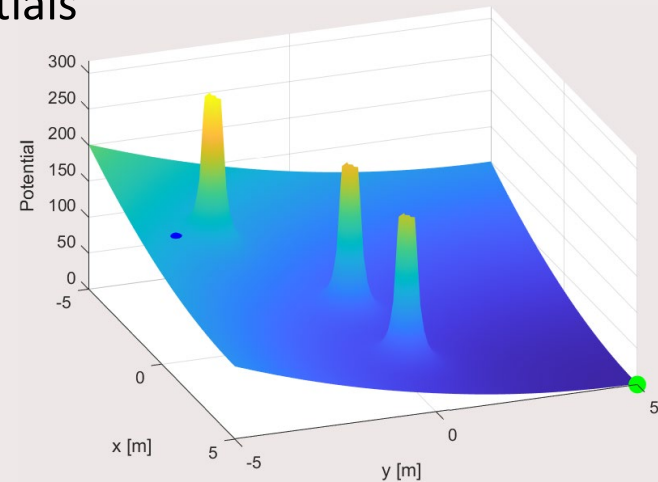
- Starting point:  $[-2, -4]$
- Goal point:  $[5, 5]$
- 3 obstacles:  $[-2, -3], [0, 5], [3, 0]$

[1] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, "Robotics: Modelling, Planning and Control," Springer Publishing Company, Incorporated, 2010

# Local navigation algorithms / artificial potential fields

- Artificial potentials
- Amplitude based on distance to object  $q_j^o$  and goal  $q_{goal}$
- Total potential field is the sum of individual potentials
- The artificial force acting on the robot is then

$$F(\mathbf{q}) = -\nabla U(\mathbf{q})$$

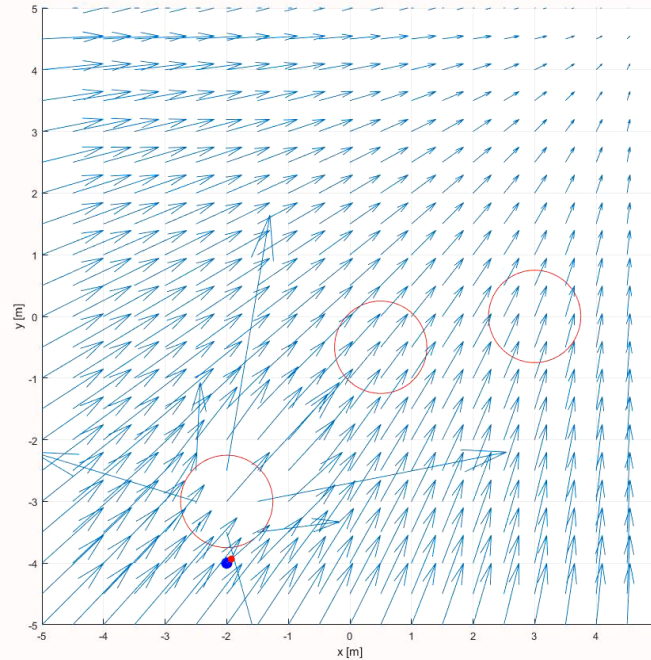


[1] B. Siciliano, L. Sciacicco, L. Villani, and G. Oriolo, "Robotics: Modelling, Planning and Control," Springer Publishing Company, Incorporated, 2010

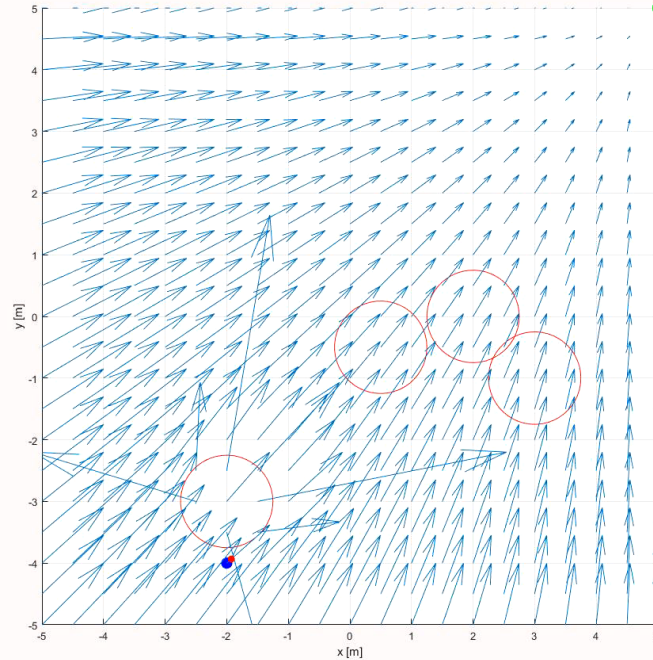
# Local navigation algorithms / artificial potential fields

- Artificial potentials
- Amplitude based on distance to object  $\mathbf{q}_j^0$  and goal  $\mathbf{q}_{goal}$
- Total potential field is the sum of individual potentials
- The artificial force acting on the robot is then
- How to use that force?
  - Point mass:  $m\ddot{\mathbf{q}} = F(\mathbf{q})$
  - Desired velocity:  $\dot{\mathbf{q}} = F(\mathbf{q})$

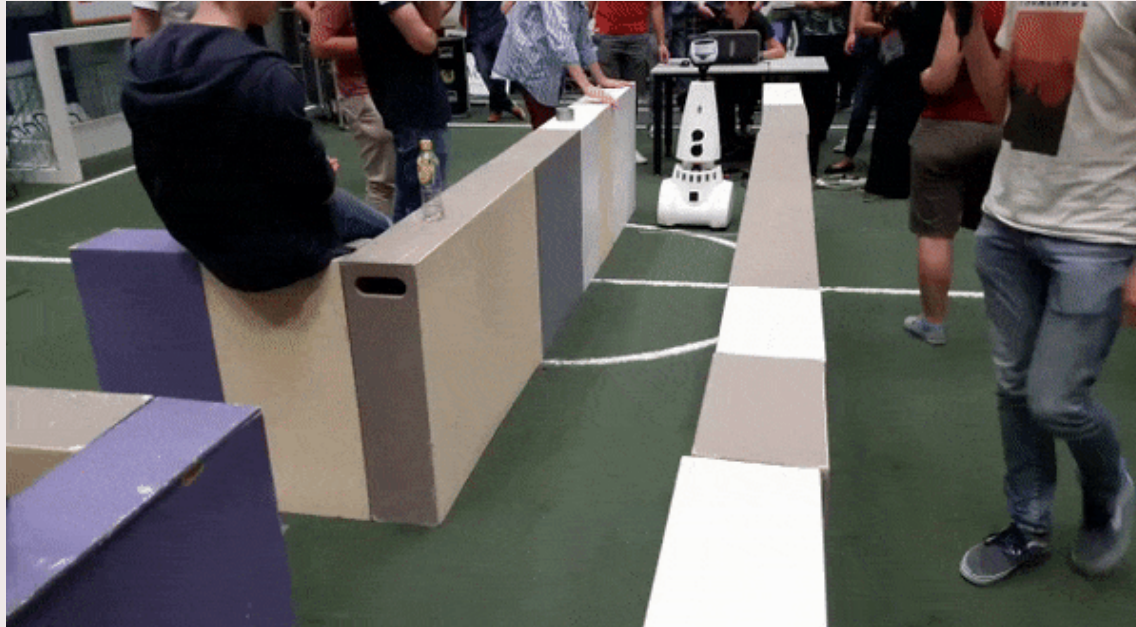
# Local navigation algorithms / artificial potential fields



# Local navigation algorithms / artificial potential fields



# Local navigation algorithms / artificial potential fields



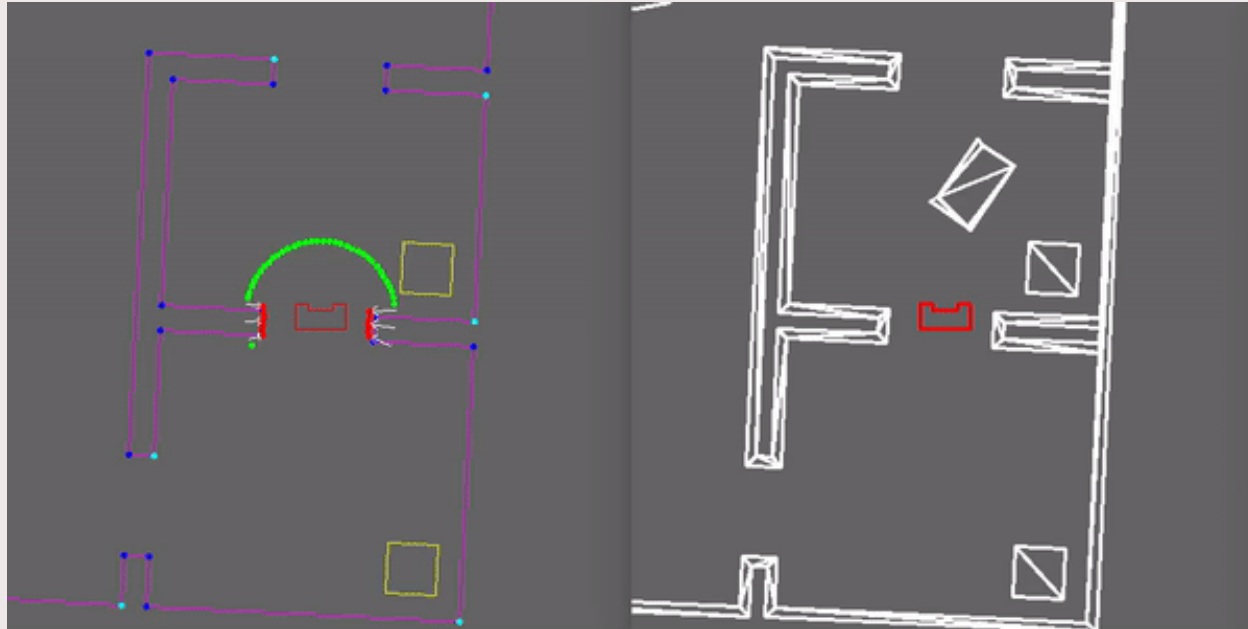
EMC 2017 – Group 10

# Local navigation algorithms / artificial potential fields

- Artificial potentials
- Amplitude based on distance to object  $\mathbf{q}_j^0$  and goal  $\mathbf{q}_{goal}$
- Total potential field is the sum of individual potentials
- The artificial force acting on the robot is then
- How to use that force?
- In the simulation videos we know everything... **How to do it in real life?**



# Local navigation algorithms / artificial potential fields

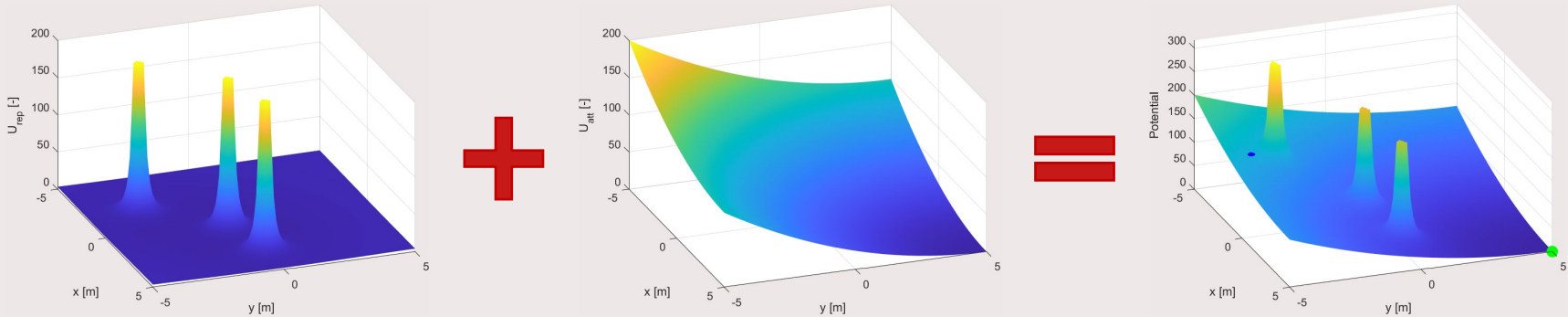


Simulation - MRC 2019 – Group 2

# Local navigation algorithms / artificial potential fields

- Artificial potentials
- Amplitude based on distance to object  $q_j^0$  and goal  $q_{goal}$
- Total potential field is the sum of individual potentials
- The artificial force acting on the robot is then
- How to use that force?
- In the simulation videos we know everything... **How to do it in real life?**
  - How to represent obstacles from laser points?
  - Include size of the robot

# Local navigation algorithms / artificial potential fields

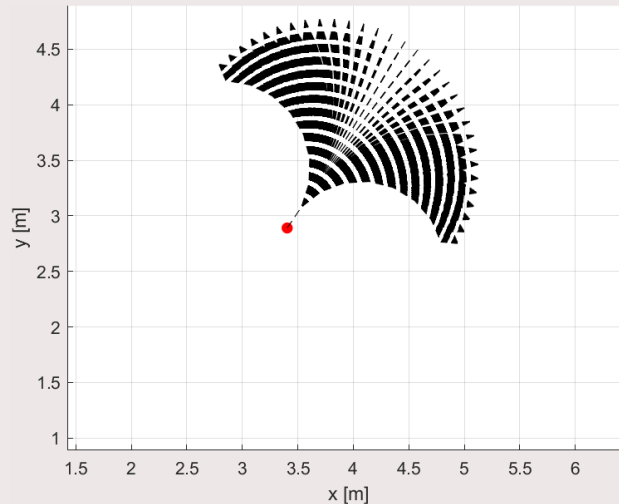


## Questions?

[1] B. Siciliano, L. Sciacicco, L. Villani, and G. Oriolo, "Robotics: Modelling, Planning and Control," Springer Publishing Company, Incorporated, 2010

# Local navigation algorithms / dynamic window approach

- Reactive collision avoidance based on robot dynamics
  - Intuition: certain velocity during certain time, see where we end and select most optimal



D. Fox, W. Burgard and S. Thrun, "The dynamic window approach to collision avoidance," in *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23-33, March 1997, doi: 10.1109/100.580977.

# Local navigation algorithms / dynamic window approach

- Reactive collision avoidance based on robot dynamics
- Consider velocities  $(v, \omega)$  during  $t$ , where  $(v, \omega)$  have to be

D. Fox, W. Burgard and S. Thrun, "The dynamic window approach to collision avoidance," in *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23-33, March 1997, doi: 10.1109/100.580977.

# Local navigation algorithms / dynamic window approach

- Reactive collision avoidance based on robot dynamics
- Consider velocities  $(v, \omega)$  during  $t$ , where  $(v, \omega)$  have to be
  - Possible: velocities are limited by robot's dynamics

$$V_s = \{v, \omega \mid v \in [v_{min}, v_{max}] \wedge \omega \in [\omega_{min}, \omega_{max}]\}$$

# Local navigation algorithms / dynamic window approach

- Reactive collision avoidance based on robot dynamics
- Consider velocities  $(v, \omega)$  during  $t$ , where  $(v, \omega)$  have to be
  - Possible: velocities are limited by robot's dynamics
  - Admissible: robot can stop before reaching closest obstacle

$$V_a = \left\{ v, \omega \mid v \leq \sqrt{2d(v, \omega)\dot{v}_b} \wedge \omega \leq \sqrt{2d(v, \omega)\dot{\omega}_b} \right\}$$

$\dot{v}_b$  and  $\dot{\omega}_b$  are maximum deceleration values

$d(v, \omega)$  is distance to closest object

# Local navigation algorithms / dynamic window approach

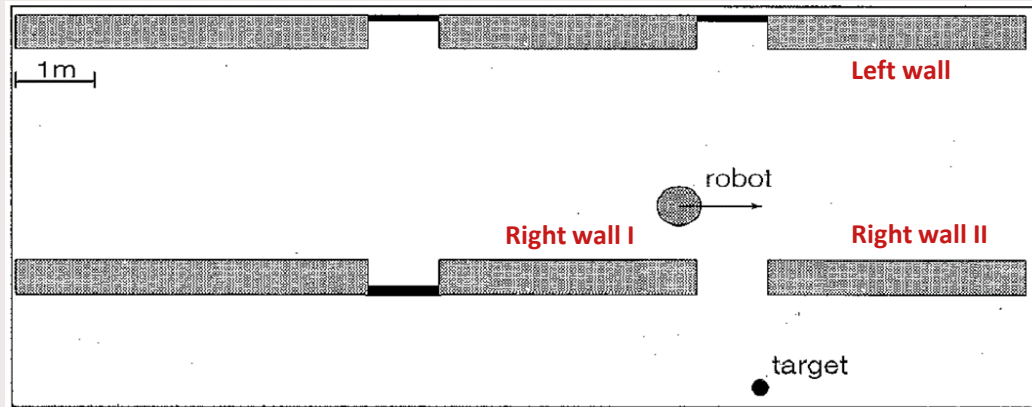
- Reactive collision avoidance based on robot dynamics
- Consider velocities  $(v, \omega)$  during  $t$ , where  $(v, \omega)$  have to be
  - Possible: velocities are limited by robot's dynamics
  - Admissible: robot can stop before reaching closest obstacle
  - Reachable: velocity and acceleration constraints (dynamic window)

$$V_d = \{v, \omega \mid v \in [v_a - \dot{v}t, v_a + \dot{v}t] \wedge \omega \in [\omega_a - \dot{\omega}t, \omega_a + \dot{\omega}t]\}$$

$v_a$  and  $\omega_a$  are actual velocities  
 $\dot{v}$  and  $\dot{\omega}$  are acceleration values



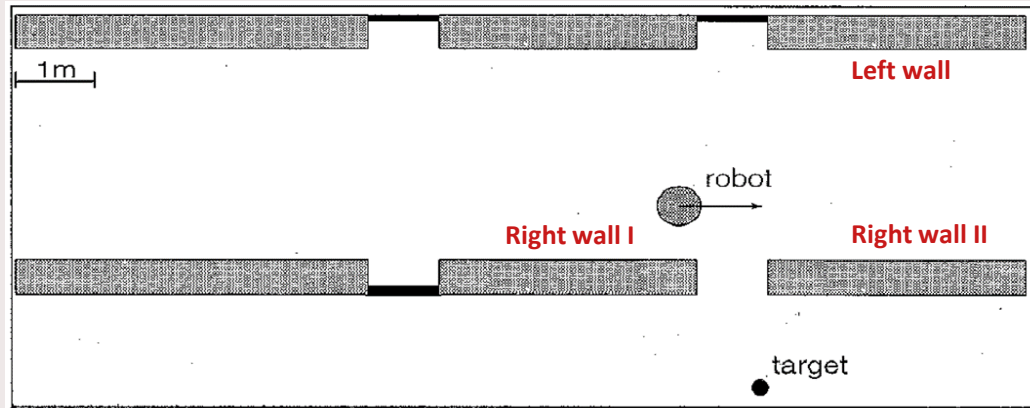
# Local navigation algorithms / dynamic window approach



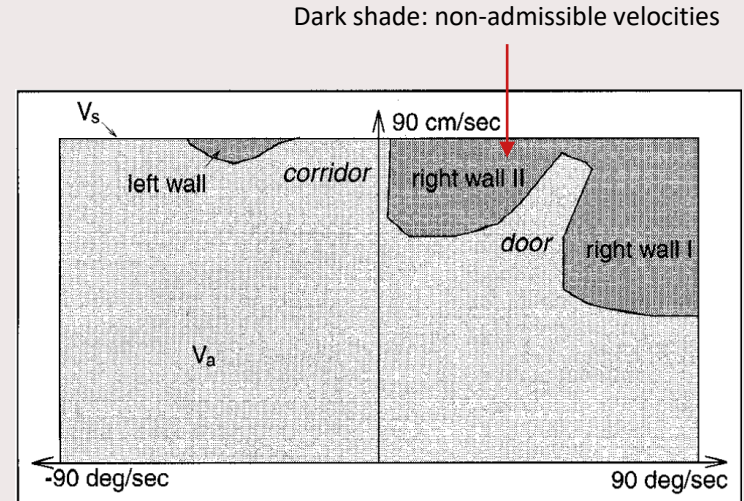
$V_s$ : possible velocities  
 $V_a$ : admissible velocities  
 $V_d$ : reachable velocities  
 $V_r$ : velocity search space

D. Fox, W. Burgard and S. Thrun, "The dynamic window approach to collision avoidance," in *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23-33, March 1997, doi: 10.1109/100.580977.

# Local navigation algorithms / dynamic window approach



- $V_s$ : possible velocities
- $V_a$ : admissible velocities
- $V_d$ : reachable velocities
- $V_r$ : velocity search space



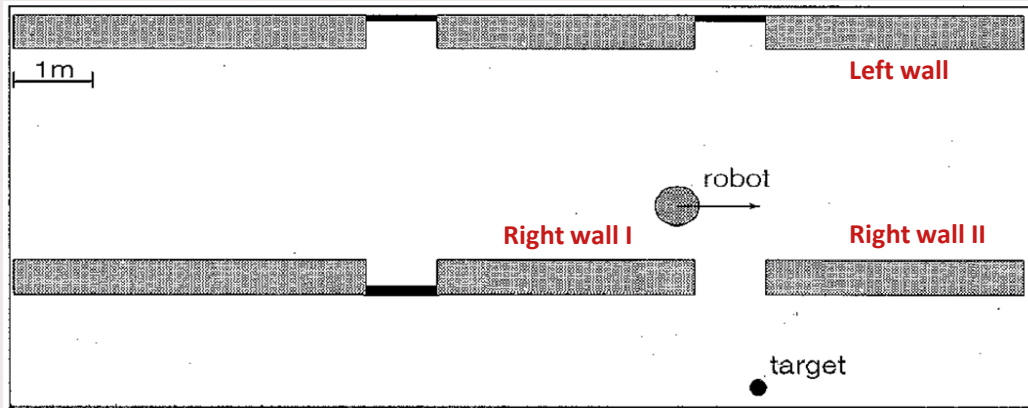
$$V_s = \{v, \omega | v \in [v_{min}, v_{max}] \wedge \omega \in [\omega_{min}, \omega_{max}]\}$$

$$V_a = \{v, \omega | v \leq \sqrt{2d(v, \omega)\dot{v}_b} \wedge \omega \leq \sqrt{2d(v, \omega)\dot{\omega}_b}\}$$

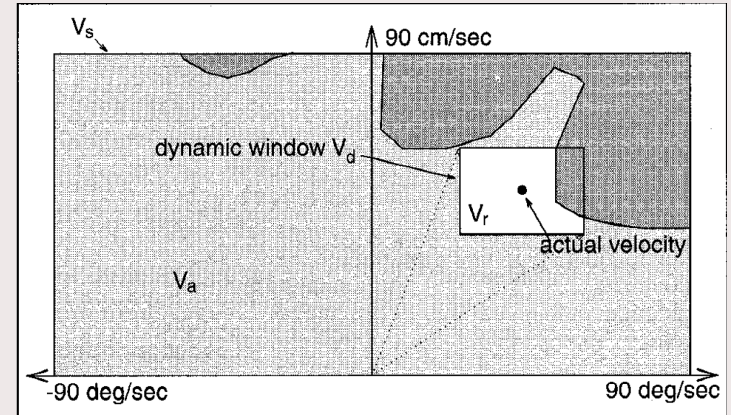
Here  $\dot{v}_b = 50 \text{ cm/s}^2$ ,  $\dot{\omega}_b = 60 \text{ deg/s}^2$

D. Fox, W. Burgard and S. Thrun, "The dynamic window approach to collision avoidance," in *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23-33, March 1997, doi: 10.1109/100.580977.

# Local navigation algorithms / dynamic window approach



- $V_s$ : possible velocities
- $V_a$ : admissible velocities
- $V_d$ : reachable velocities
- $V_r$ : velocity search space



$$V_d = \{v, \omega \mid v \in [v_a - \dot{v}\Delta t, v_a + \dot{v}\Delta t] \wedge \omega \in [\omega_a - \dot{\omega}\Delta t, \omega_a + \dot{\omega}\Delta t]\}$$

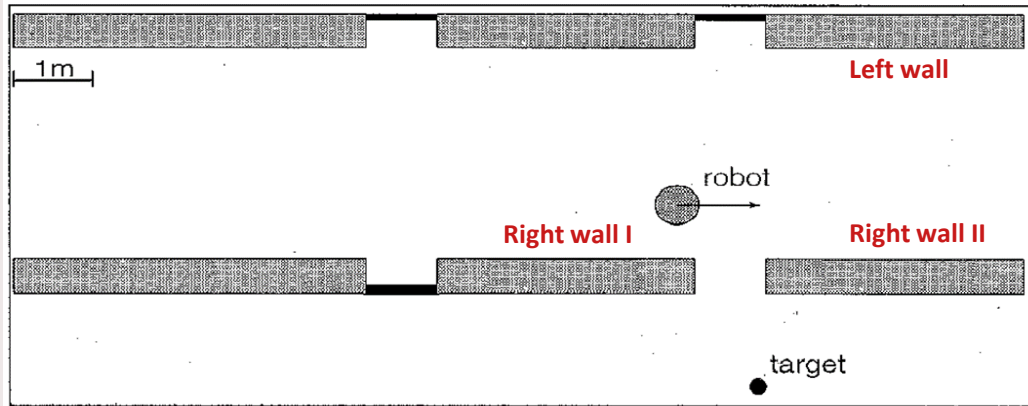
D. Fox, W. Burgard and S. Thrun, "The dynamic window approach to collision avoidance," in *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23-33, March 1997, doi: 10.1109/100.580977.

# Local navigation algorithms / dynamic window approach

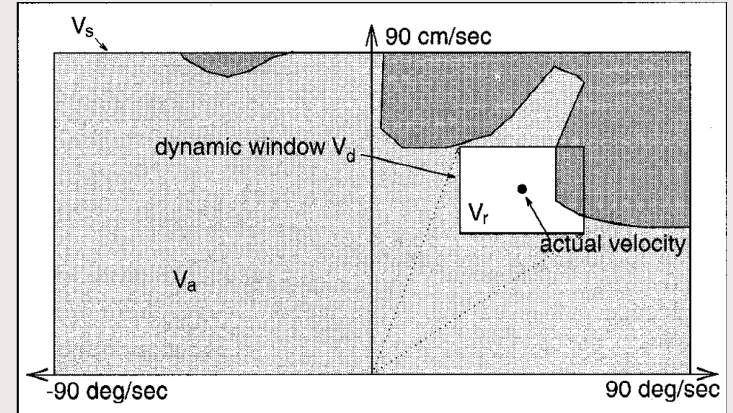
- Reactive collision avoidance based on robot dynamics
- Consider velocities  $(v, \omega)$  during  $t$ : possible, admissible, reachable
- Generate search space
  - Intersection of  $V_s, V_a$  and  $V_d$  provides search space  $V_r$ 
$$V_r = V_s \cap V_a \cap V_d$$
$$\rightarrow \text{gives } (v_{range}, \omega_{range}) \in V_r \text{ at each time step}$$

D. Fox, W. Burgard and S. Thrun, "The dynamic window approach to collision avoidance," in *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23-33, March 1997, doi: 10.1109/100.580977.

# Local navigation algorithms / dynamic window approach



- $V_s$ : possible velocities
- $V_a$ : admissible velocities
- $V_d$ : reachable velocities
- $V_r$ : velocity search space



$$\rightarrow V_r = V_s \cap V_a \cap V_d \text{ (white area)}$$

$$\rightarrow (v_{range}, \omega_{range}) \in V_r$$

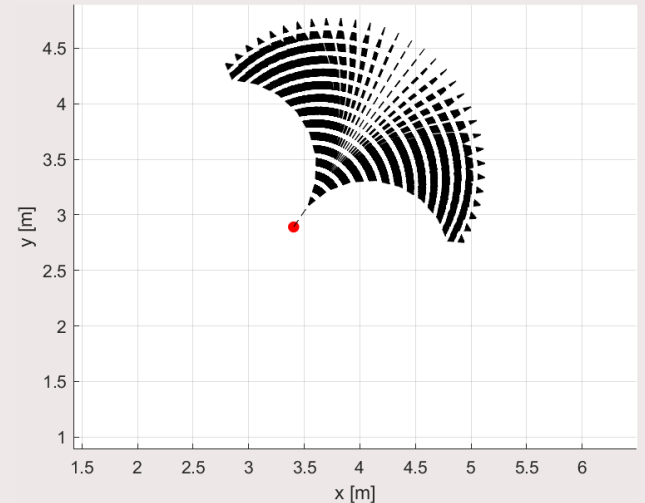
D. Fox, W. Burgard and S. Thrun, "The dynamic window approach to collision avoidance," in *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23-33, March 1997, doi: 10.1109/100.580977.

# Local navigation algorithms / dynamic window approach

- Reactive collision avoidance based on robot dynamics
- Consider velocities  $(v, \omega)$  during  $t$ : possible, admissible, reachable
- Generate search space

$x(0), y(0)$  and  $\theta(0)$  are current position

```
for  $j = 1:\text{len}(v_{\text{range}})$ 
  for  $k = 1:\text{len}(\omega_{\text{range}})$ 
    for  $i = 0:N$ 
       $x(i+1) = x(i) + \Delta t \cdot v_{\text{range}}(j) \cdot \cos(\theta(i))$ 
       $y(i+1) = y(i) + \Delta t \cdot v_{\text{range}}(j) \cdot \sin(\theta(i))$ 
       $\theta(i+1) = \theta(i) + \Delta t \cdot \omega_{\text{range}}(k)$ 
```



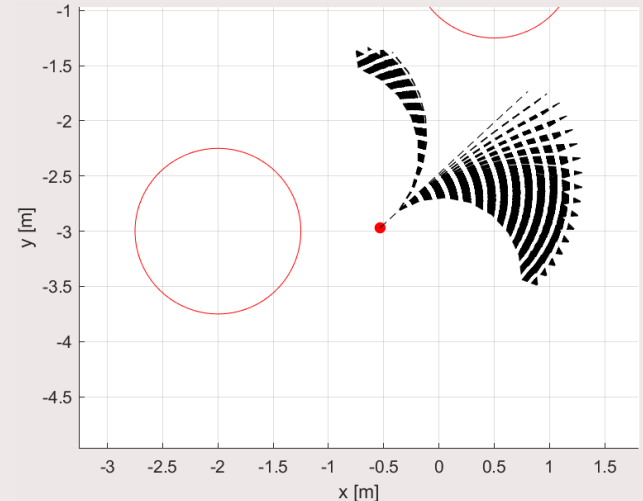
D. Fox, W. Burgard and S. Thrun, "The dynamic window approach to collision avoidance," in *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23-33, March 1997, doi: 10.1109/100.580977.

# Local navigation algorithms / dynamic window approach

- Reactive collision avoidance based on robot dynamics
- Consider velocities  $(v, \omega)$  during  $t$ : possible, admissible, reachable
- Generate search space

$x(0), y(0)$  and  $\theta(0)$  are current position

```
for  $j = 1:\text{len}(v_{\text{range}})$ 
  for  $k = 1:\text{len}(\omega_{\text{range}})$ 
    for  $i = 0:N$ 
       $x(i+1) = x(i) + \Delta t \cdot v_{\text{range}}(j) \cdot \cos(\theta(i))$ 
       $y(i+1) = y(i) + \Delta t \cdot v_{\text{range}}(j) \cdot \sin(\theta(i))$ 
       $\theta(i+1) = \theta(i) + \Delta t \cdot \omega_{\text{range}}(k)$ 
```



D. Fox, W. Burgard and S. Thrun, "The dynamic window approach to collision avoidance," in *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23-33, March 1997, doi: 10.1109/100.580977.

# Local navigation algorithms / dynamic window approach

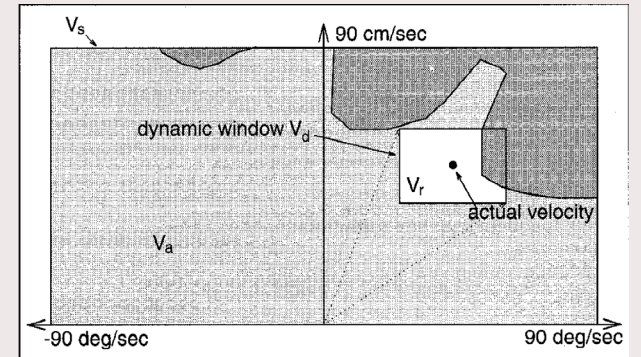
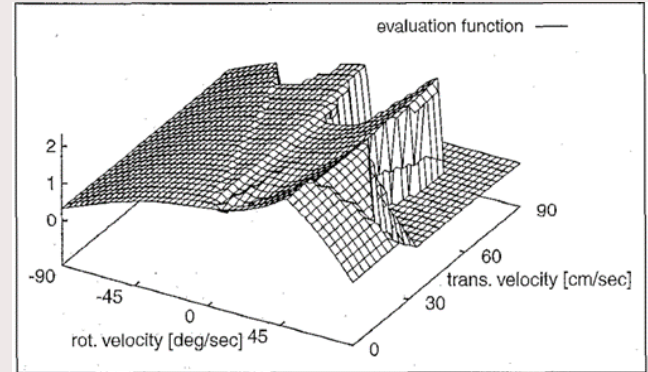
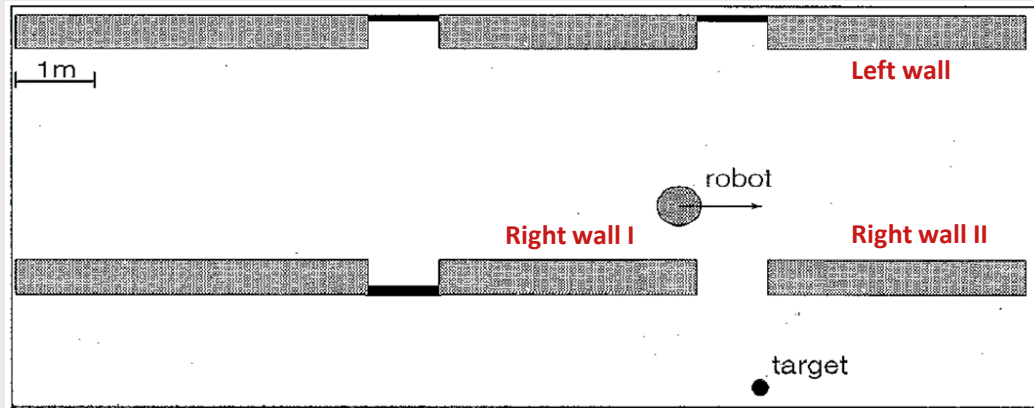
- Reactive collision avoidance based on robot dynamics
- Consider velocities  $(v, \omega)$  during  $t$ : possible, admissible, reachable
- Generate search space
- Maximize objective function  $G$

$$G(v, \omega) = \sigma(k_h h(v, \omega) + k_d d(v, \omega) + k_s s(v, \omega))$$

- $h(v, \omega)$ : target heading towards goal
- $d(v, \omega)$ : distance to closest obstacle on trajectory
- $s(v, \omega)$ : forward velocity

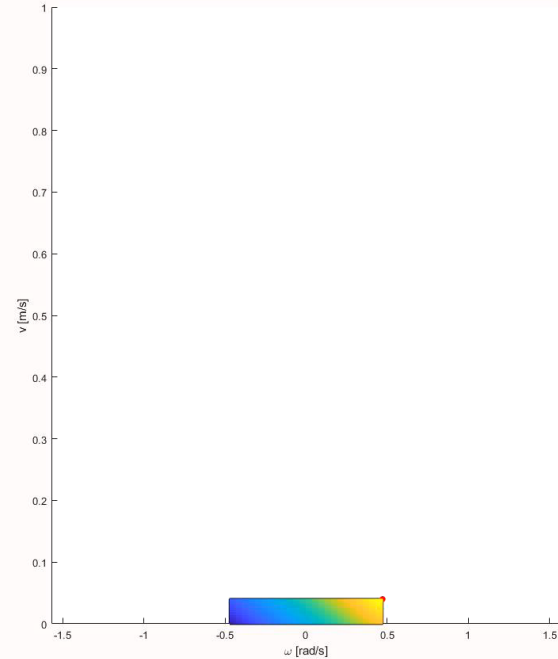
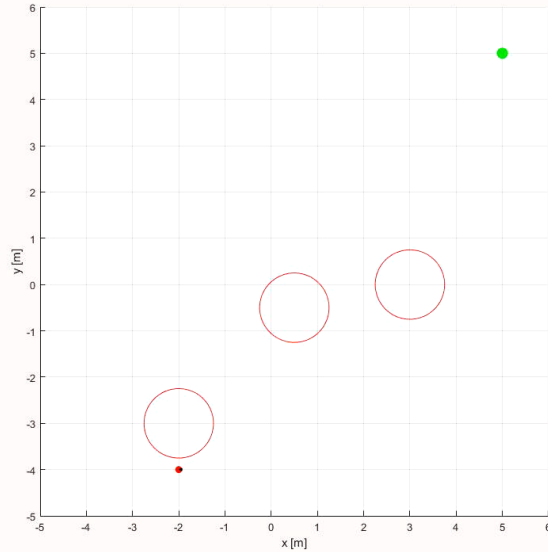


# Local navigation algorithms / dynamic window approach



D. Fox, W. Burgard and S. Thrun, "The dynamic window approach to collision avoidance," in *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23-33, March 1997, doi: 10.1109/100.580977.

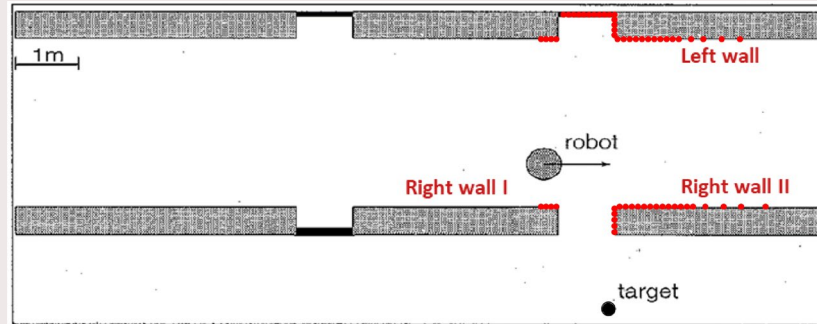
# Local navigation algorithms / dynamic window approach



D. Fox, W. Burgard and S. Thrun, "The dynamic window approach to collision avoidance," in *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23-33, March 1997, doi: 10.1109/100.580977.

# Local navigation algorithms / dynamic window approach

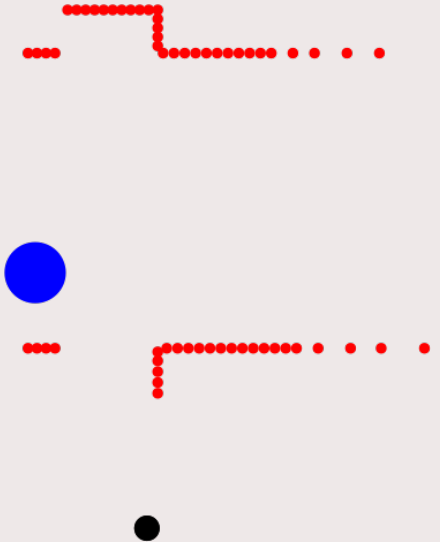
- Reactive collision avoidance based on robot dynamics
- Consider velocities  $(v, \omega)$  during  $t$ : possible, admissible, reachable
- Generate search space
- Maximize objective function  $G$
- Again, we have all information in simulation videos...



D. Fox, W. Burgard and S. Thrun, "The dynamic window approach to collision avoidance," in *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23-33, March 1997, doi: 10.1109/100.580977.

# Local navigation algorithms / dynamic window approach

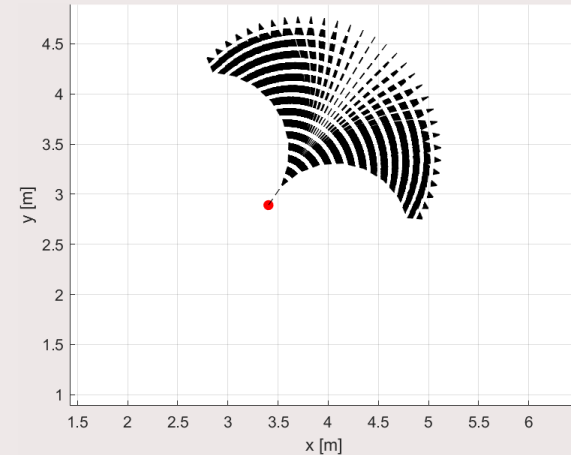
- How to represent the obstacles?
- Available information:
  - Laser range points
  - Trajectory from discretized velocities might fall between two points
- Also, incorporate the size of the robot
  - In the video, robot is a point mass



D. Fox, W. Burgard and S. Thrun, "The dynamic window approach to collision avoidance," in *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23-33, March 1997, doi: 10.1109/100.580977.

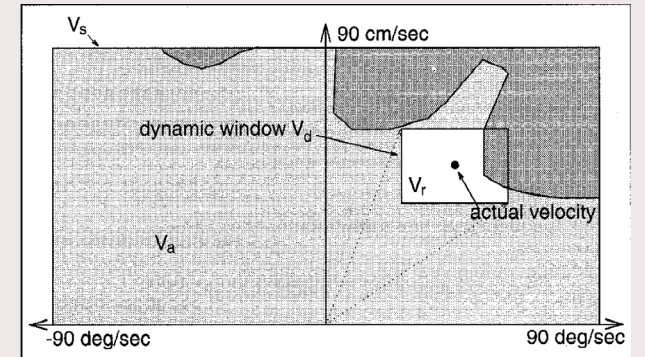
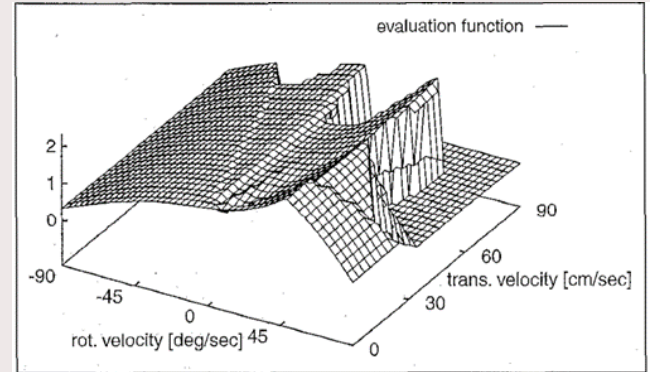
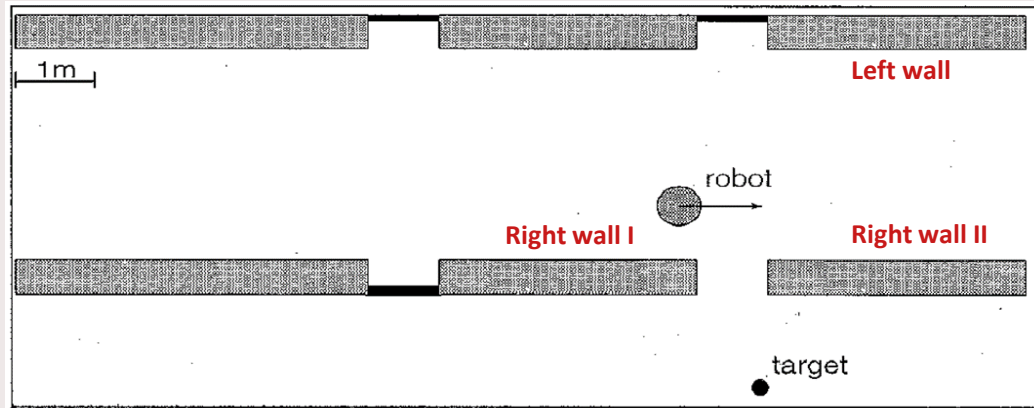
# Local navigation algorithms / dynamic window approach

- Reactive collision avoidance based on robot dynamics
- Consider velocities  $(v, \omega)$  during  $t$ : possible, admissible, reachable
- Generate search space
- Maximize objective function  $G$
- Again, we have all information in simulation videos...
- Implementation
  - How to check if a path is valid?
  - How discretize  $v_{range}$  and  $\omega_{range}$ ?
  - How to account for robot size?



D. Fox, W. Burgard and S. Thrun, "The dynamic window approach to collision avoidance," in *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23-33, March 1997, doi: 10.1109/100.580977.

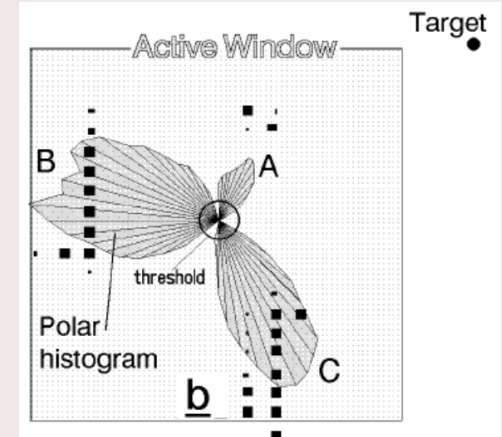
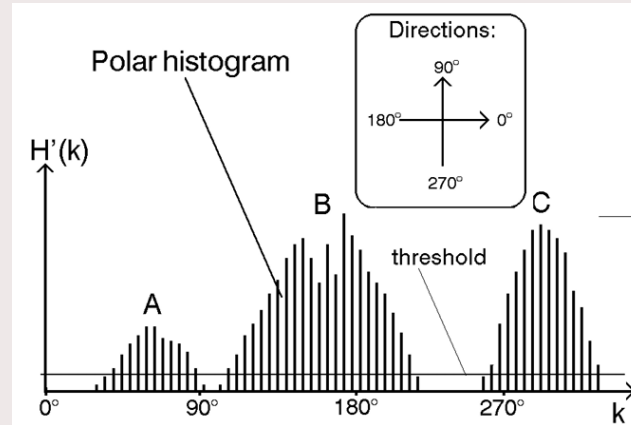
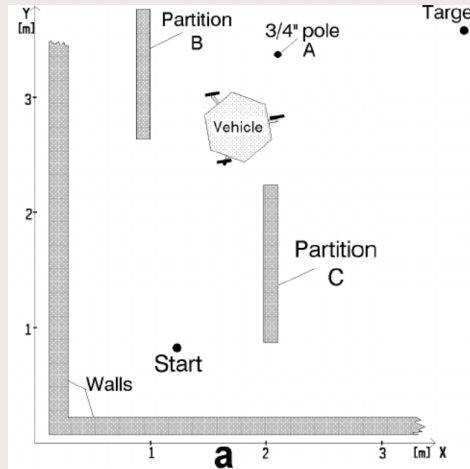
# Local navigation algorithms / dynamic window approach



Questions?

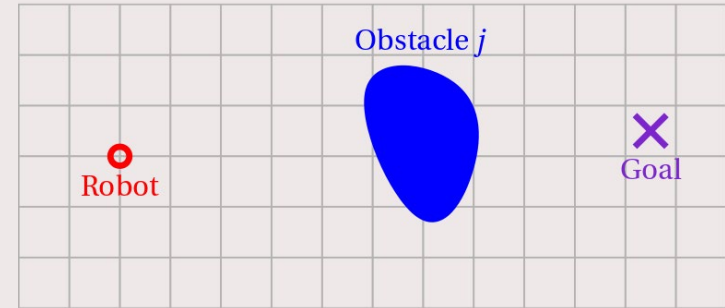
# Local navigation algorithms / vector field histograms

- Treat objects as vectors in a 2D Cartesian histogram grid, and create a polar histogram to determine possible 'open spaces' to get to the goal



# Local navigation algorithms / vector field histograms

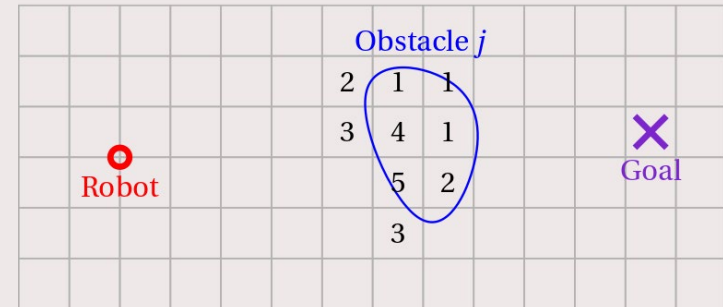
- 2D Cartesian histogram grid





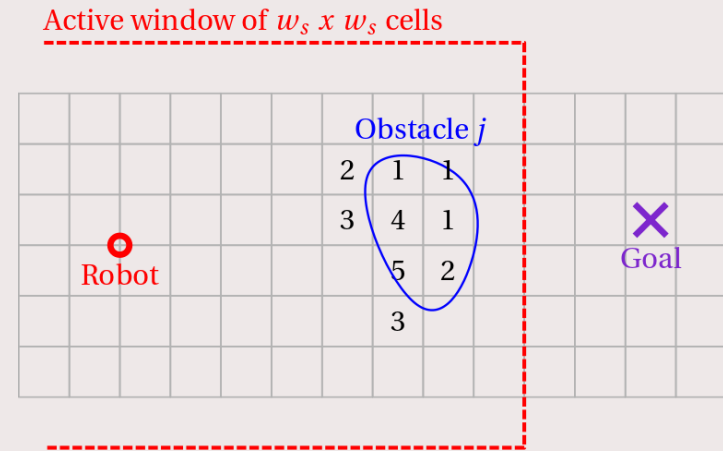
# Local navigation algorithms / vector field histograms

- 2D Cartesian histogram grid
  - each cell holds a certainty (or confidence) value  $c_{i,j}$  of that cell containing an obstacle



# Local navigation algorithms / vector field histograms

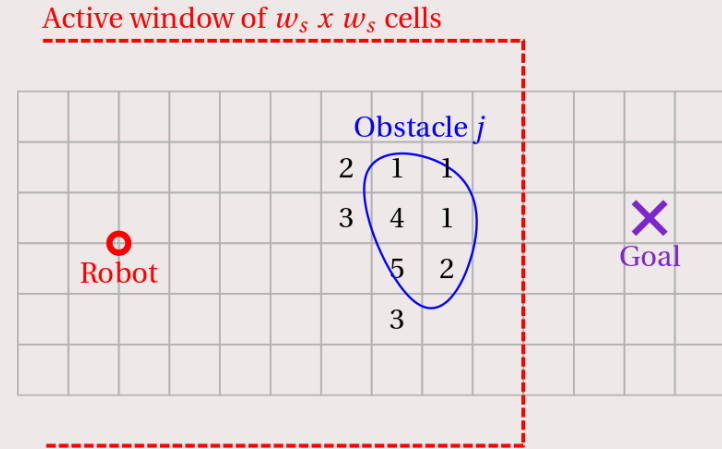
- 2D Cartesian histogram grid
  - each cell holds a certainty (or confidence) value  $c_{i,j}$  of that cell containing an obstacle
  - Active window



Note that the active window should be square and centered around robot, drawing is purely for visualization of the approach

# Local navigation algorithms / vector field histograms

- 2D Cartesian histogram grid
  - each cell holds a certainty (or confidence) value  $c_{i,j}$  of that cell containing an obstacle
  - Active window
  - Each active cell is treated as obstacle vector with
    - direction  $\beta_{i,j} = \text{atan2}(y_j - y_0, x_i - x_0)$
    - magnitude  $m_{i,j} = c_{i,j}^2(a - bd_{i,j})$ 
      - Choose  $a, b$  such that  $a - bd_{max} = 0$
      - $d_{max} = \frac{\sqrt{2}}{2}(w_s - 1)$
      - see [1] for further explanation on the values of  $a$  and  $b$



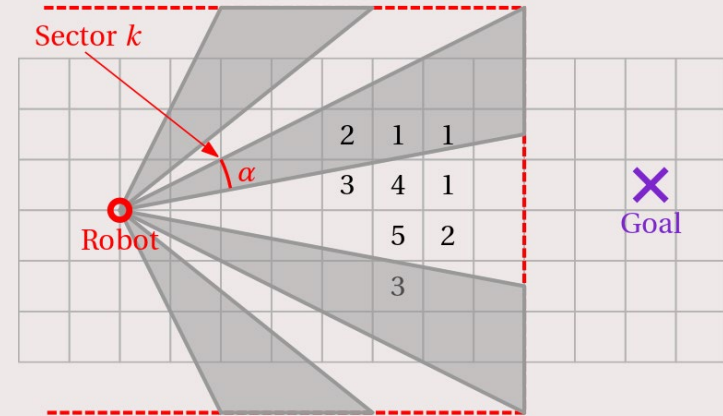
[1] J. Borenstein and Y. Koren, "The vector field histogram-fast obstacle avoidance for mobile robots," in IEEE Transactions on Robotics and Automation, vol. 7, no. 3, pp. 278-288, June 1991, doi: 10.1109/70.88137.

# Local navigation algorithms / vector field histograms

- 2D Cartesian histogram grid
- Polar histogram
  - Sector  $k$  corresponds to angular resolution  $\alpha$

$$\alpha = \frac{360^\circ}{n}$$

$n$  is an integer,  $k = 0, 1, 2, \dots, n - 1$

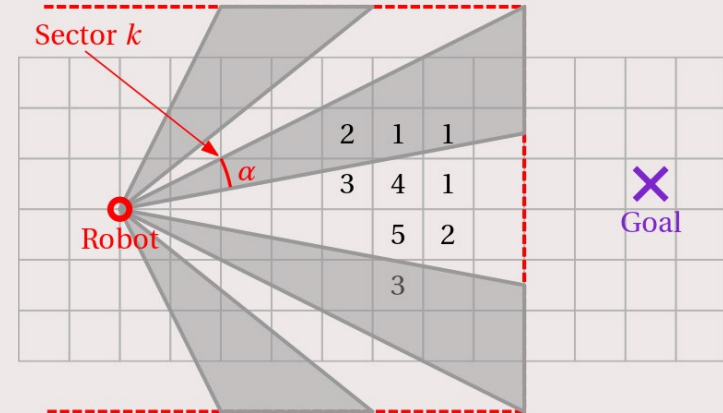


[1] J. Borenstein and Y. Koren, "The vector field histogram-fast obstacle avoidance for mobile robots," in IEEE Transactions on Robotics and Automation, vol. 7, no. 3, pp. 278-288, June 1991, doi: 10.1109/70.88137.

# Local navigation algorithms / vector field histograms

- 2D Cartesian histogram grid
- Polar histogram
  - Sector  $k$  corresponds to angular resolution  $\alpha$
  - Link between each cell  $c_{i,j}$  and  $k$

$$k = \text{int}\left(\frac{\beta_{i,j}}{\alpha}\right)$$



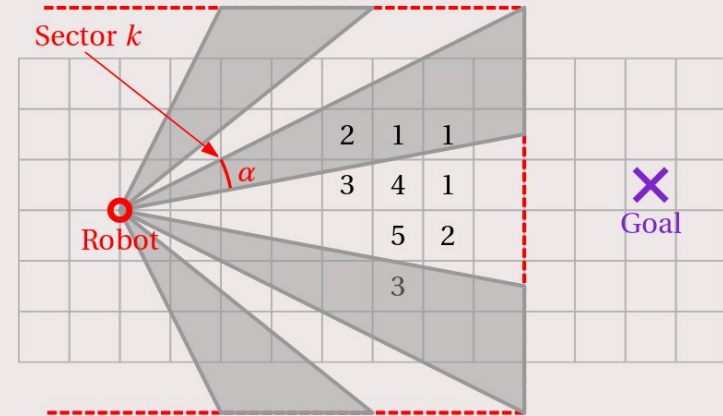
[1] J. Borenstein and Y. Koren, "The vector field histogram-fast obstacle avoidance for mobile robots," in IEEE Transactions on Robotics and Automation, vol. 7, no. 3, pp. 278-288, June 1991, doi: 10.1109/70.88137.

# Local navigation algorithms / vector field histograms

- 2D Cartesian histogram grid
- Polar histogram
  - Sector  $k$  corresponds to angular resolution  $\alpha$
  - Link between each cell  $c_{i,j}$  and  $k$
  - For each sector  $k$ , polar obstacle density  $h_k$  is

$$h_k = \sum_{i,j} m_{i,j}$$

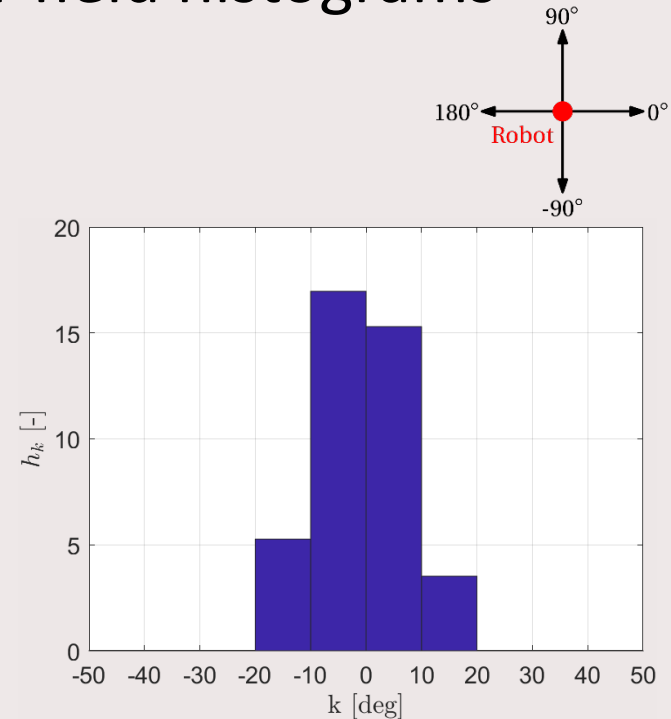
Note: needs smoothing due to discrete map, see [1]



[1] J. Borenstein and Y. Koren, "The vector field histogram-fast obstacle avoidance for mobile robots," in IEEE Transactions on Robotics and Automation, vol. 7, no. 3, pp. 278-288, June 1991, doi: 10.1109/70.88137.

# Local navigation algorithms / vector field histograms

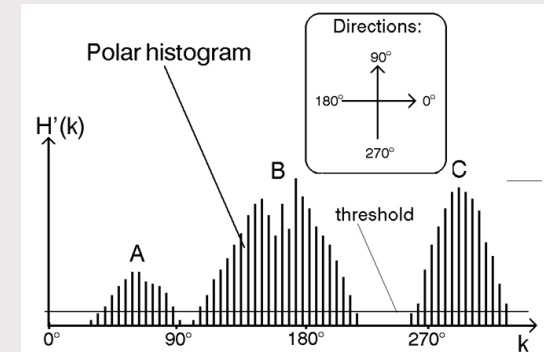
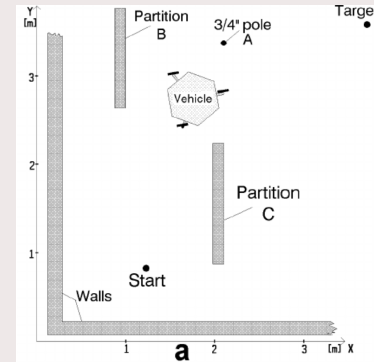
- 2D Cartesian histogram grid
- Polar histogram
  - Sector  $k$  corresponds to angular resolution  $\alpha$
  - Link between each cell  $c_{i,j}$  and  $k$
  - For each sector  $k$ , polar obstacle density  $h_k$
  - Resulting histogram
    - Note that the figure only shows  $[-50^\circ, 50^\circ]$ , but the histogram is actually  $[-180^\circ, 180^\circ]$
    - Note that no smoothing is applied



[1] J. Borenstein and Y. Koren, "The vector field histogram-fast obstacle avoidance for mobile robots," in IEEE Transactions on Robotics and Automation, vol. 7, no. 3, pp. 278-288, June 1991, doi: 10.1109/70.88137.

# Local navigation algorithms / vector field histograms

- 2D Cartesian histogram grid
- Polar histogram
- Steering direction
  - Smoothed polar histogram  $H'(k)$  [1]

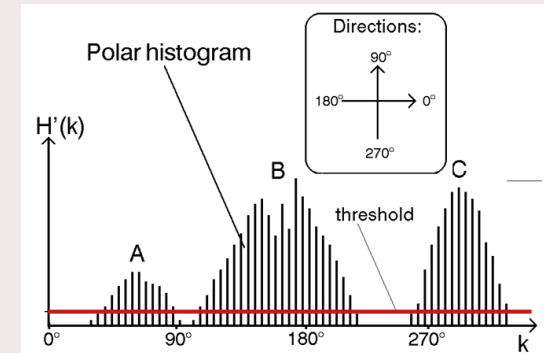
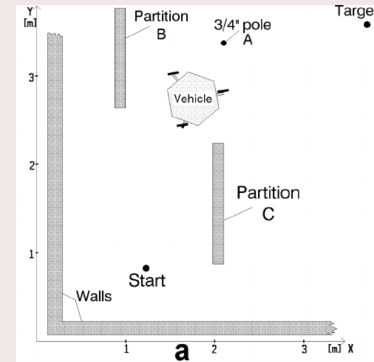


[1] J. Borenstein and Y. Koren, "The vector field histogram-fast obstacle avoidance for mobile robots," in IEEE Transactions on Robotics and Automation, vol. 7, no. 3, pp. 278-288, June 1991, doi: 10.1109/70.88137.



# Local navigation algorithms / vector field histograms

- 2D Cartesian histogram grid
- Polar histogram
- Steering direction
  - Smoothed polar histogram  $H'(k)$  [1]
  - Candidate valleys:  $H'(k)$  below **threshold**



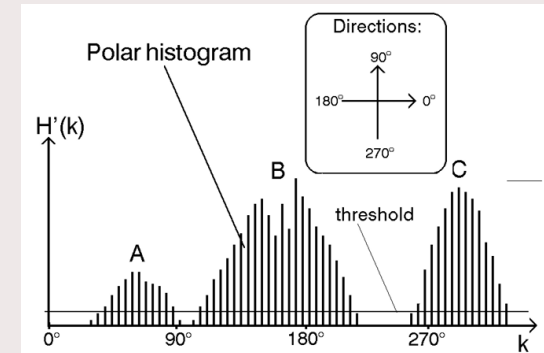
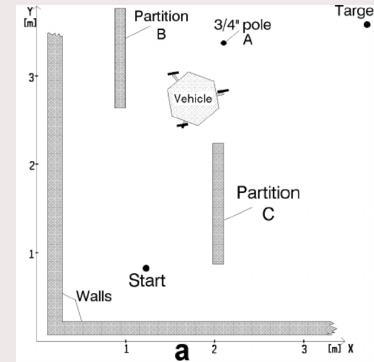
[1] J. Borenstein and Y. Koren, "The vector field histogram-fast obstacle avoidance for mobile robots," in IEEE Transactions on Robotics and Automation, vol. 7, no. 3, pp. 278-288, June 1991, doi: 10.1109/70.88137.

# Local navigation algorithms / vector field histograms

- 2D Cartesian histogram grid
- Polar histogram
- Steering direction
  - Smoothed polar histogram  $H'(k)$  [1]
  - Candidate valleys:  $H'(k)$  below threshold
  - Angle  $\theta$  is the middle of candidate valley

$$\theta = \frac{1}{2} \alpha(k_l + k_r)$$

$k_l$  and  $k_r$  are left and right boundary of selected valley



[1] J. Borenstein and Y. Koren, "The vector field histogram-fast obstacle avoidance for mobile robots," in IEEE Transactions on Robotics and Automation, vol. 7, no. 3, pp. 278-288, June 1991, doi: 10.1109/70.88137.

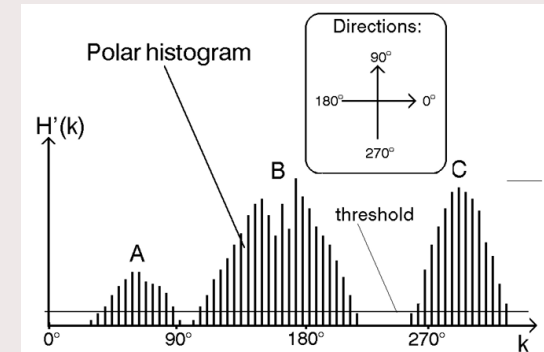
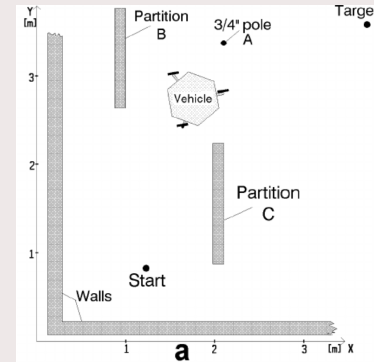
# Local navigation algorithms / vector field histograms

- 2D Cartesian histogram grid
- Polar histogram
- Steering direction
  - Smoothed polar histogram  $H'(k)$  [1]
  - Candidate valleys:  $H'(k)$  below threshold
  - Angle  $\theta$  is the middle of candidate valley

$$\theta = \frac{1}{2} \alpha(k_l + k_r)$$

$k_l$  and  $k_r$  are left and right boundary of selected valley

- Select the valley with closest match to goal direction



[1] J. Borenstein and Y. Koren, "The vector field histogram-fast obstacle avoidance for mobile robots," in IEEE Transactions on Robotics and Automation, vol. 7, no. 3, pp. 278-288, June 1991, doi: 10.1109/70.88137.

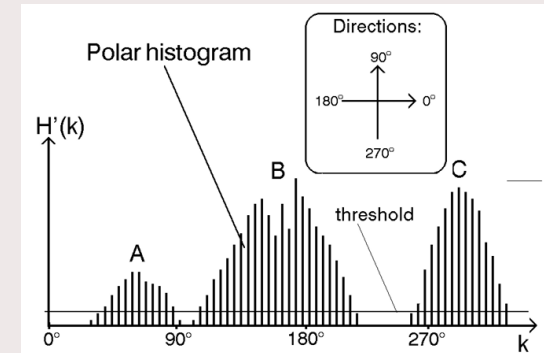
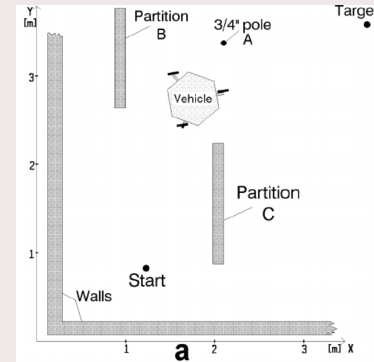
# Local navigation algorithms / vector field histograms

- 2D Cartesian histogram grid
- Polar histogram
- Steering direction
  - Smoothed polar histogram  $H'(k)$  [1]
  - Candidate valleys:  $H'(k)$  below threshold
  - Angle  $\theta$  is the middle of candidate valley

$$\theta = \frac{1}{2} \alpha(k_l + k_r)$$

$k_l$  and  $k_r$  are left and right boundary of selected valley

- Select the valley with closest match to goal direction
- Controller (e.g., PI) to align robot with goal direction



[1] J. Borenstein and Y. Koren, "The vector field histogram-fast obstacle avoidance for mobile robots," in IEEE Transactions on Robotics and Automation, vol. 7, no. 3, pp. 278-288, June 1991, doi: 10.1109/70.88137.

# Local navigation algorithms / vector field histograms

- 2D Cartesian histogram grid
- Polar histogram
- Steering direction
- Velocity control
- Anticipatory reduction:  $v' = V_{max} \left( 1 - \frac{1}{h_m} \min(h'_c, h_m) \right)$

$h'_c$ : obstacle density in current direction of travel

$h_m$ : empirically determined constant to obtain sufficient speed reduction

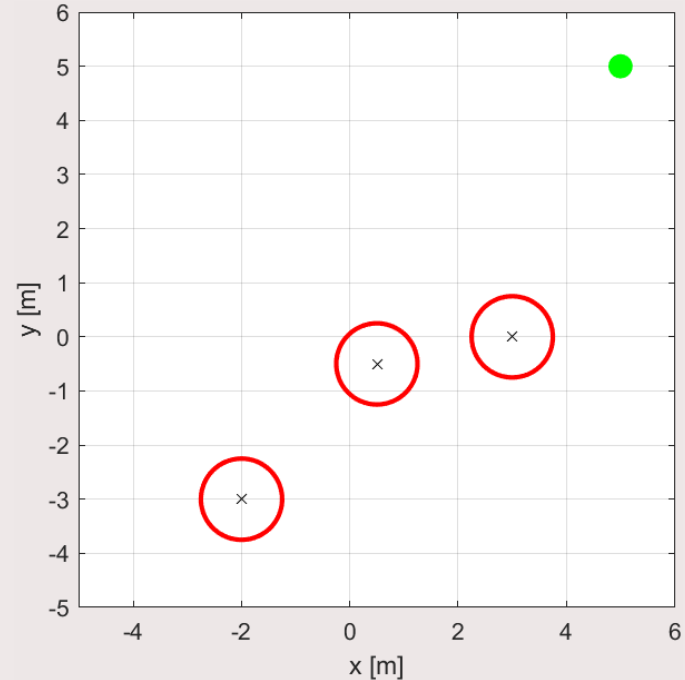
# Local navigation algorithms / vector field histograms

- 2D Cartesian histogram grid
- Polar histogram
- Steering direction
- Velocity control
  - Anticipatory reduction:  $v' = V_{max} \left( 1 - \frac{1}{h_m} \min(h'_c, h_m) \right)$
  - Steering speed reduction:  $v = v' \left( 1 - \frac{\dot{\theta}}{\dot{\theta}_{max}} \right) + V_{min}$

$h'_c$ : obstacle density in current direction of travel  
 $h_m$ : empirically determined constant to obtain sufficient speed reduction  
 $\dot{\theta}$ : steering rate

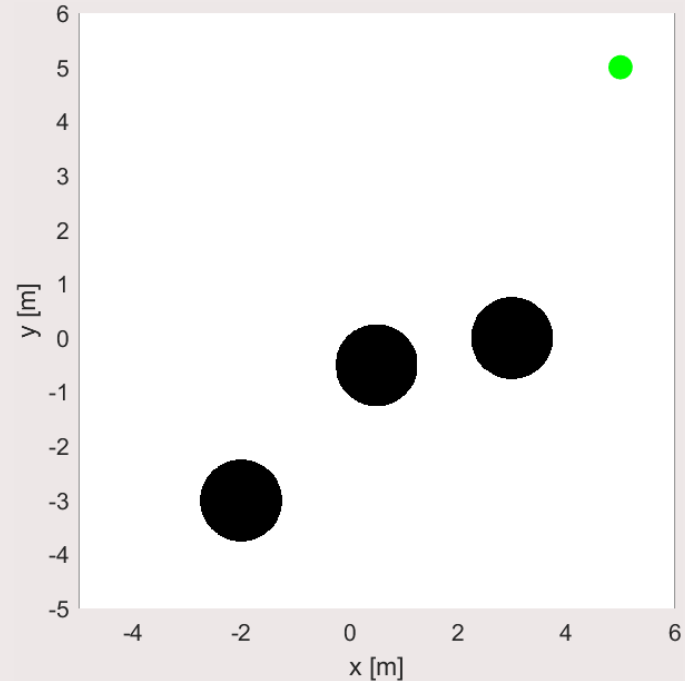
# Local navigation algorithms / vector field histograms

- 2D Cartesian histogram grid
- Polar histogram
- Steering direction
- Velocity control
- Example
  - Grid world map to create histogram grid



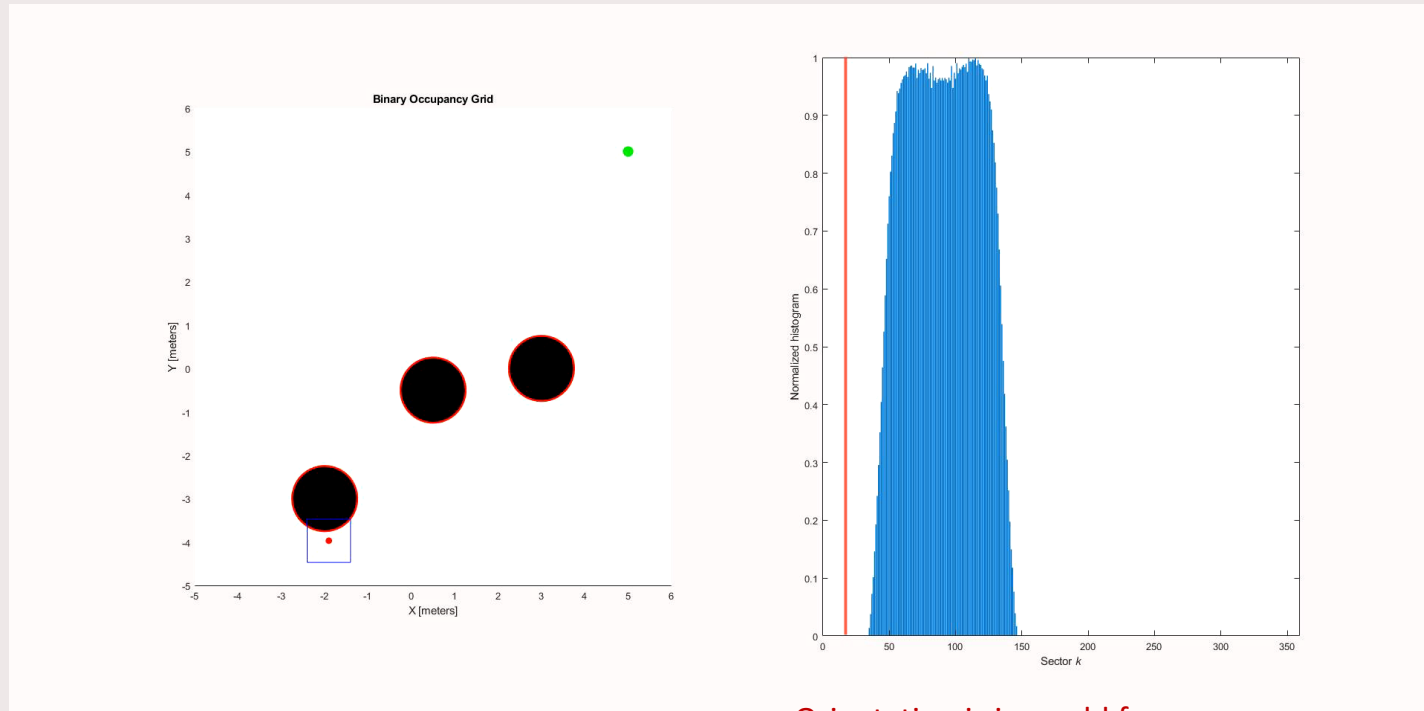
# Local navigation algorithms / vector field histograms

- 2D Cartesian histogram grid
- Polar histogram
- Steering direction
- Velocity control
- Example
  - Grid world map to create histogram grid
  - Assumed that obstacle position is fully known





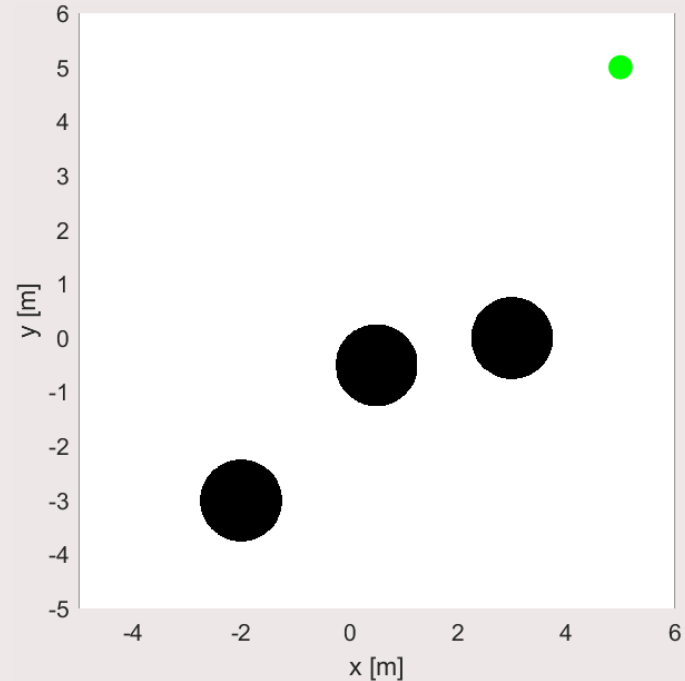
# Local navigation algorithms / vector field histograms



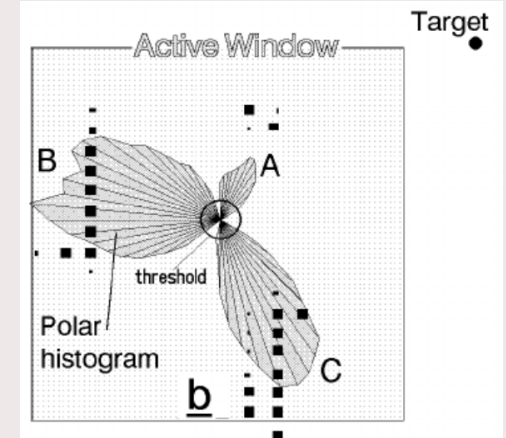
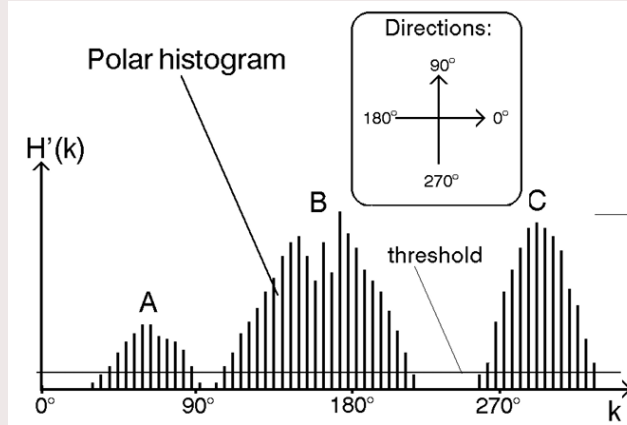
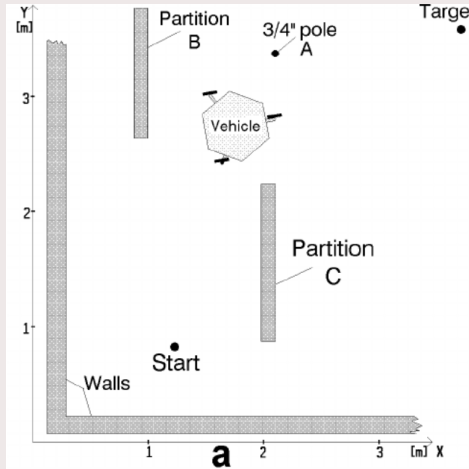
Orientation is in world frame

# Local navigation algorithms / vector field histograms

- 2D Cartesian histogram grid
- Polar histogram
- Steering direction
- Velocity control
- Implementation considerations
  - Again, think about the size of the robot
  - How to create the Cartesian histogram grid from sensor data?
  - What is the desired angle if there are no obstacles in the active window?



# Local navigation algorithms / vector field histograms



Questions?

# Local navigation algorithms / comparison of discussed approaches

- Artificial Potential Fields
  - Repulsion from objects and attraction to goal
  - Simple and computationally efficient
  - Suffers from local minimal and not optimal paths

# Local navigation algorithms / comparison of discussed approaches

- Artificial Potential Fields
  - Repulsion from objects and attraction to goal
  - Simple and computationally efficient
  - Suffers from local minimal and not optimal paths
- Dynamic Window Approach
  - Generate feasible action space based on robot dynamics within time horizon
  - Considers robot dynamics → collision-free and feasible trajectories
  - Requires accurate sensor data, might struggle with densely-populated environments

# Local navigation algorithms / comparison of discussed approaches

- Artificial Potential Fields
  - Repulsion from objects and attraction to goal
  - Simple and computationally efficient
  - Suffers from local minimal and not optimal paths
- Dynamic Window Approach
  - Generate feasible action space based on robot dynamics within time horizon
  - Considers robot dynamics → collision-free and feasible trajectories
  - Requires accurate sensor data, might struggle with densely-populated environments
- Vector Field Histograms
  - Create polar histogram of confidence on object location
  - Computationally efficient, robust to noisy sensor data
  - Can struggle with narrow passages and sharp corners

# Local navigation algorithms / other possible approaches

- Optimization based
  - Minimize objective function limited by constraints and system dynamics to find the 'optimal' path or trajectory
  - Objective function:
    - Distance/time to goal,
    - Smoothness of trajectory,
    - Comfort (acceleration/jerk),
    - Safety related.

$$\begin{aligned} & \min \int_0^T J(x(t), u(t)) \\ & \text{subject to} \\ & x(0) = x_0 \\ & \dot{x}(t) = f(x(t), u(t)) \\ & g(x(t), u(t)) \leq 0 \\ & \underline{u} \leq u(t) \leq \bar{u} \\ & \underline{x} \leq x(t) \leq \bar{x} \end{aligned}$$

# Local navigation algorithms / other possible approaches

- Optimization based
- Learning based
  - Relies heavily on training sensor data,
  - Train a learning model (e.g., neural network) to
    - Predict behaviour of environment
    - Detect obstacles
    - Decision-making
  - Based on real-life sensor data, create necessary output





<https://www.youtube.com/watch?v=FwT4TSRsiVw>

# Local navigation algorithms / other possible approaches

- Optimization based
- Learning based
- **Note:**
  - We have explained three approaches from a wide range of possibilities
  - In the exercises, you are allowed to implement approaches not treated in this lecture
  - But note that more complex is not necessarily better..
  - Additionally, note that the explained algorithms directly provide control outputs

# Footnote: world representation

- All sensor info treated the same
- In more complex environments different objects should be treated differently based on their semantic context
  - E.g., keep more distance to humans.

# Recap

- What is the robot navigation problem?
  - Find a feasible path or trajectory from a given initial pose (A) to the desired final pose (B)
- What is the goal of local navigation?
  - Go from A to B using the global path as a guide
- Local navigation algorithms: properties
- Local navigation algorithms: examples
  - Artificial potential fields
  - Dynamic window approach
  - Vector field histogram
  - Optimization and learning based methods

# Assignment

- Divide your group into two (equal sized) groups
- Enable your robot to drive through a corridor to a goal position by implementing **two different local navigation algorithms** (one by each subgroup)
- Answer the provided questions, provide videos of simulations and testing on the field, and upload your code (with comments!)
- Final remark:
  - You will use one of the algorithms in the final challenge
  - Create a **function for each algorithm** (which use the same input + output) to enable easy implementation and testing

# Literature

S. M. LaValle, "Planning Algorithms," Cambridge University Press, Cambridge, 2006, doi: 10.1017/CBO9780511546877.

B. Siciliano and O. Khatib, Eds., "Springer Handbook of Robotics," Springer, Berlin, Heidelberg, 2008, ISBN: 978-3-540-23957-4.

B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, "Robotics: Modelling, Planning and Control," Springer Publishing Company, Incorporated, 2010

D. Fox, W. Burgard and S. Thrun, "The dynamic window approach to collision avoidance," in *IEEE Robotics & Automation Magazine*, vol. 4, no. 1, pp. 23-33, March 1997, doi: 10.1109/100.580977.

J. Borenstein and Y. Koren, "The vector field histogram-fast obstacle avoidance for mobile robots," in *IEEE Transactions on Robotics and Automation*, vol. 7, no. 3, pp. 278-288, June 1991, doi: 10.1109/70.88137.