SPRING

Winter School, Grenoble
Feb 19th-23rd, 2024
Social Robotics Artificial Intelligence Multimedia

Human-aware Navigation

Model Predictive Control &
Deep Reinforcement Learning

Chris Reinke

(Alex Auternaud, Timothee Wintz, Victor Sanchez)

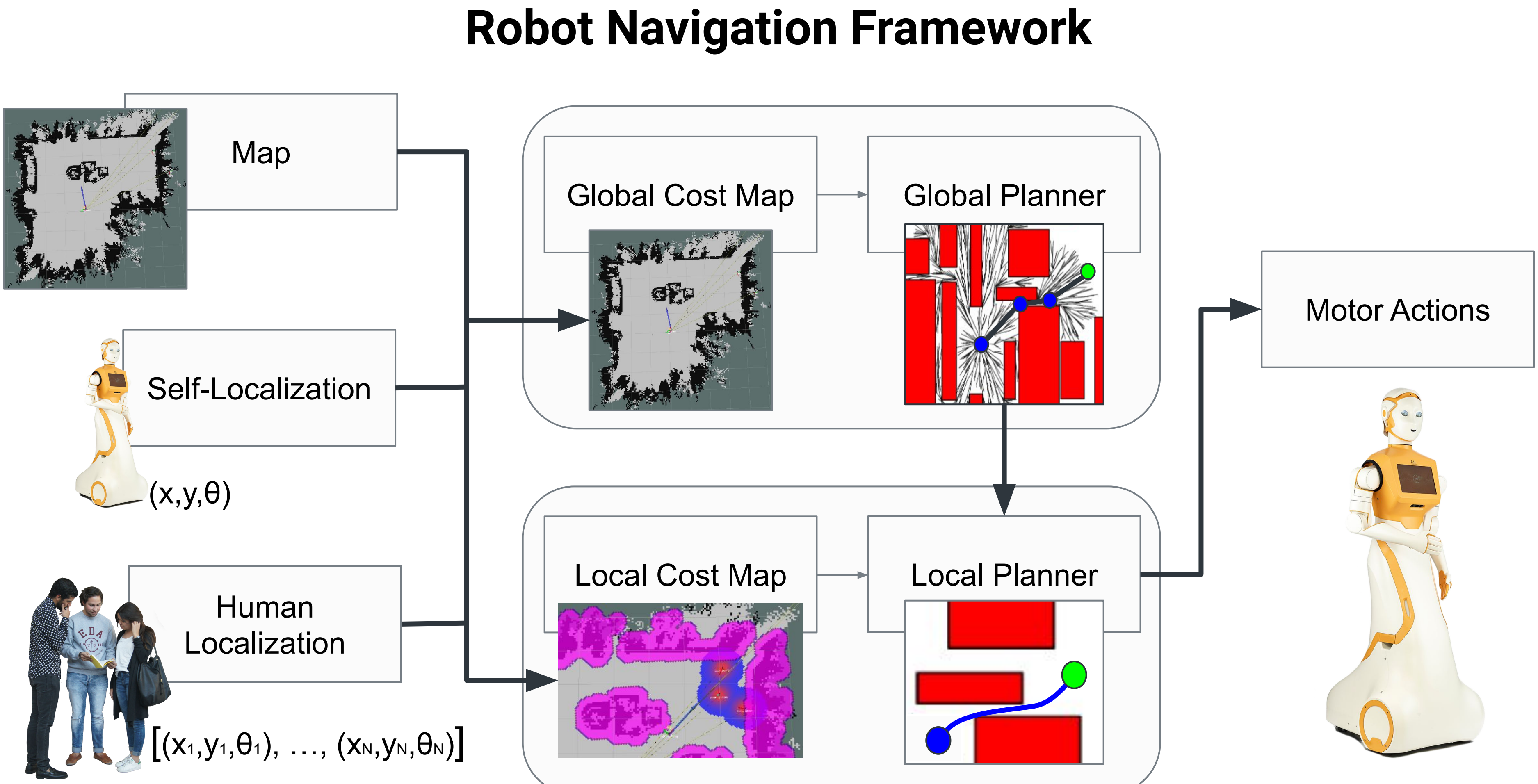
2024

inria informatics mathematics

ROBOT LEARN

Human-aware Navigation

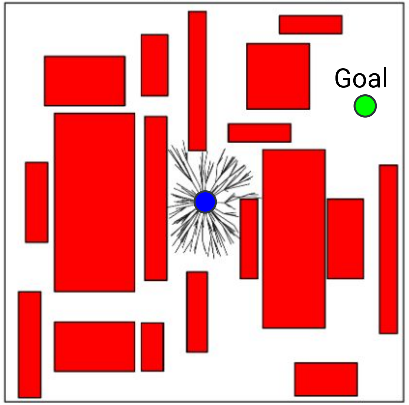
- Tasks
  - Navigate to a goal position, taking humans / groups into account
  - Join humans /groups to start interactions
  - Follow and guide humans / groups


Robot Navigation Framework
Map
Self-Localization
(x,y,θ)
Human Localization
[(x1,y1,θ1), …, (xN,yN,θN)]
Global Cost Map
Global Planner
Local Cost Map
Local Planner
Motor Actions

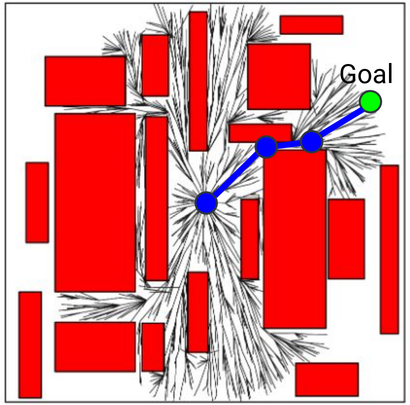
Global Planner

- Generating a high level navigation plan
- Create a series of waypoints for the local planner to achieve
- Usually has simplifications, for example:
  - Only basic shape for robot (circle or ellipse)
  - No dynamics model of robot
- Example: Fast Marching Tree (FMT)

FMT* Tree, First 100 Edges

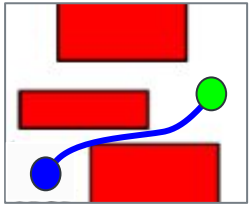


FMT* Tree, First 1000 Edges

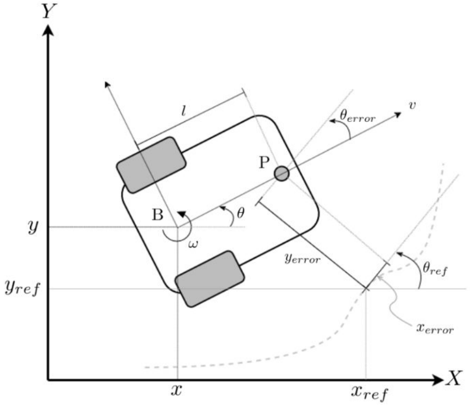


Janson et al. (2015)

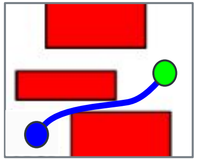
Local Planner



- Navigates to waypoints defined by the global planner
- Takes into account a realistic model of robot and environment
- Examples of possible solutions:
    - Planning: Model Predictive Control (MPC)
    - Learning: Deep Reinforcement Learning (DRL)



Model Predictive Control



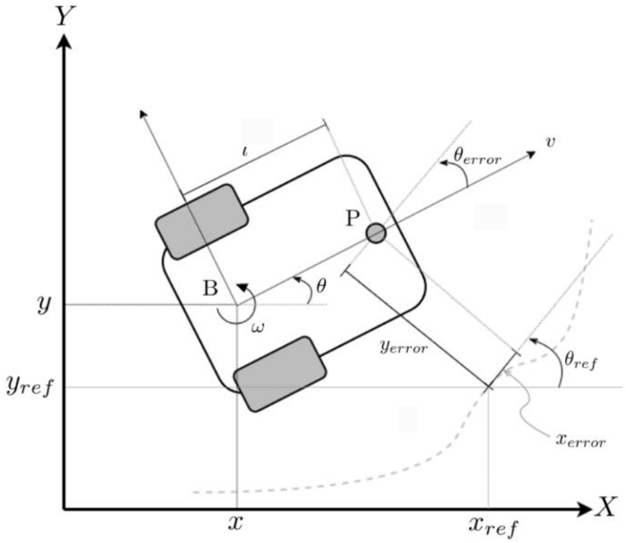
- Objective: $\boldsymbol{U}^* = \arg\min_{\boldsymbol{U}} \mathcal{J}(\boldsymbol{U}; \boldsymbol{s}(0), \boldsymbol{s}^*)$

- Objective function depends on actions, start state, and goal state:

$$\mathcal{J}(\boldsymbol{U}; \boldsymbol{s}(0), \boldsymbol{s}^*) = \sum_{t=1}^{T} \mathcal{L}(\boldsymbol{s}(t), \boldsymbol{s}^*) + \mathcal{R}(\boldsymbol{u}(t-1))$$

$$\boldsymbol{s}(t) = (x(t), y(t), \theta(t), \boldsymbol{\alpha}(t)) \qquad \boldsymbol{u}(t) = (v, \omega, \dot{\boldsymbol{\alpha}}(t))$$

- Optimize with respect to a
  - System model: $\boldsymbol{s}(t+1) = f(\boldsymbol{s}(t), \boldsymbol{u}(t))$
  - Constraints: $g(\boldsymbol{s}(0), \boldsymbol{u}(0), \boldsymbol{s}(1), \boldsymbol{u}(1), \ldots) \geq \boldsymbol{0}$



$$f(\boldsymbol{s}(t), \boldsymbol{u}(t))$$

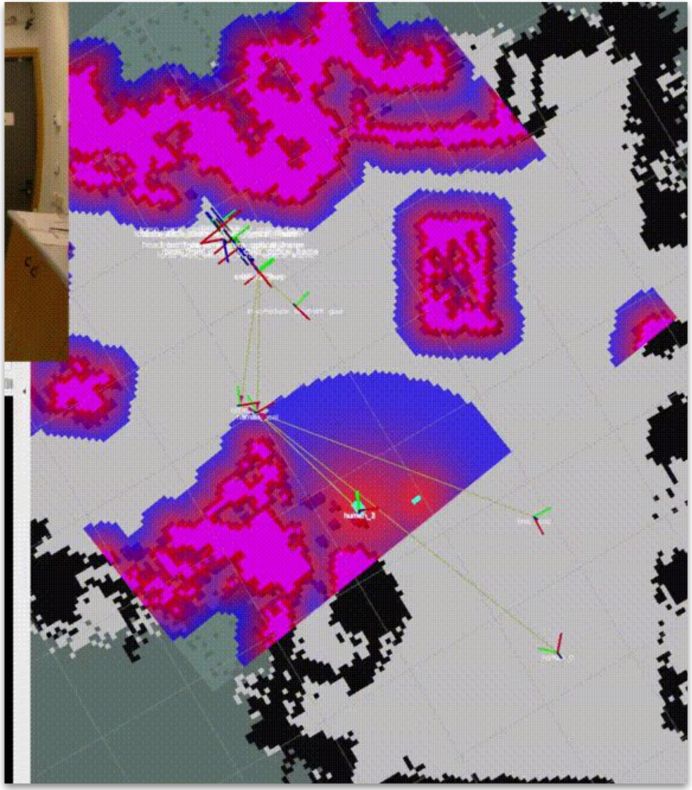
$$\begin{pmatrix} x(t+1) \\ y(t+1) \\ \theta(t+1) \\ \boldsymbol{\alpha}(t+1) \end{pmatrix} = \begin{pmatrix} x(t) - \Delta_t \sin(\theta(t))v \\ y(t) + \Delta_t \cos(\theta(t))v \\ \theta(t) + \Delta_t \omega \\ \boldsymbol{\alpha}(t) + \Delta_t \dot{\boldsymbol{\alpha}}(t) \end{pmatrix}$$

Camacho & Bordons (2007)

Cost Maps

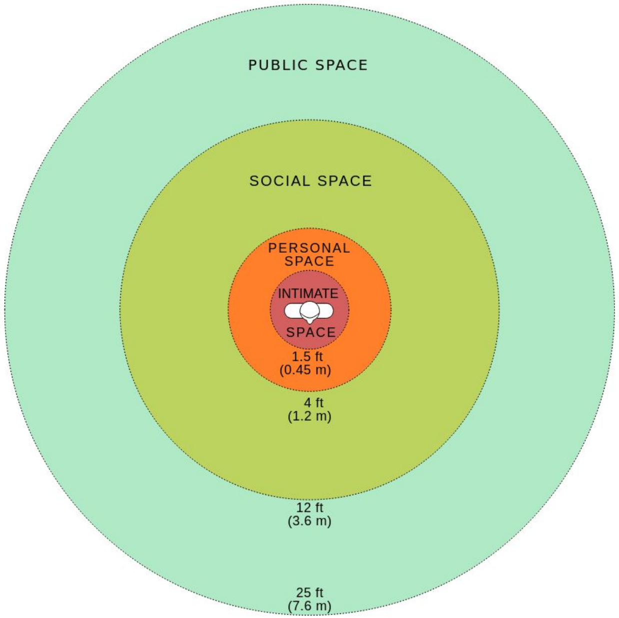
- Can combine different costs, for example:
  - Collisions
  - Distance to goal point
  - Energy usage

- In Social Robotics, it has to take into account humans and groups !

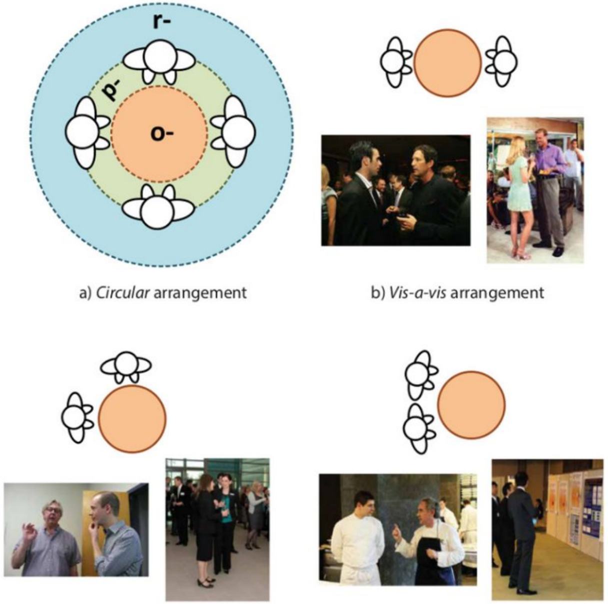


Social Spaces of Humans and Groups

Humans: Proxemics



Groups: F-formations



a) *Circular* arrangement

b) *Vis-a-vis* arrangement

c) *L*-arrangement

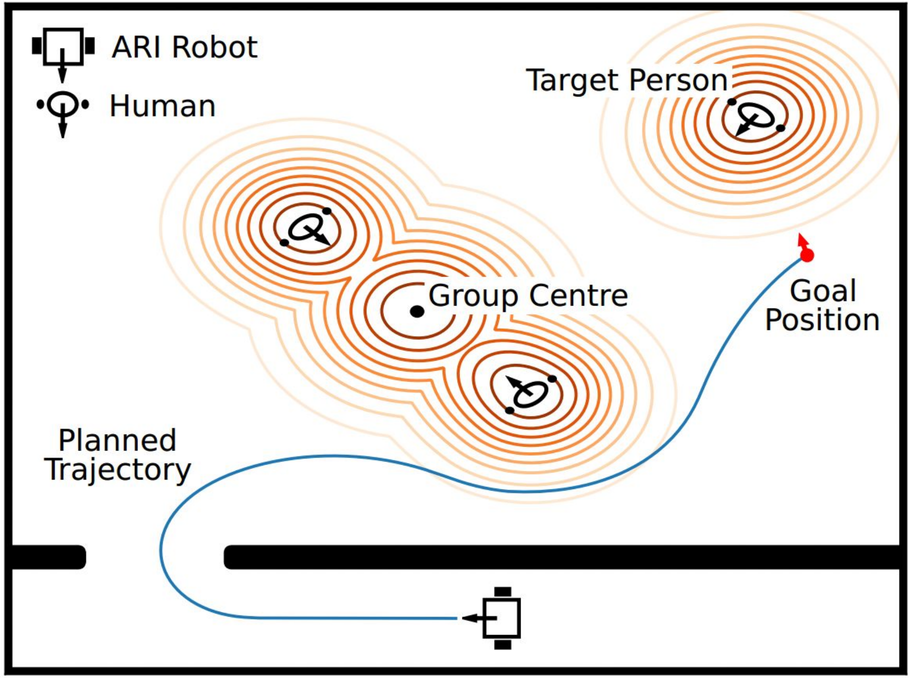
d) *Side-by-side* arrangement

Hall (1968)

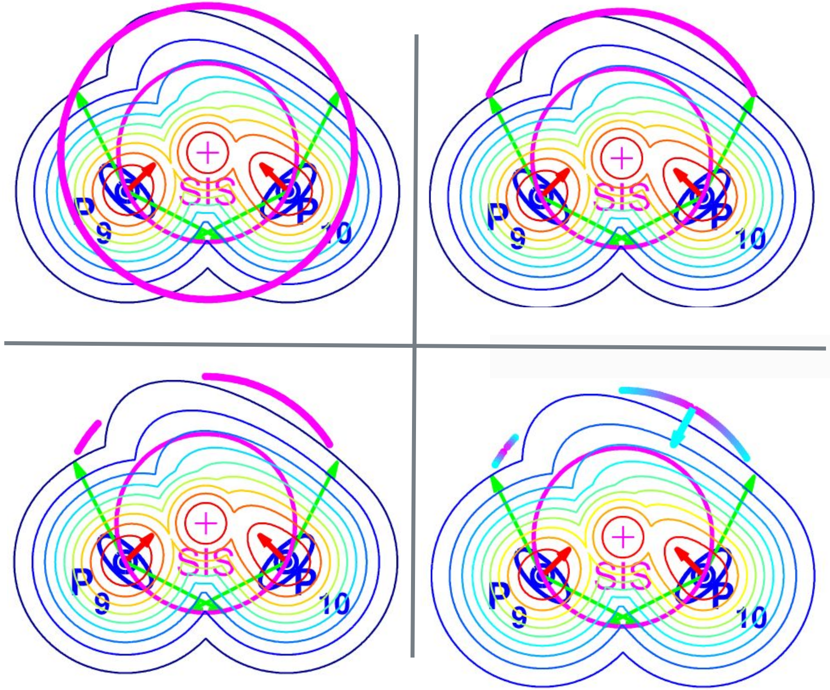
Kendon (1990)

Social Model

Model spaces with Gaussian like functions

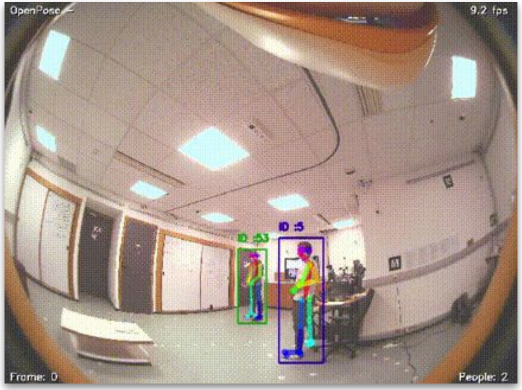


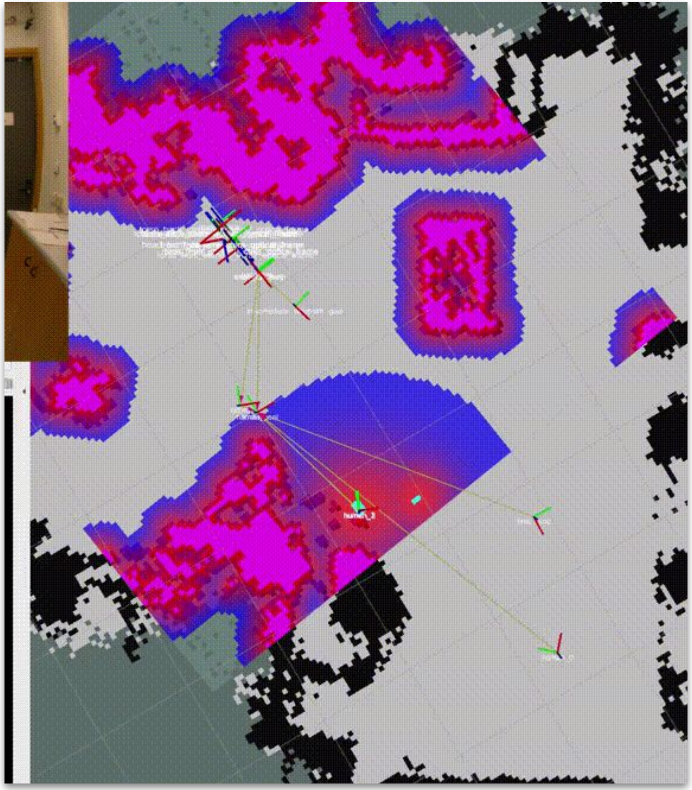
Identification of approach point

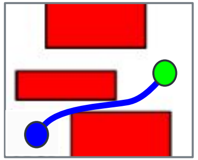


Truong et al. (2017)

Social MPC

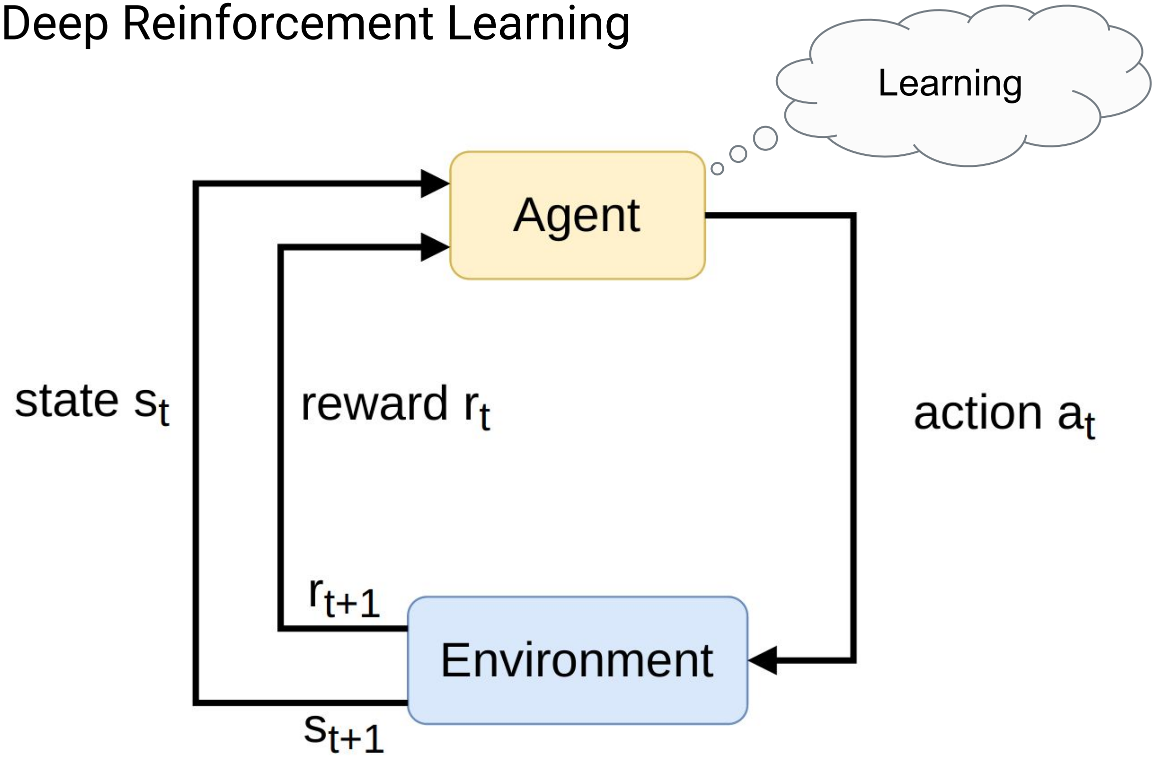






Local Planner

- Alternative: Deep Reinforcement Learning



Reinforcement Learning

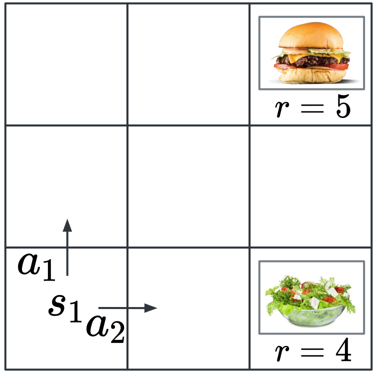
- Learning to solve multi-step decision problems

- Tasks are modeled as MDPs:
  - Set of states: S
  - Set of actions: A
  - Transition function: $\Pr(s_{t+1} | s_t, a_t)$
  - Reward function: $R(s_t)$

- Goal: Maximization of return $E\left[\sum_{t=0}^{T=\infty} \gamma^t R(s_t)\right]$

- Policy: $a_t = \pi(s_t)$

Grid-world example



Barto & Sutton (2018)

Reinforcement Learning

- Q-function (model-free, value-based):

$$Q^\pi(s_t, a_t) = E\left[r_t + \gamma^1 r_{t+1} + \gamma^2 r_{t+2} + \ldots\right]$$

$$Q^\pi(s_t, a_t) = E\left[r_t + \gamma Q^\pi(s_{t+1}, a_{t+1})\right]$$

$$Q^*(s_t, a_t) = E\left[r_t + \gamma \max_{a_{t+1}} Q^*(s_{t+1}, a_{t+1})\right]$$

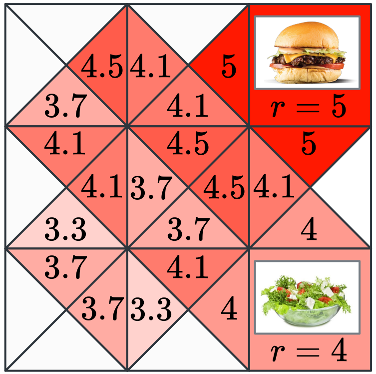
- Value-based policy, e.g. ε-Greedy:

$$\pi(s) = \begin{cases} \arg\max_a Q(s,a) & \mid \text{with } \Pr = 1 - \epsilon \\ \text{random action} & \mid \text{with } \Pr = \epsilon \end{cases}$$

- Q-Learning: After each transition (s_t, a_t, r_t, s_{t+1})

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left(\underbrace{r_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1})}_{\text{observation}} - \underbrace{Q(s_t, a_t)}_{\text{prediction}} \right)$$

Grid-world example



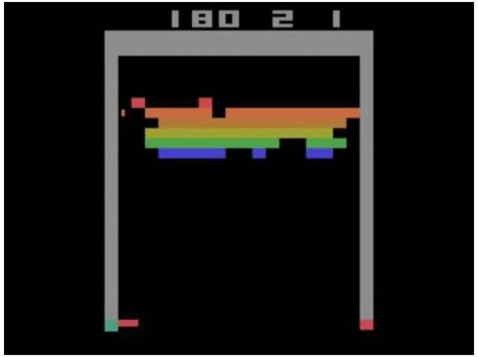
$\gamma = 0.9$

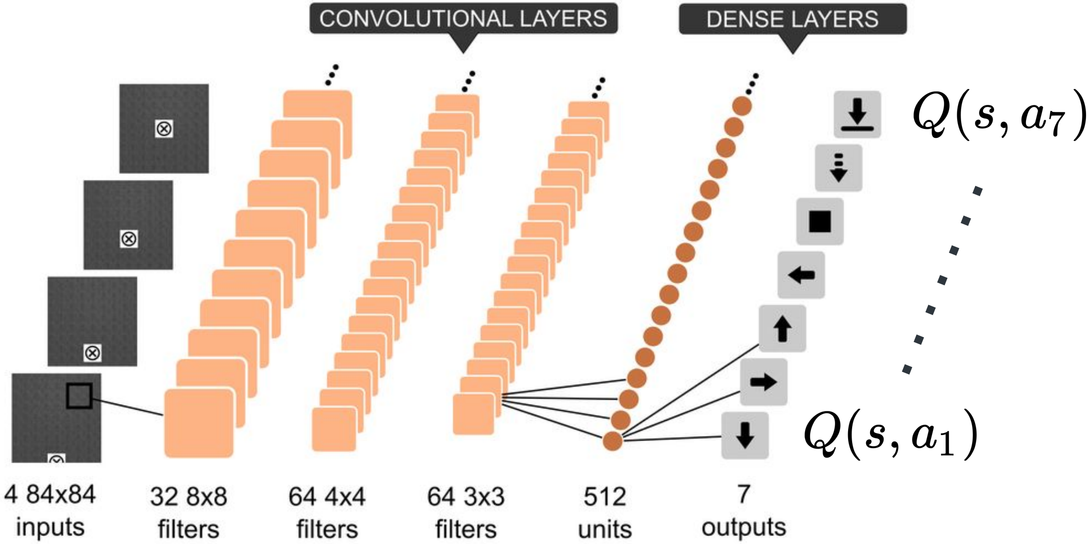
Barto & Sutton (2018)

Deep Reinforcement Learning

- Usage of neural networks to represent the policy and actions

<u>Atari games</u>





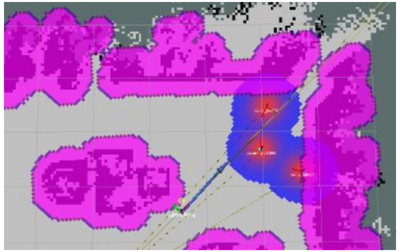
$$L(w) = \mathbb{E}[(\underbrace{r + \gamma \max_{a'} Q(s', a', w)}_{Target} - Q(s, a, w))^2]$$

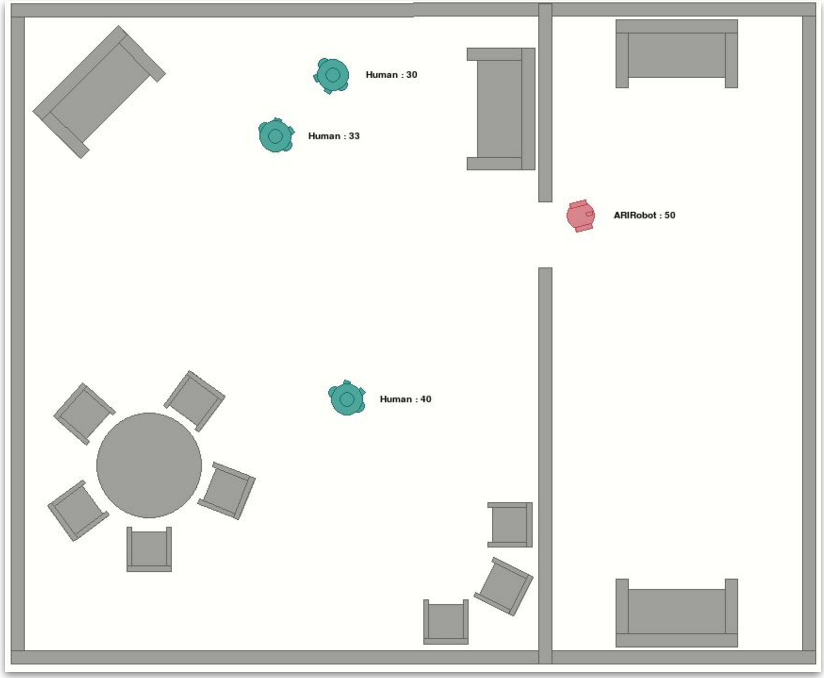
Mnih et al. (2013)

2D Simulator

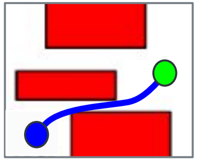
- Training a DRL navigation controller in simulation
  - 2D environment with obstacles
  - Robot with high-level sensor input
  - Humans based on social force model

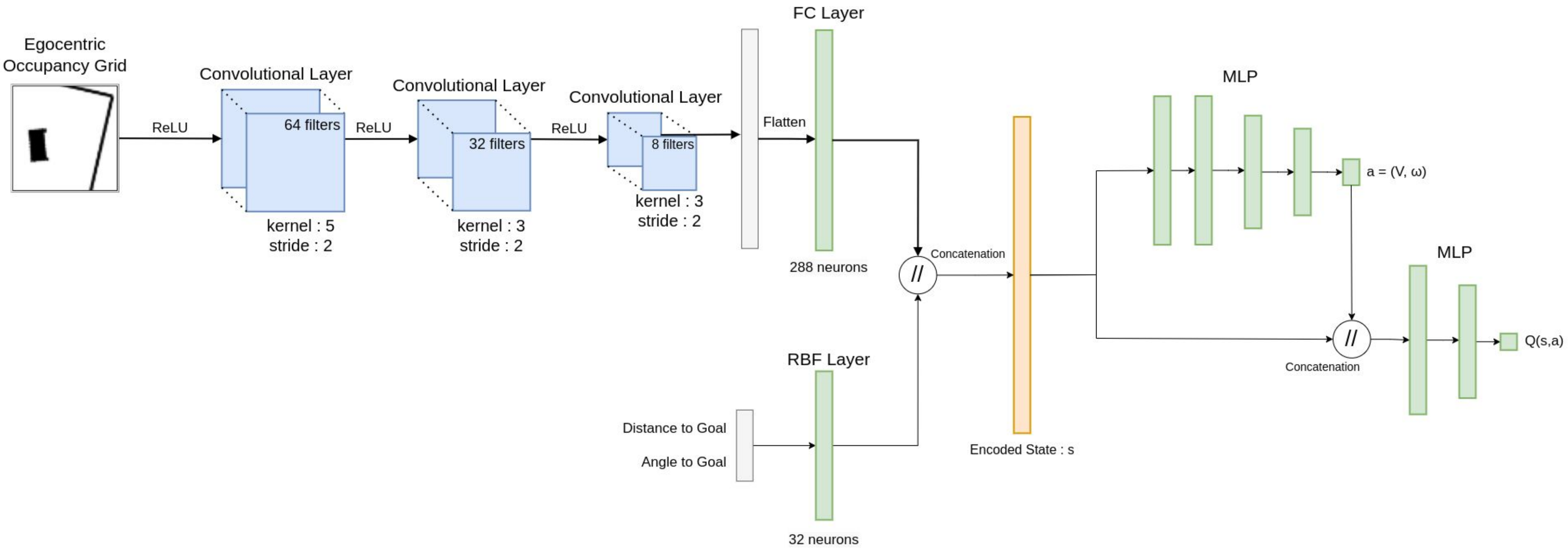
- Reward function: Cost maps





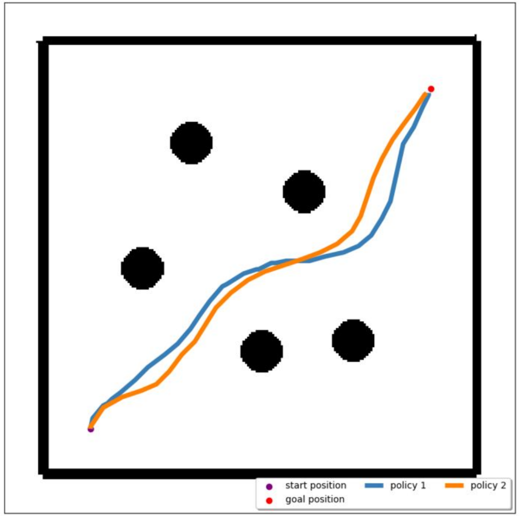
DRL Agent

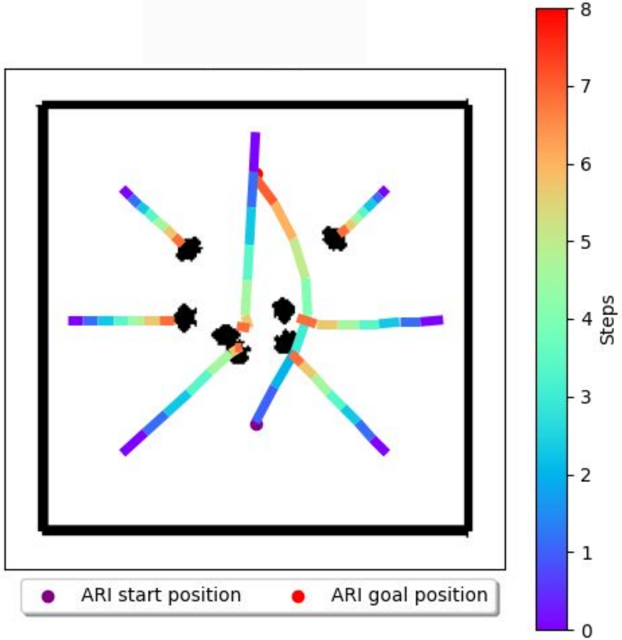




Egocentric Occupancy Grid
ReLU
Convolutional Layer
64 filters
kernel : 5
stride : 2
ReLU
Convolutional Layer
32 filters
kernel : 3
stride : 2
ReLU
Convolutional Layer
8 filters
kernel : 3
stride : 2
Flatten
FC Layer
288 neurons
Concatenation
//
RBF Layer
Distance to Goal
Angle to Goal
32 neurons
Encoded State : s
MLP
a = (V, ω)
MLP
//
Concatenation
Q(s,a)

Object Avoidance and Human-aware Navigation





→ Hands On Session

References

Janson, L., Schmerling, E., Clark, A., & Pavone, M. (2015). Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions. *The International journal of robotics research*, *34*(7), 883-921.

Truong, X. T., & Ngo, T. D. (2017). “To approach humans?”: A unified framework for approaching pose prediction and socially aware robot navigation. *IEEE Transactions on Cognitive and Developmental Systems*, *10*(3), 557-572.

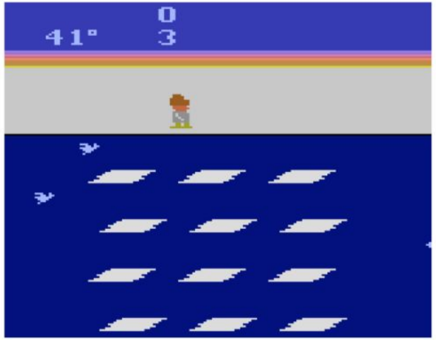
Hall, E. T., Birdwhistell, R. L., Bock, B., Bohannan, P., Diebold Jr, A. R., Durbin, M., ... & Vayda, A. P. (1968). Proxemics. *Current anthropology*, *9*(2/3), 83-108.

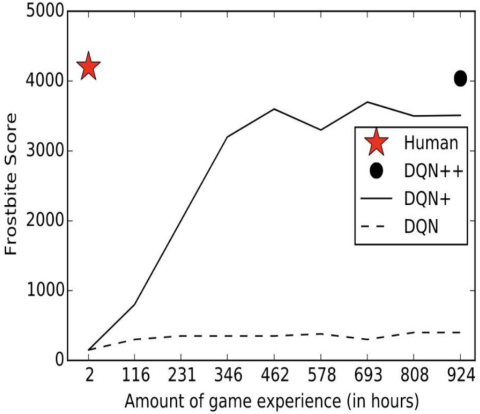
Kendon, A. (1990). Conducting interaction: Patterns of behavior in focused encounters. CUP Archive, volume 7.

Camacho E. F., & Bordons, C. (2007). Model predictive control. Springer.

Deep Reinforcement Learning

- Problem: Training needs a lot of data and time!



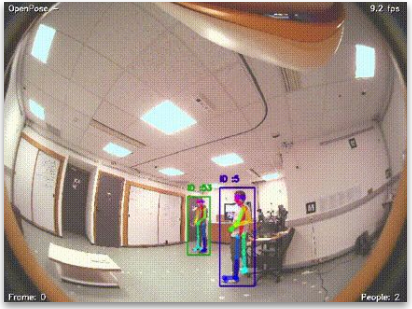


- Robotics:
  - Collecting data is time and cost intensive
  - Robots can break

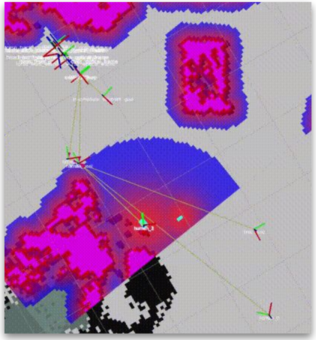
Lake et al. (2017)

Deep Reinforcement Learning

- Train in simulation and use policy in robot (sim2real transfer)
  - Problem: Domain shift
- Solution for SPRING:
  - Use high-level observations as policy input which can be easier simulated



Camera



Map