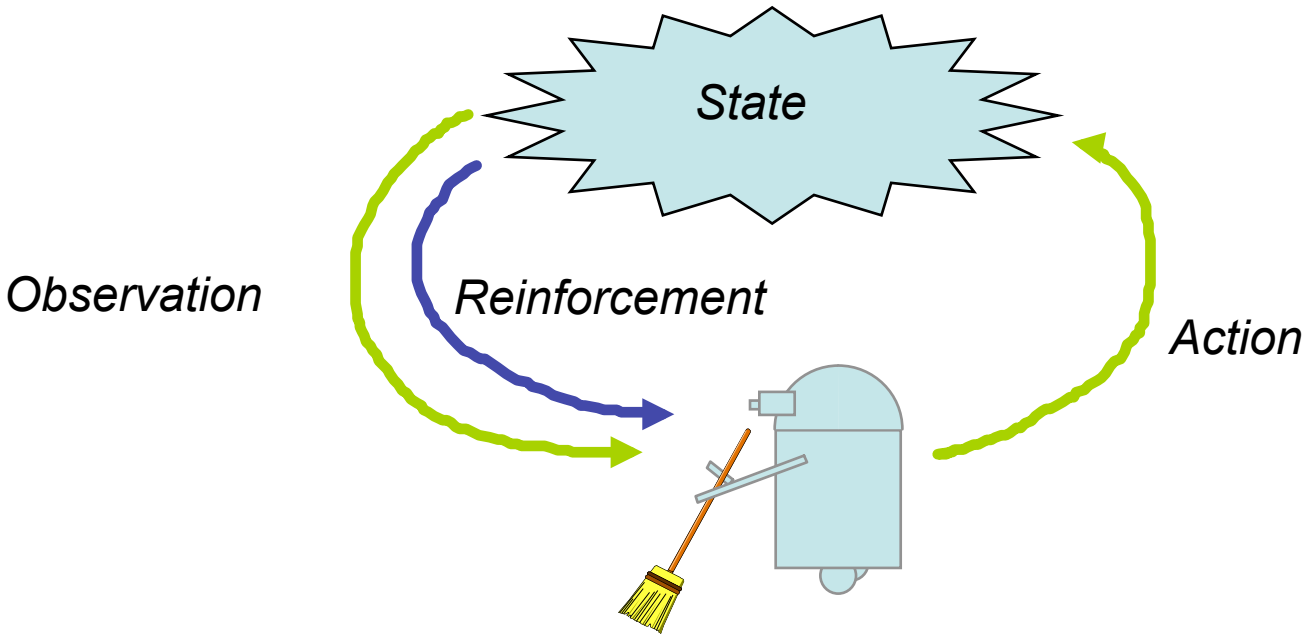
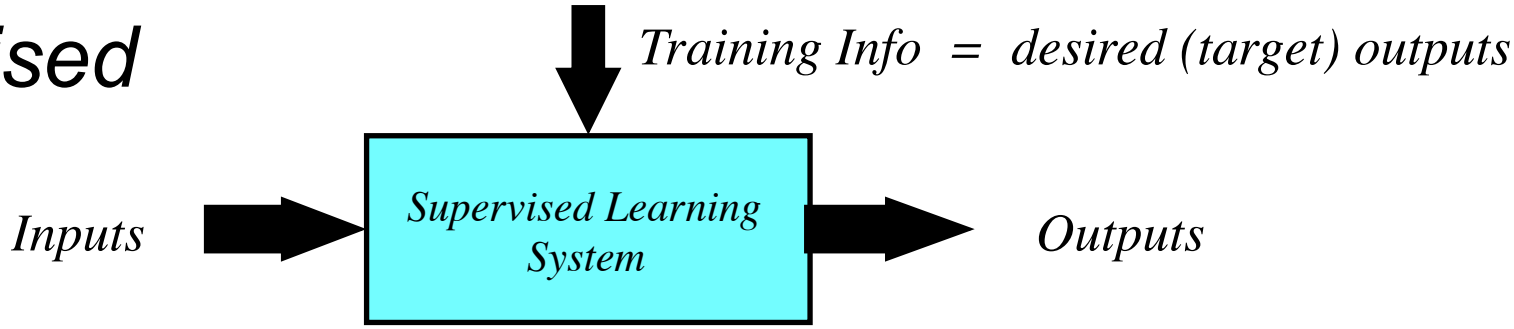
4 Reinforcement Learning (RL)

- Learning by interaction with the environment to accomplish a predefined goal
- Learn what to do – how to map situations with actions – to maximize a numerical reward value



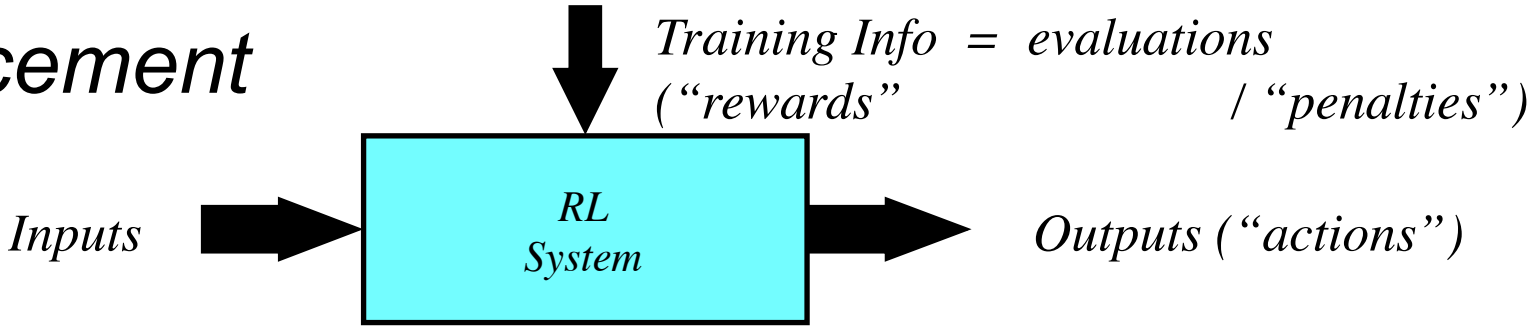
Supervised / Reinforcement learning

Supervised



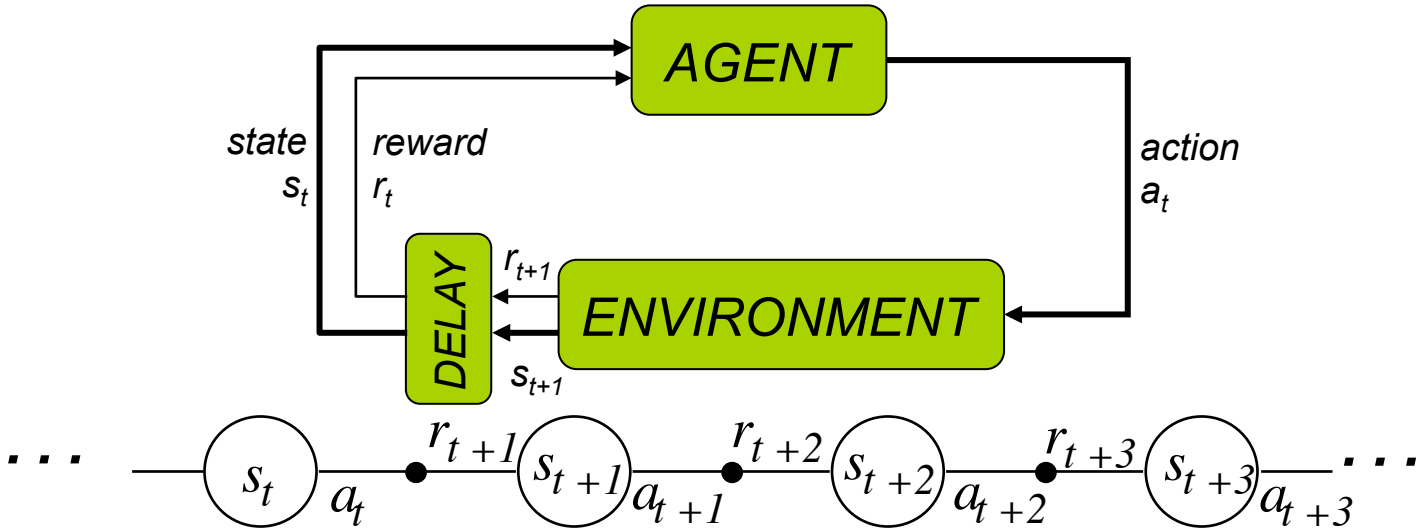
Error = (target output − actual output)

Reinforcement



Objective: get as much reward as possible

Reinforcement Learning Problem

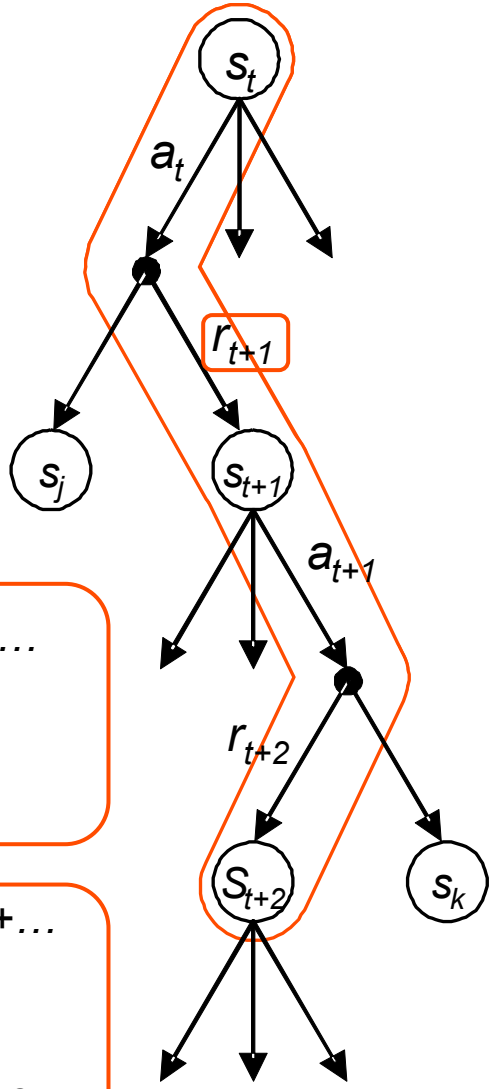


Agent: - ***Learner*** *and* ***decision maker****.*
- *Input: the* ***state*** *"s_t" and a* ***reward*** *"r_t" (numerical value)*
- *Output: an* ***action*** *"a_t"*
- *Goal: to* ***maximize*** *the* ***amount of future rewards*** Σr_t

Environment: - *Everything* ***outside*** *the agent.*
- *It reacts to actions with a* ***new state****.*
- *A* ***Reinforcement Function*** *provides the new* ***reward****.*

RL elements

- ***Policy Function, $\pi(s,a)$: S → A***
  - *Learner function*
  - *State/action mapping*
  - *Kinds: random, greedy, ε-greedy, optimal*
- ***Dynamics Function, $\{s_t,a_t\} \rightarrow s_{t+1}$***
  - *Environment function*
  - *Unknown function*
- ***Reinforcement Function, $r(s_{t+1},a_t)=r_{t+1}$***
  - *Environment function*
  - *Defines the goal of the learner*
- ***Value Function, $V^\pi(s)$; $Q^\pi(s,a)$***
  - *Learner function*
  - *Expected sum of*
  - *Kinds:- State-Value*
    - *Action-Value function $Q^\pi(s,a)$*



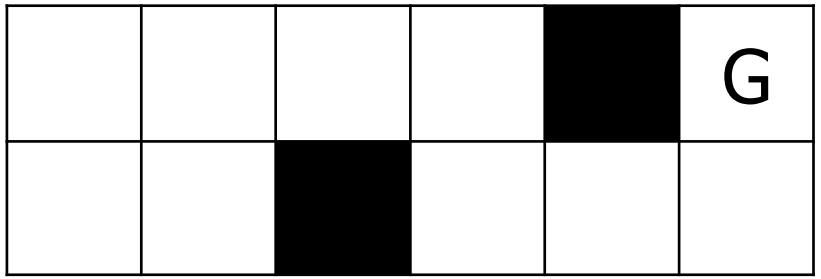
$V^\pi(s_t)=r_{t+1}+ r_{t+2}+\ldots$
$\pi(s_t)=a_t$
$\pi(s_{t+1})=a_{t+1}$
$\pi(s_{t+2})=\ldots$

$Q^\pi(s_t,a_t)=r_{t+1}+ r_{t+2}+\ldots$
$\pi(s_{t+1})=a_{t+1}$
$\pi(s_{t+2})=\ldots$

π^* easy to extract from Q !!*

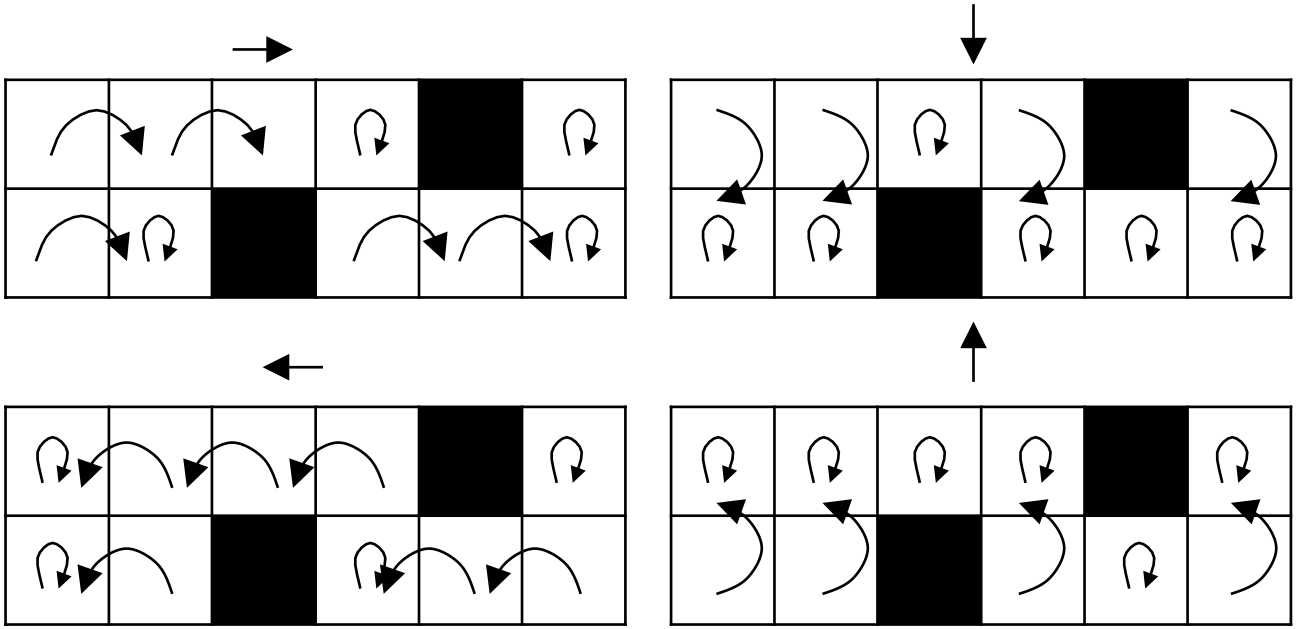
Example

Environment:



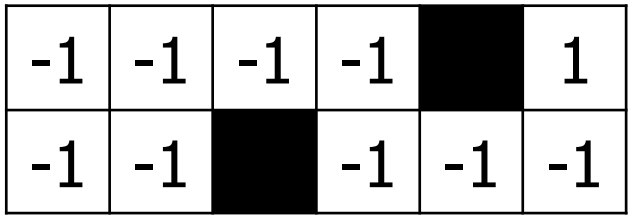
Actions:

Dynamics:

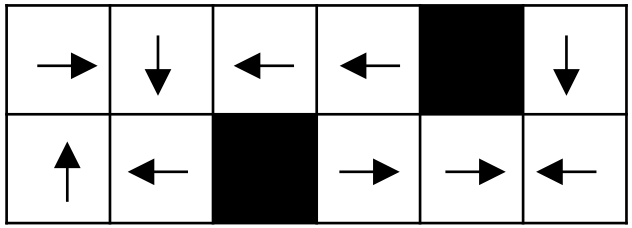


Example

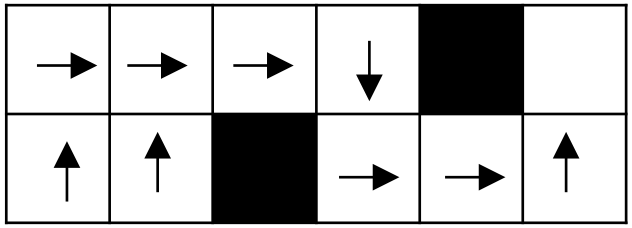
The reinforcement function:



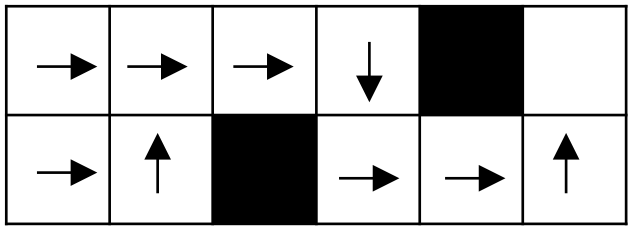
A policy:



An optimal policy, π^*_1 :

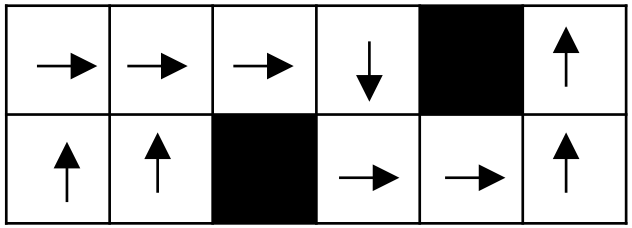


Another optimal policy, π^*_2:



Example

Policy, π^*_1 :



The state value function, $V^*_1(s)$:

-5	-4	-3	-2	■	
-6	-5	■	-1	0	1

The action value function, $Q^*_1(s,a)$:

→

-5	-4	-3	-3	■	
-6	-6	■	-1	0	0

↓

-7	-6	-4	-2	■	
-7	-6	■	-2	-1	0

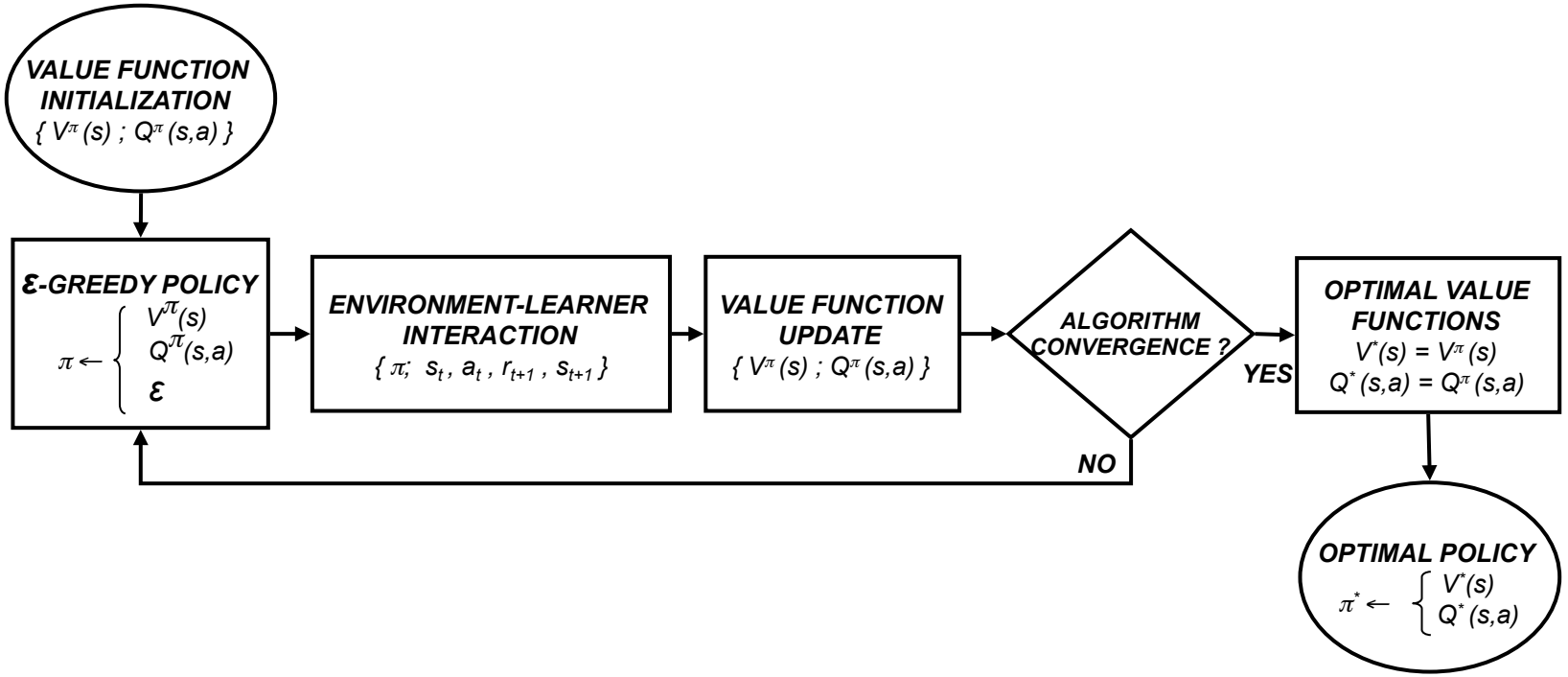
←

-6	-6	-5	-4	■	
-7	-7	■	-2	-2	-1

↑

-6	-5	-4	-3	■	
-6	-5	■	-3	-1	1

Intuitive Solution



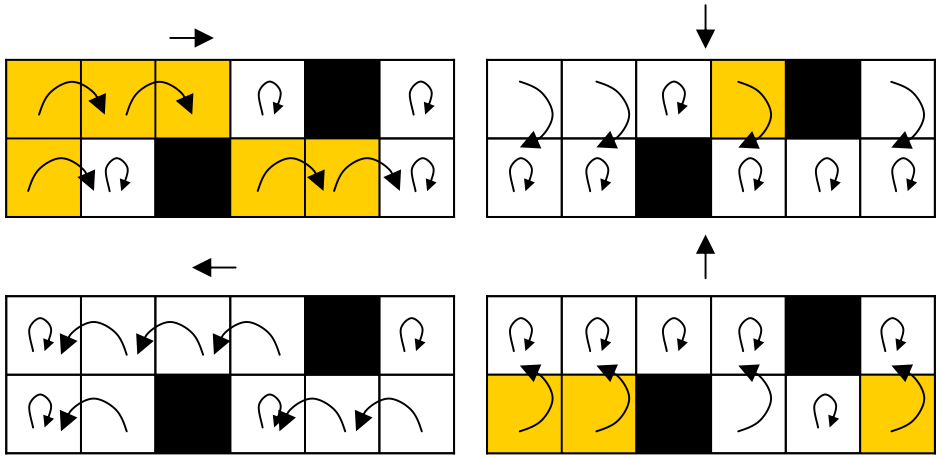
π* from V*

V(s):*

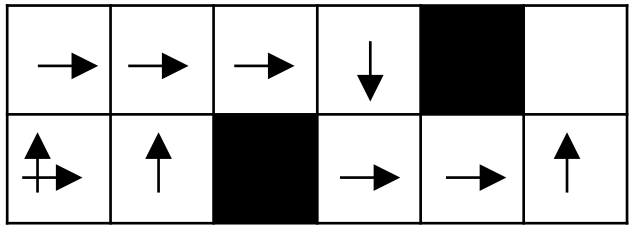
-5	-4	-3	-2	■	
-6	-5	■	-1	0	1

+

Dynamics:

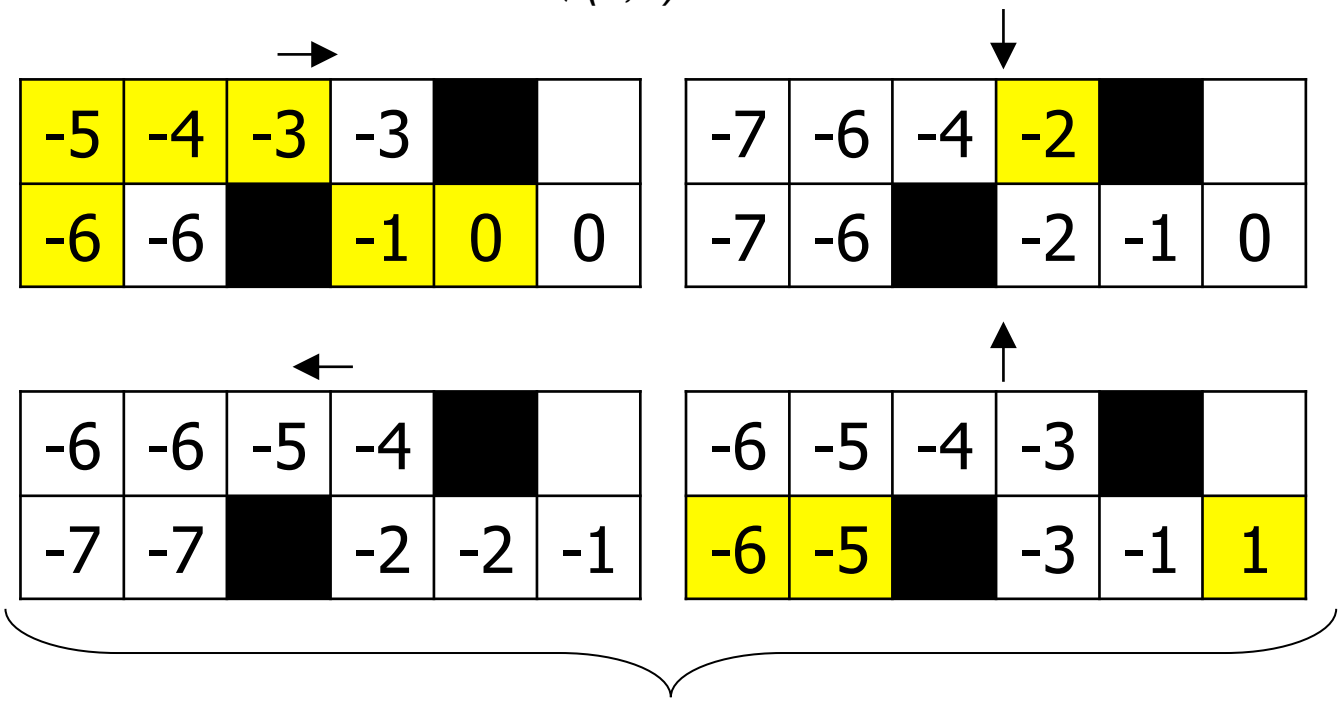


π:*

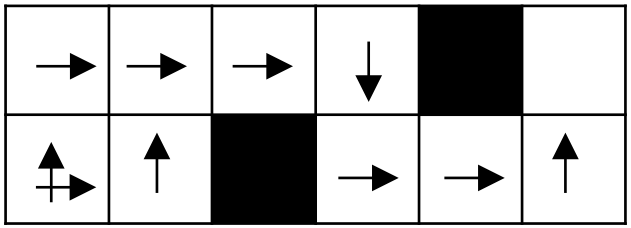


π^* from Q*

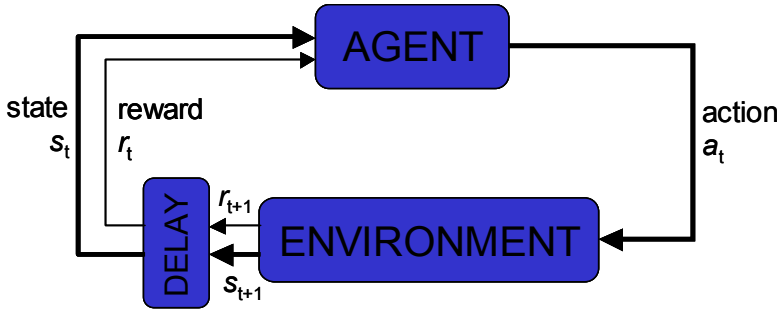
$Q^*(s,a)$:



π^*: $s_i \rightarrow a_{max}$ | Max $Q^*(s_i,a_{max})$:

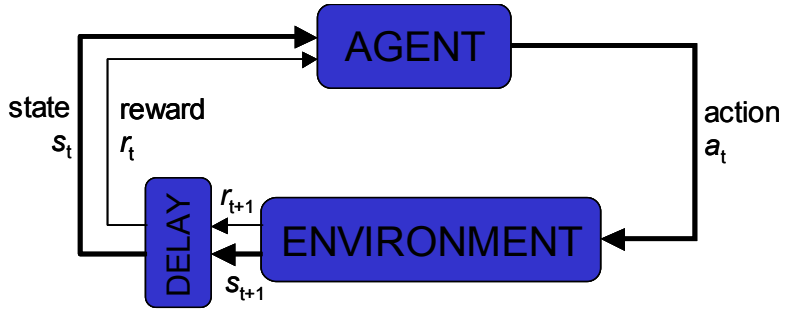


RL features



- Learning by rewards penalization, without examples
- Trial error search
- Dilemma between exploration / exploitation
  - exploration: to find the optimal actions
  - exploitation: to receive a higher reward
- Delayed rewards
  - ignore immediate reward to obtain better future rewards

RL elements

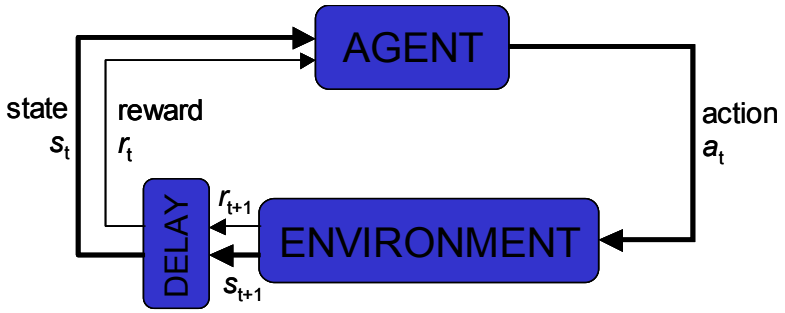


- Policy:
  - mapping between states s_t and actions a_t

$$\pi(s,a): S \rightarrow A$$

  - *The goal is to find the **optimal policy** π^**
- Reinforcement function
  - Mapping between each state/action pair with a reward
  - Defines the goal to be fulfilled
  - An external knowledge is necessary

RL elements



- Value functions
  - They contain the expected sum of rewards when following a particular policy.
  - Kinds:
    - State Value Function: sum of rewards that can be expected when being in a particular state

$$\mathbf{V^{\pi}(s)}$$

    - Action Value Function: sum of rewards that can be expected when being in a particular state and a particular action is applied:

$$\mathbf{Q^{\pi}(s,a)}$$

Representation with FMDP

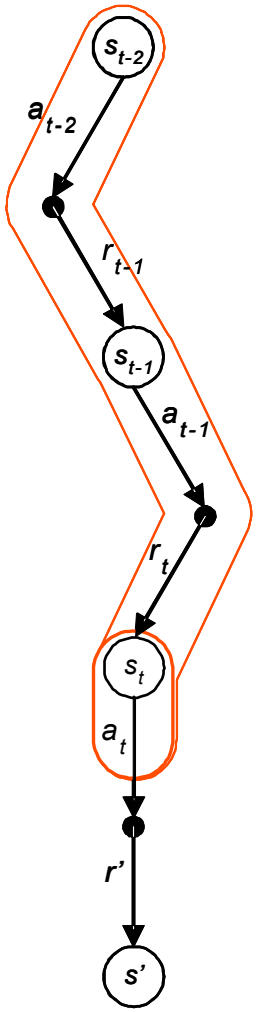
- FMDP: Finite Markov Decision Process
- Mathematical formulation of the RLP
- Constraints
  - Markov Property:

$$\Pr\left\{s_{t+1}=s',r_{t+1}=r \mid s_t,a_t,r_t,s_{t-1},a_{t-1},\mathrm{K}\ ,r_1,s_0,a_0\right\}=$$

$$\Pr\left\{s_{t+1}=s',r_{t+1}=r \mid s_t,a_t\right\}$$

→ Complete state

  - Finite actions and states



Definition of a RLP with FMDP

- Discrete set of states and actions
- State Transition Probability:

$$P^{a}_{ss'} = \Pr\left\{ s_{t+1} = s' \mid s_t = s, a_t = a \right\} \quad \text{for all } s, s' \in S, a \in A(s).$$

- Reinforcements Probability:

$$R^{a}_{ss'} = E\left\{ r_{t+1} \mid s_t = s, a_t = a, s_{t+1} = s' \right\} \quad \text{for all } s, s' \in S, a \in A(s).$$

$P^{a}_{ss'}$ and $R^{a}_{ss'}$ specify the dynamics of the FMDP

Optimal value functions

Using the Bellman equation:

- Optimal State Value function

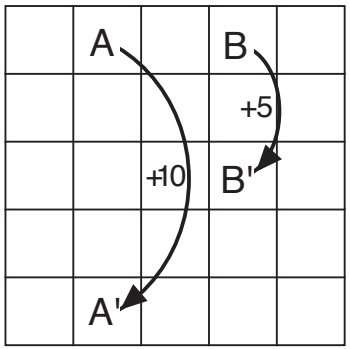
$$V^*(s) = \max_{a \in A(s)} \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma V^*(s') \right]$$

- Optimal Action Value Function

$$Q^*(s,a) = \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma \max_{a'} Q^*(s',a') \right]$$

Obtaining an optimal policy

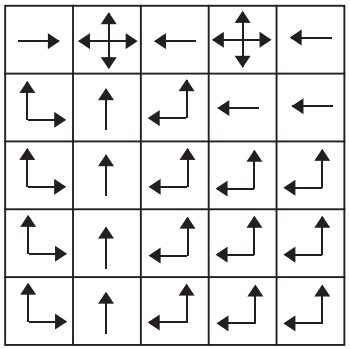
- From V^*: A policy $\pi(s,a)$, for that $\forall s \in S$, the new state s' accomplish that $V^*(s') > V^*(s)$ is an optimal policy



a) gridworld

22.0	24.4	22.0	19.4	17.5
19.8	22.0	19.8	17.8	16.0
17.8	19.8	17.8	16.0	14.4
16.0	17.8	16.0	14.4	13.0
14.4	16.0	14.4	13.0	11.7

b) V^*

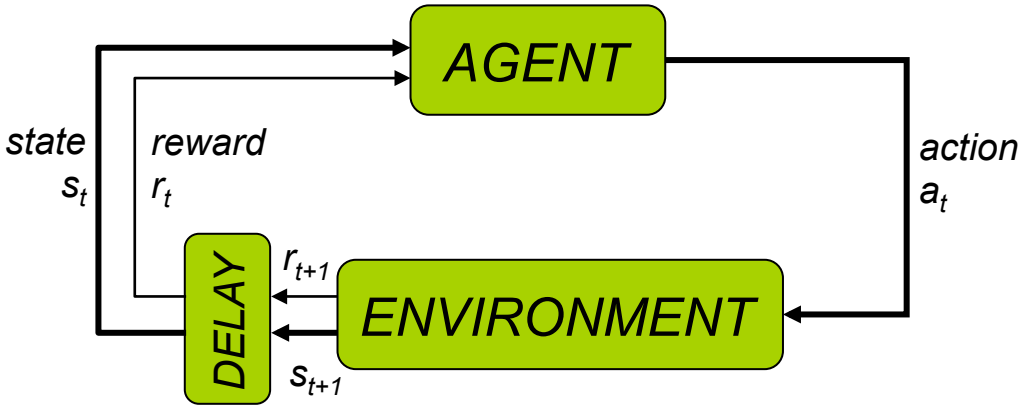


c) π^*

- From Q^*: Given the function $Q^*(s,a)$

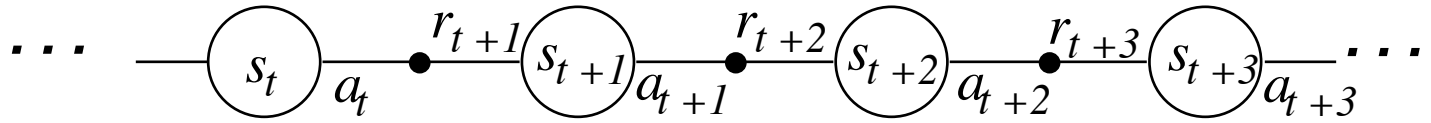
$$\pi^*(s) = \arg\max_{a \in A(s)} Q^*(s,a)$$

Solutions to the RLP. Methodologies



$$P^a_{ss'} = \Pr\left\{ s_{t+1} = s' \mid s_t = s, a_t = a \right\}$$

$$R^a_{ss'} = E\left\{ r_{t+1} \mid s_t = s, a_t = a, s_{t+1} = s' \right\}$$



- Dynamic Programming
- Monte-Carlo Methods
- Temporal Differences

Dynamic Programming

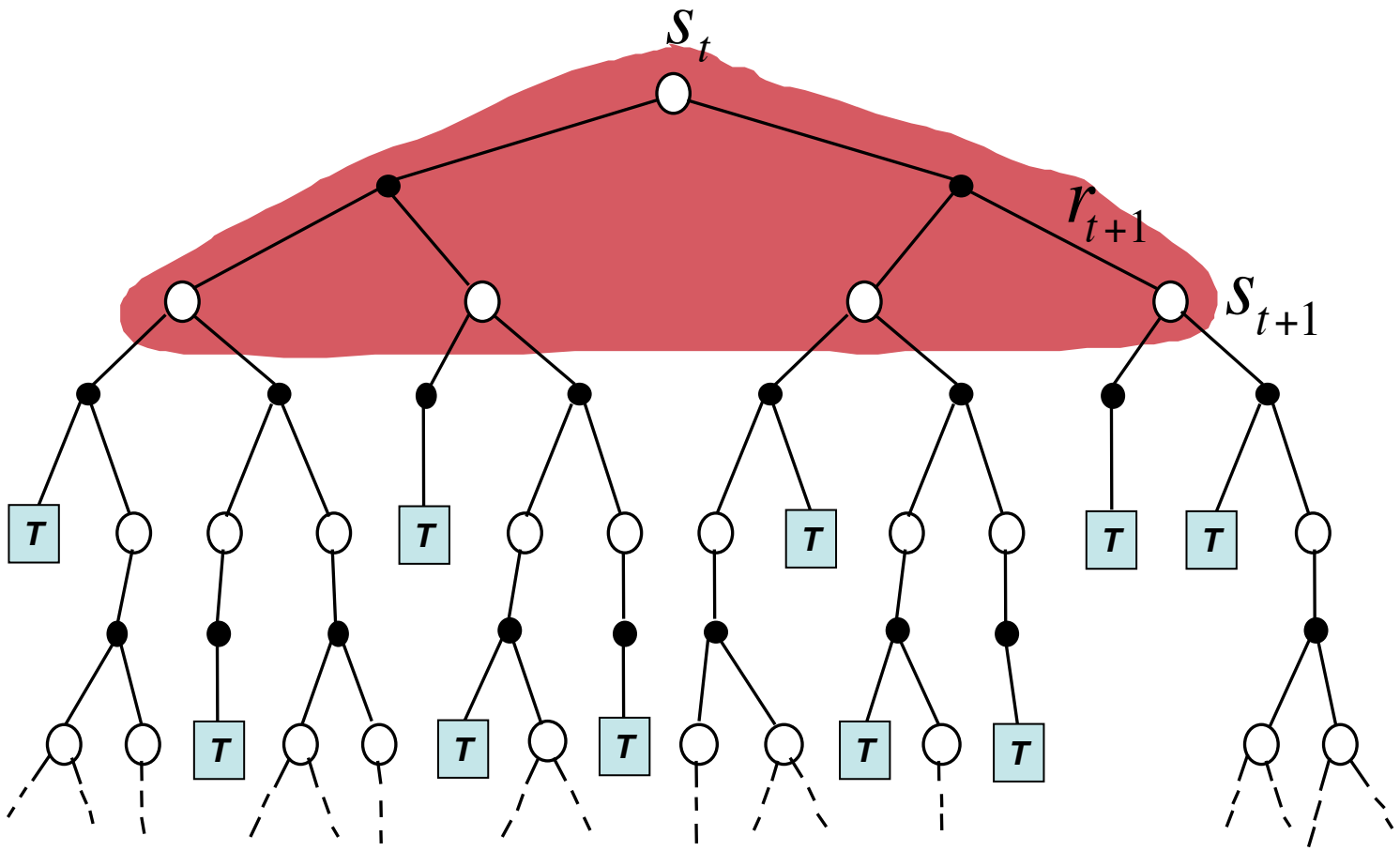
- Algorithms that find the optimal policies from the model of the environment $P^a_{ss'}$, i $R^a_{ss'}$,

$$V^*(s) = \max_{a \in A(s)} \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma V^*(s') \right]$$

$$Q^*(s,a) = \sum_{s'} P^a_{ss'} \left[R^a_{ss'} + \gamma \max_{a'} Q^*(s',a') \right]$$

Dynamic Programming

$$V(s_t) \leftarrow E_\pi \left\{ r_{t+1} + \gamma V(s_{t+1}) \right\}$$



Monte-Carlo Methods

- **Do not** use the model of the environment to update the value functions.
- No estimation of the future accumulated reward.
- Value functions are updated at the end of the episode.

Simple every - visit Monte Carlo method :

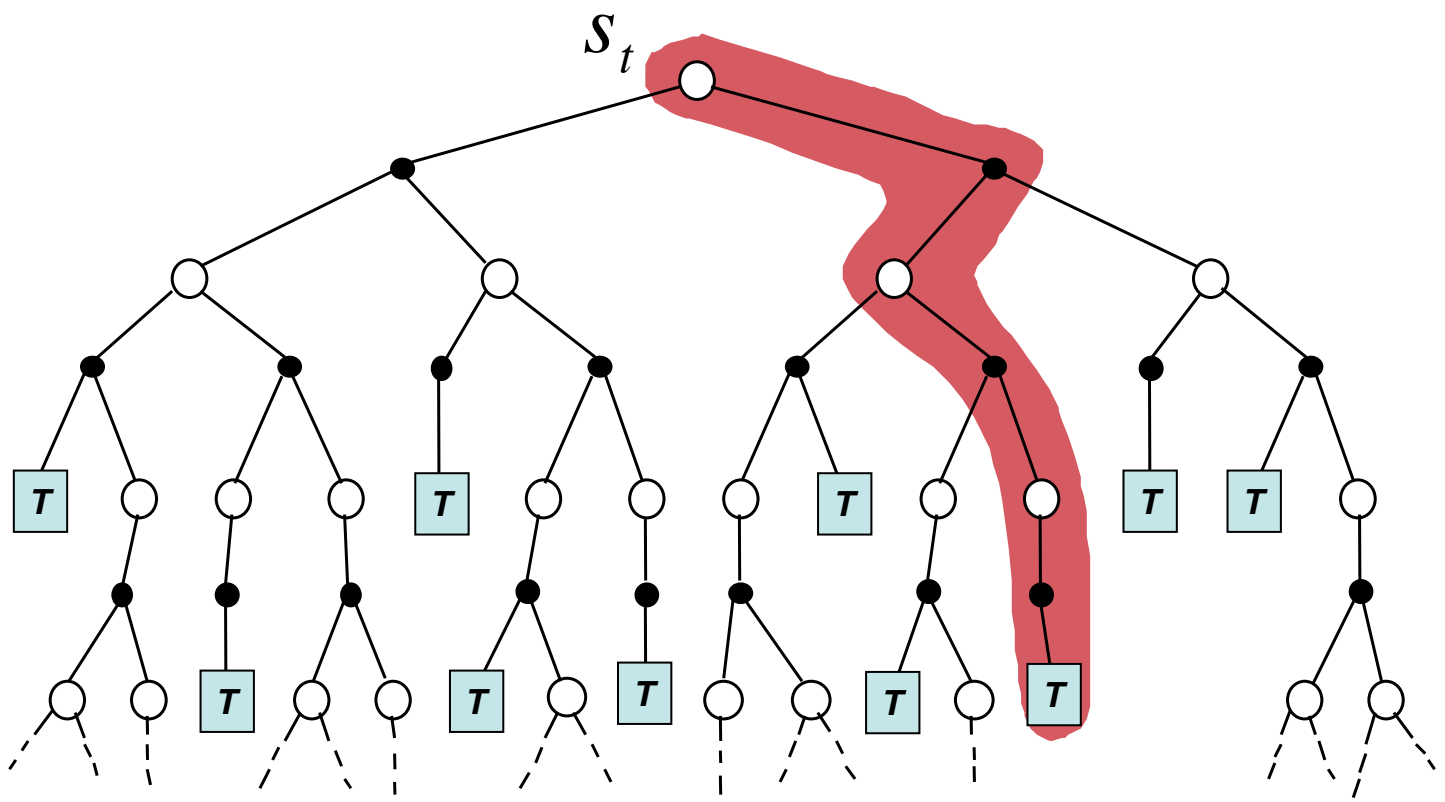
$$V(s_t) \leftarrow V(s_t) + \alpha \left[R_t - V(s_t) \right]$$

target: *the actual return after time t*

Monte-Carlo Methods

$$V(s_t) \leftarrow V(s_t) + \alpha\left[R_t - V(s_t)\right]$$

where R_t is the actual return following state s_t.



Temporal differences

- **Do not** use the model of the environment to update the value functions.
- An estimation of the future accumulated reward is done to update value functions (**online** learning).

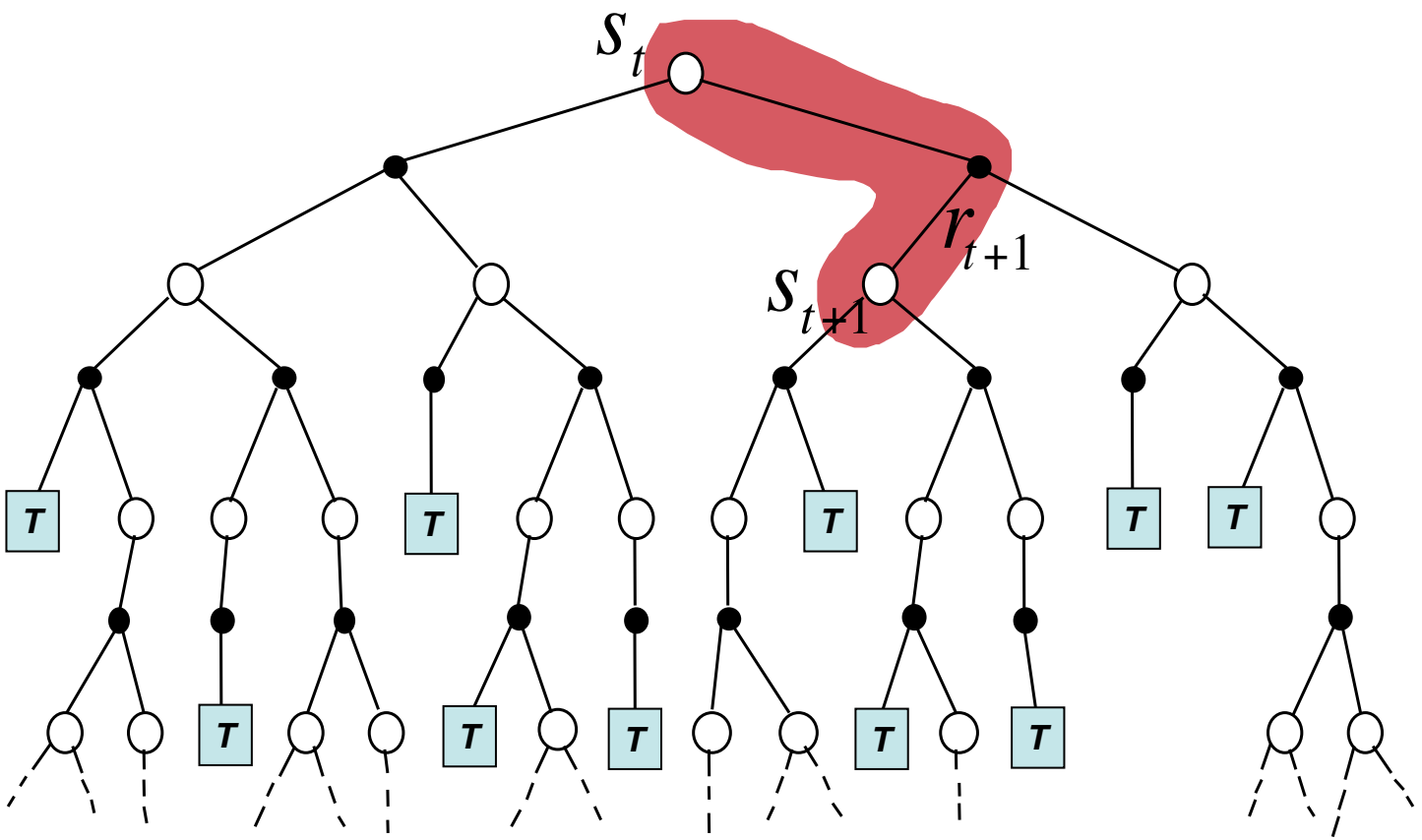
The simplest TD method, TD(0):

$$V(s_t) \leftarrow V(s_t) + \alpha\left[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)\right]$$

target: *an estimate of the return*

Temporal differences

$$V(s_t) \leftarrow V(s_t) + \alpha\left[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)\right]$$



Methodologies comparison

	ONLINE LEARNING	DYNAMICS FUNCTION
Dynamic Programming	YES	YES
Monte Carlo Methods	NO	NO
Temporal Difference	YES	NO

suitable for robotics

- *Actor Critic methods*
- *Q_learning*
- *SARSA*

Q_learning

- **TD method**
- **Action-Value Function:** Q(s,a)
- Off-policy **learning : any policy can be followed**
- **Convergence** guaranteed **as far as all the (s,a) pairs are** regularly **visited**

Q_learning algorithm

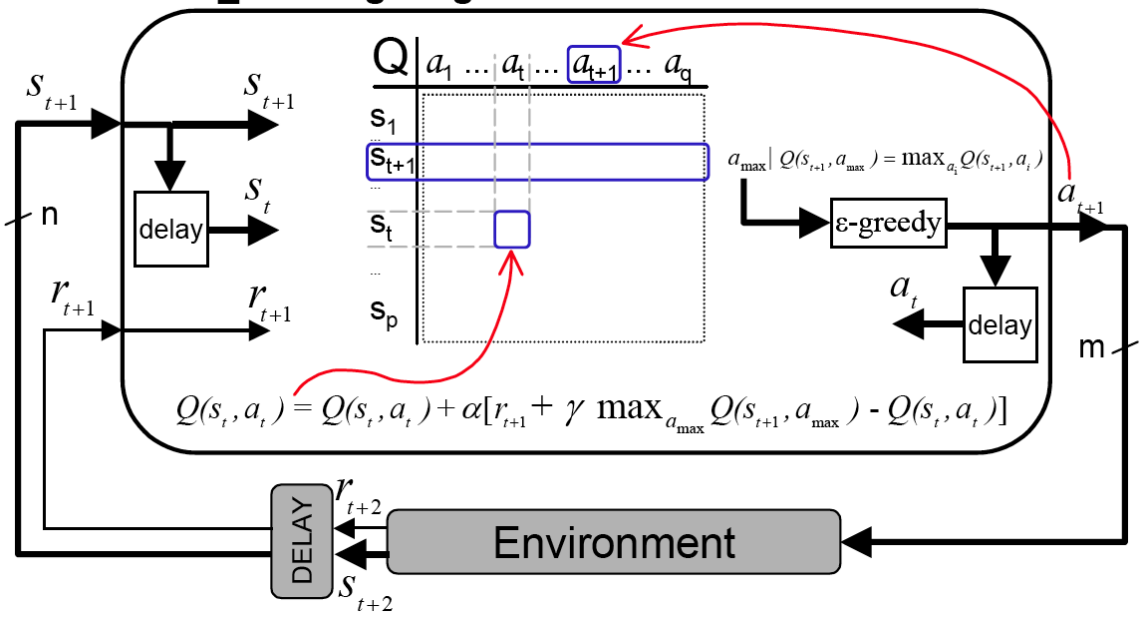
Initialize Q(s,a) randomly
Initialize s
Repeat

Choose *a* following ε-greedy
Take action *a*, observe *r*, *s'*
$Q(s,a) \leftarrow Q(s,a) + \alpha [r + \gamma \ max Q_{a'}(s',a') - Q(s,a)]$
$s \leftarrow s'$

until Q(s,a) is optimal

Q_learning algorithm

Q_learning diagram



$$Q(s,a) \leftarrow Q(s,a) + \alpha [\, r + \gamma \; max Q_{a'}(s',a') - Q(s,a)]$$

parameters

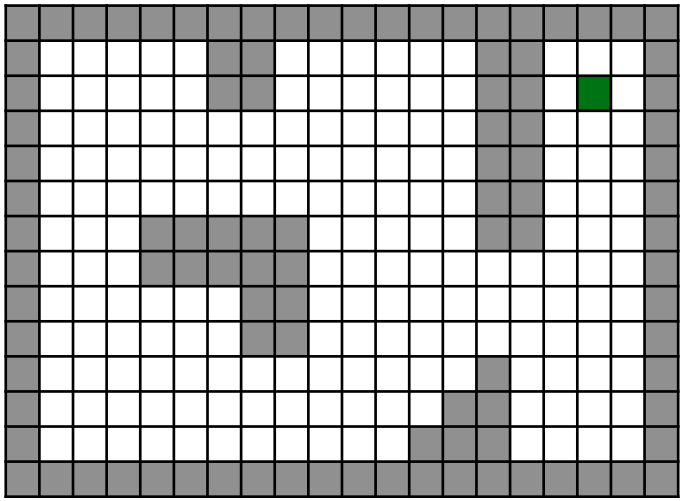
α: learning rate

γ: discount factor

ε: exploration

***target**: an estimate of the return*

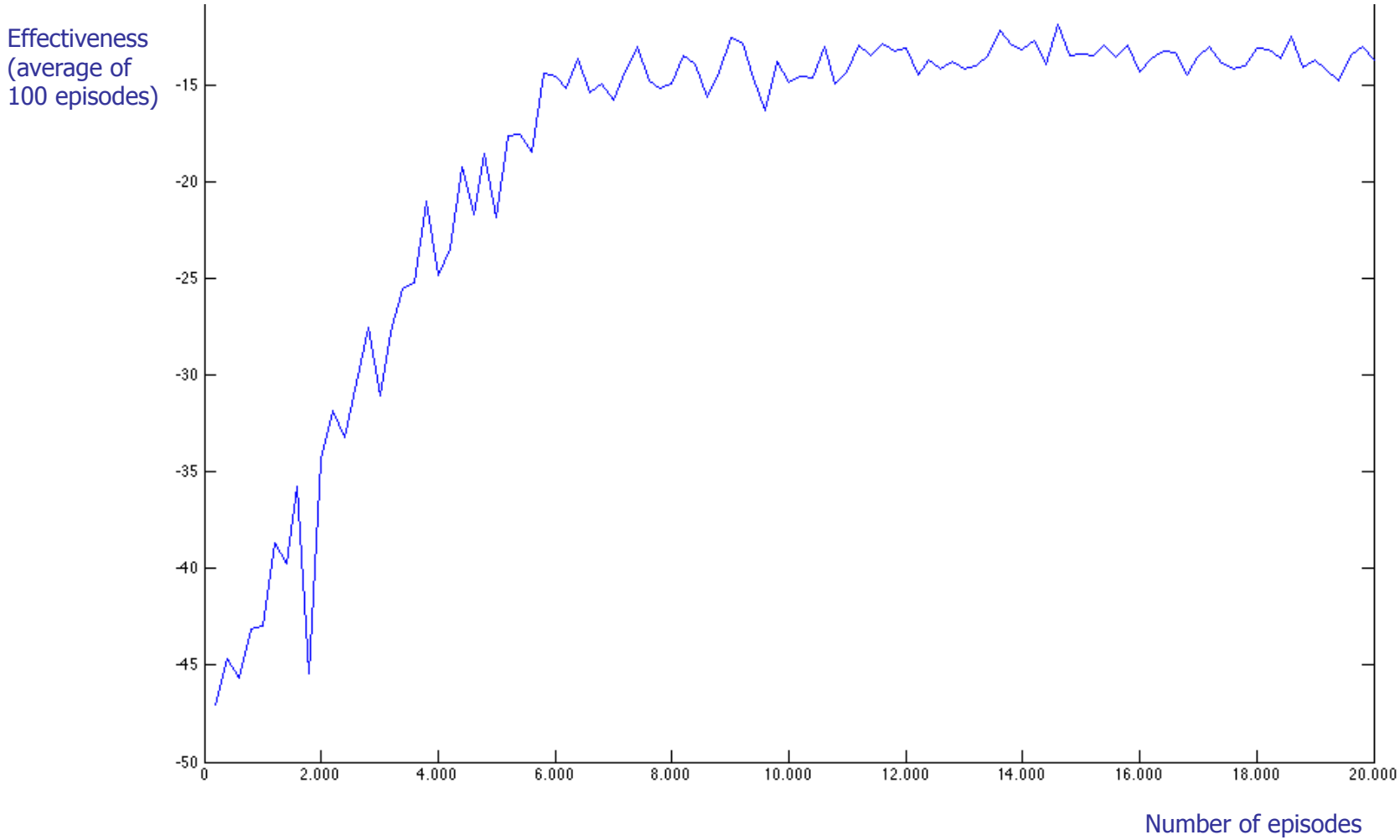
Example: Path planning



- States: 2D finite environment, 20x14 cells
- Actions: 4-point connectivity
- Reward: "-1" in all cells except the goal "+1"
- Dynamics: 1 step in the action direction unless obstacle is found
- Q function: 20x14x4 = 1120 cells
- Random initial position
- Max. iterations per episode = 50
- Number of episodes = 20.000
- Effectiveness : "averaged number of iterations to reach the goal"
- Learning parameters
  - Learning rate: α = 0.1
  - Discount factor: γ = 0.9
  - Exploration: ε = 0.3

Example: Path planning

- Effectiveness evolution



Example: Path planning

- Q function

↑

0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000
0,000	-9,290	-9,293	-9,267	-9,200	-9,090	0,000	0,000	-8,567	-8,560	-8,477	-8,261	-8,110	-7,921	0,000	0,000	-1,090	-0,927	-0,169	0,000
0,000	-9,247	-9,252	-9,183	-9,120	-9,023	0,000	0,000	-8,463	-8,477	-8,333	-8,159	-7,961	-7,735	0,000	0,000	-0,100	0,000	1,000	0,000
0,000	-9,181	-9,186	-9,119	-9,024	-8,917	-8,796	-8,663	-8,512	-8,349	-8,165	-7,961	-7,735	-7,484	0,000	0,000	-1,090	-1,090	-0,100	0,000
0,000	-9,086	-9,109	-9,022	-8,916	-8,796	-8,663	-8,514	-8,349	-8,166	-7,962	-7,735	-7,484	-7,204	0,000	0,000	-1,981	-1,981	-1,090	0,000
0,000	-9,023	-9,025	-8,917	-8,796	-8,663	-8,514	-8,349	-8,166	-7,962	-7,735	-7,484	-7,204	-6,893	0,000	0,000	-2,783	-2,783	-1,981	0,000
0,000	-9,097	-9,106	-9,024	0,000	0,000	0,000	0,000	0,000	-7,484	-7,484	-7,204	-6,893	-6,548	0,000	0,000	-3,505	-3,505	-2,783	0,000
0,000	-9,183	-9,162	-9,114	0,000	0,000	0,000	0,000	0,000	-7,204	-7,204	-6,893	-6,548	-6,165	-5,738	-5,265	-4,739	-4,154	-3,505	0,000
0,000	-9,235	-9,171	-9,162	-9,090	-8,985	-8,887	0,000	0,000	-7,484	-7,484	-7,204	-6,893	-6,548	-6,165	-5,738	-5,265	-4,739	-4,154	0,000
0,000	-9,155	-9,180	-9,107	-9,019	-8,916	-8,796	0,000	0,000	-7,735	-7,735	-7,484	-7,204	-6,893	-6,548	-6,164	-5,738	-5,265	-4,739	0,000
0,000	-9,108	-9,121	-9,025	-8,917	-8,796	-8,663	-8,514	-8,349	-8,166	-7,962	-7,735	-7,483	-7,201	0,000	-6,137	-6,147	-5,738	-5,265	0,000
0,000	-9,160	-9,191	-9,117	-9,019	-8,909	-8,796	-8,663	-8,514	-8,349	-8,165	-7,961	-7,723	0,000	0,000	-6,321	-6,519	-6,155	-5,738	0,000
0,000	-9,232	-9,197	-9,180	-9,093	-9,005	-8,897	-8,785	-8,653	-8,496	-8,346	-8,157	0,000	0,000	0,000	-6,607	-6,528	-6,429	-6,164	0,000
0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000

←

0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000
0,000	-9,320	-9,259	-9,193	-9,109	-9,006	0,000	0,000	-8,540	-8,490	-8,285	-8,133	-7,927	-7,697	0,000	0,000	-1,089	-0,091	-0,784	0,000
0,000	-9,304	-9,232	-9,198	-9,112	-9,019	0,000	0,000	-8,567	-8,477	-8,325	-8,158	-7,959	-7,734	0,000	0,000	-1,090	0,000	-1,057	0,000
0,000	-9,255	-9,188	-9,112	-9,023	-8,917	-8,663	-8,514	-8,513	-8,349	-8,165	-7,962	-7,735	-7,484	0,000	0,000	-0,100	1,000	-0,100	0,000
0,000	-9,145	-9,114	-9,022	-8,916	-8,796	-8,663	-8,514	-8,349	-8,166	-7,962	-7,735	-7,484	-7,204	0,000	0,000	-1,090	-0,100	-1,090	0,000
0,000	-9,115	-9,025	-8,917	-8,796	-8,663	-8,514	-8,349	-8,166	-7,962	-7,735	-7,484	-7,204	-6,893	0,000	0,000	-1,981	-1,090	-1,981	0,000
0,000	-9,025	-8,917	-8,796	0,000	0,000	0,000	0,000	0,000	-7,735	-7,484	-7,204	-6,893	-6,548	0,000	0,000	-2,783	-1,981	-2,783	0,000
0,000	-9,122	-9,025	-8,917	0,000	0,000	0,000	0,000	0,000	-7,484	-7,204	-6,893	-6,548	-6,165	-5,265	-4,739	-3,505	-2,783	-3,505	0,000
0,000	-9,204	-9,122	-9,025	-9,001	-8,909	-8,793	0,000	0,000	-7,204	-6,893	-6,548	-6,165	-5,738	-5,265	-4,739	-4,154	-3,505	-4,154	0,000
0,000	-9,226	-9,174	-9,114	-9,010	-8,915	-8,796	0,000	0,000	-7,484	-7,204	-6,893	-6,548	-6,165	-5,738	-5,265	-4,739	-4,154	-4,739	0,000
0,000	-9,148	-9,119	-9,025	-8,917	-8,796	-8,663	-8,349	-8,166	-7,735	-7,484	-7,204	-6,893	-6,548	0,000	-5,738	-5,265	-4,739	-5,265	0,000
0,000	-9,123	-9,025	-8,917	-8,796	-8,663	-8,514	-8,349	-8,166	-7,962	-7,735	-7,484	-7,204	0,000	0,000	-6,164	-5,738	-5,265	-5,738	0,000
0,000	-9,209	-9,122	-9,025	-8,917	-8,796	-8,663	-8,514	-8,349	-8,166	-7,962	-7,735	0,000	0,000	0,000	-6,540	-6,164	-5,738	-6,164	0,000
0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000

Example: Path planning

Q function

↓

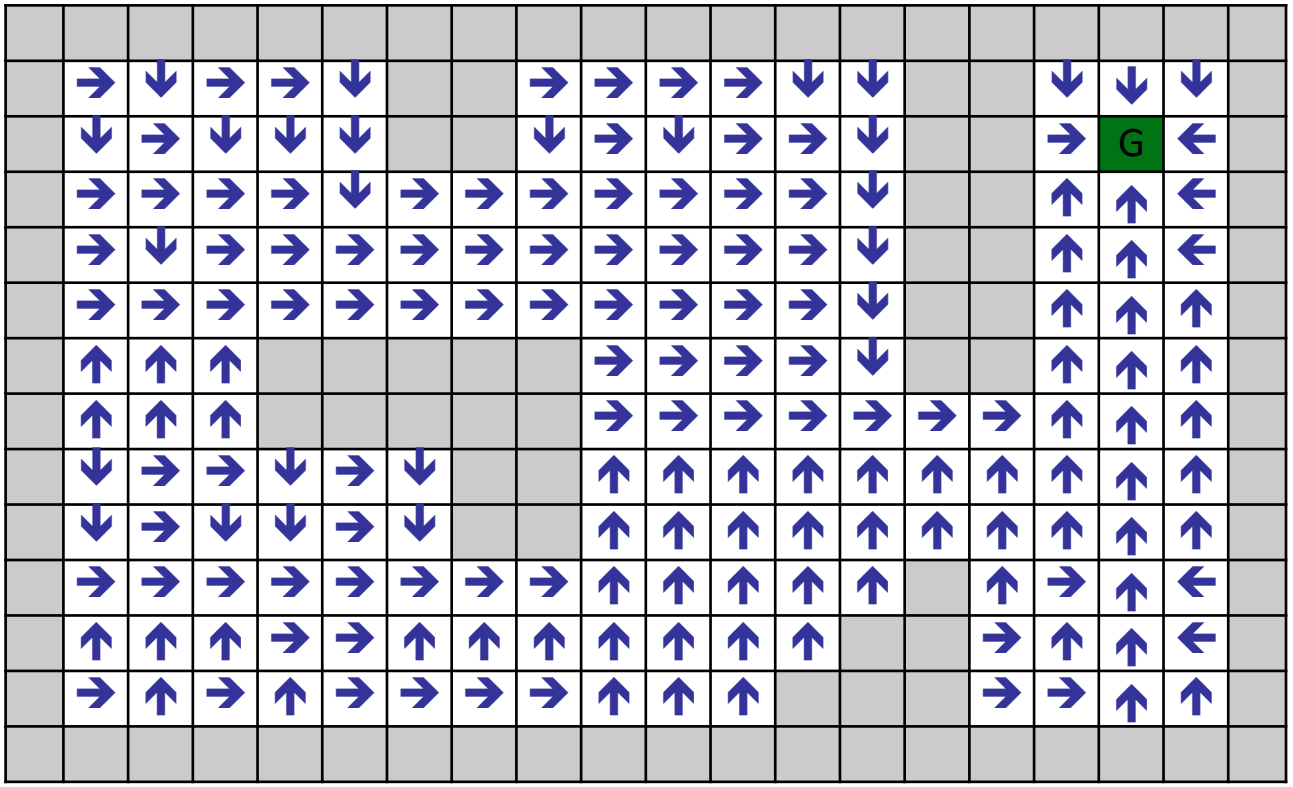
0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000
0,000	-9,286	-9,210	-9,123	-9,025	-9,004	0,000	0,000	-8,512	-8,349	-8,165	-7,962	-7,735	-7,717	0,000	0,000	-0,100	-0,815	-0,946	0,000
0,000	-9,210	-9,123	-9,025	-8,917	-8,915	0,000	0,000	-8,349	-8,166	-7,962	-7,735	-7,484	-7,483	0,000	0,000	1,000	0,000	-0,096	0,000
0,000	-9,123	-9,025	-8,917	-8,796	-8,663	-8,514	-8,349	-8,166	-7,962	-7,735	-7,484	-7,204	-7,204	0,000	0,000	-0,100	-1,090	-1,090	0,000
0,000	-9,025	-8,917	-8,796	-8,663	-8,514	-8,349	-8,166	-7,962	-7,735	-7,484	-7,204	-6,893	-6,893	0,000	0,000	-1,090	-1,981	-1,981	0,000
0,000	-8,917	-8,796	-8,663	-8,514	-8,349	-8,166	-7,962	-7,735	-7,484	-7,204	-6,893	-6,548	-6,548	0,000	0,000	-1,981	-2,783	-2,783	0,000
0,000	-9,025	-8,917	-8,916	0,000	0,000	0,000	0,000	0,000	-7,204	-6,893	-6,548	-6,165	-6,165	0,000	0,000	-2,783	-3,505	-3,505	0,000
0,000	-9,122	-9,025	-9,023	0,000	0,000	0,000	0,000	0,000	-6,893	-6,548	-6,165	-5,738	-5,265	-4,739	-4,154	-3,505	-4,154	-4,153	0,000
0,000	-9,204	-9,122	-9,025	-8,917	-8,796	-8,793	0,000	0,000	-7,204	-6,893	-6,548	-6,165	-5,738	-5,265	-4,739	-4,154	-4,739	-4,738	0,000
0,000	-9,122	-9,025	-8,917	-8,796	-8,663	-8,663	0,000	0,000	-7,484	-7,204	-6,893	-6,548	-6,165	-5,738	-5,265	-4,739	-5,265	-5,254	0,000
0,000	-9,025	-8,917	-8,796	-8,663	-8,514	-8,349	-8,166	-7,962	-7,735	-7,484	-7,204	-6,893	-6,891	0,000	-5,738	-5,265	-5,738	-5,693	0,000
0,000	-9,123	-9,025	-8,917	-8,796	-8,663	-8,514	-8,349	-8,166	-7,962	-7,735	-7,484	-7,482	0,000	0,000	-6,164	-5,738	-6,151	-6,063	0,000
0,000	-9,209	-9,122	-9,025	-8,917	-8,796	-8,663	-8,514	-8,349	-8,166	-7,962	-7,958	0,000	0,000	0,000	-6,539	-6,164	-6,483	-6,390	0,000
0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000

→

0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000
0,000	-9,286	-9,210	-9,123	-9,025	-8,917	0,000	0,000	-8,512	-8,349	-8,165	-7,962	-7,735	-7,484	0,000	0,000	-0,100	1,000	-0,100	0,000
0,000	-9,210	-9,123	-9,025	-8,917	-8,796	0,000	0,000	-8,349	-8,166	-7,962	-7,735	-7,484	-7,204	0,000	0,000	-1,090	0,000	-1,034	0,000
0,000	-9,123	-9,025	-8,917	-8,796	-8,663	-8,514	-8,349	-8,166	-7,962	-7,735	-7,484	-7,204	-6,893	0,000	0,000	-1,981	-1,090	-1,981	0,000
0,000	-9,025	-8,917	-8,796	-8,663	-8,514	-8,349	-8,166	-7,962	-7,735	-7,484	-7,204	-6,893	-6,548	0,000	0,000	-2,783	-1,981	-2,783	0,000
0,000	-9,109	-9,025	-8,917	-8,663	-8,514	-8,349	-8,166	-7,962	-7,484	-7,204	-6,893	-6,548	-6,165	0,000	0,000	-3,505	-2,783	-3,505	0,000
0,000	-9,185	-9,118	-9,022	0,000	0,000	0,000	0,000	0,000	-7,204	-6,893	-6,548	-6,165	-5,738	0,000	0,000	-4,154	-3,505	-4,154	0,000
0,000	-9,172	-9,175	-9,106	0,000	0,000	0,000	0,000	0,000	-7,484	-7,204	-6,893	-6,548	-6,165	-5,738	-5,265	-4,739	-4,154	-4,737	0,000
0,000	-9,204	-9,122	-9,025	-8,917	-8,796	-8,663	0,000	0,000	-7,735	-7,484	-7,204	-6,893	-6,548	-6,165	-5,738	-5,265	-4,739	-5,257	0,000
0,000	-9,122	-9,025	-8,917	-8,796	-8,663	-8,514	0,000	0,000	-7,962	-7,735	-7,484	-7,204	-6,893	-6,164	-6,164	-5,738	-5,265	-5,724	0,000
0,000	-9,190	-9,122	-9,025	-8,917	-8,796	-8,663	-8,514	-8,349	-8,166	-7,962	-7,735	-7,483	-6,889	0,000	-6,331	-6,149	-5,738	-6,062	0,000
0,000	-9,243	-9,190	-9,100	-9,020	-8,914	-8,796	-8,662	-8,514	-8,349	-8,165	-7,962	-7,482	0,000	0,000	-6,376	-6,420	-6,162	-6,306	0,000
0,000	-9,244	-9,158	-9,090	-9,013	-8,903	-8,783	-8,649	-8,504	-8,343	-8,161	-7,949	0,000	0,000	0,000	-6,714	-6,477	-6,113	-6,366	0,000
0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000

Example: Path planning

- Optimal policy extracted from Q function



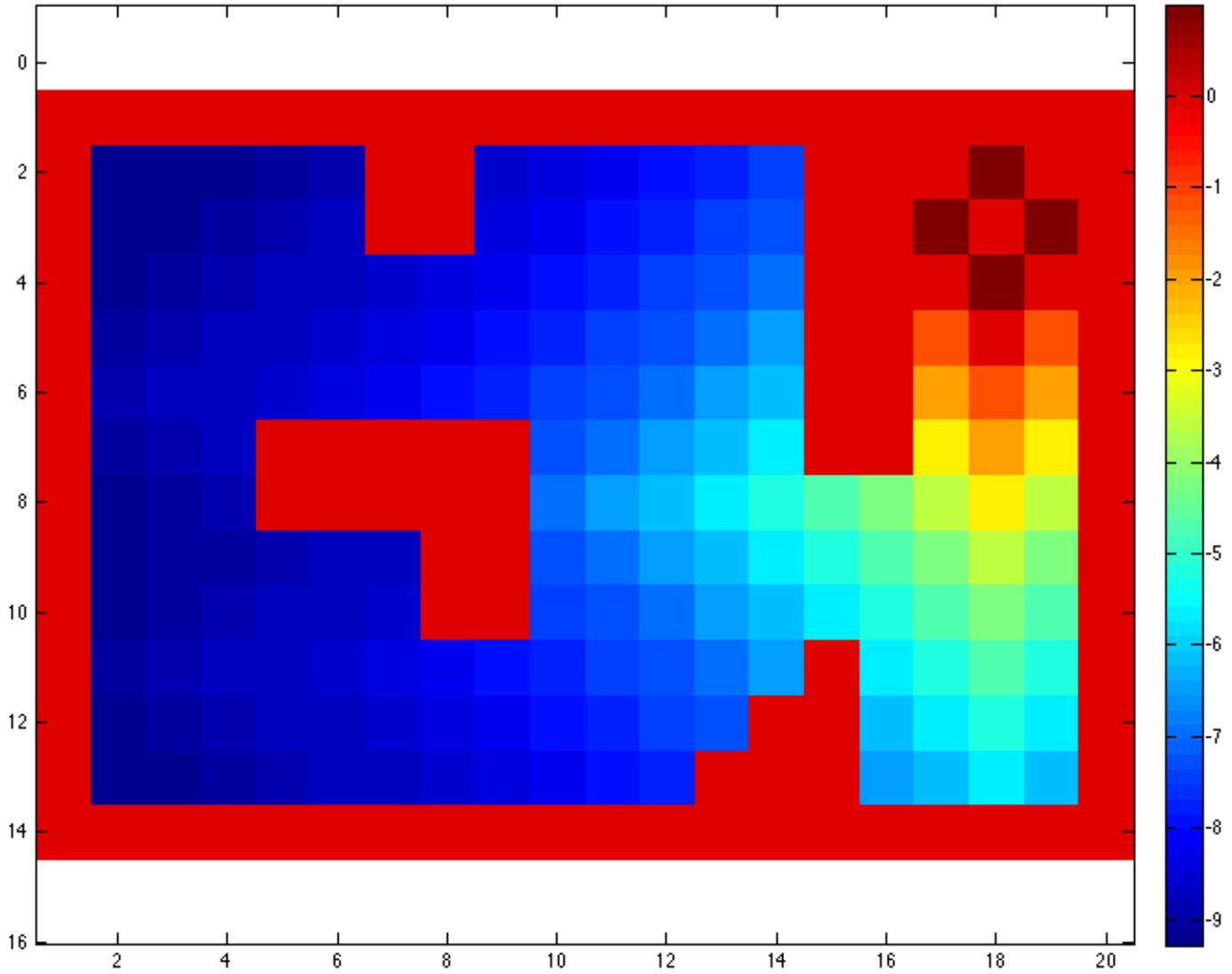
Example: Path planning

- V function extracted from Q

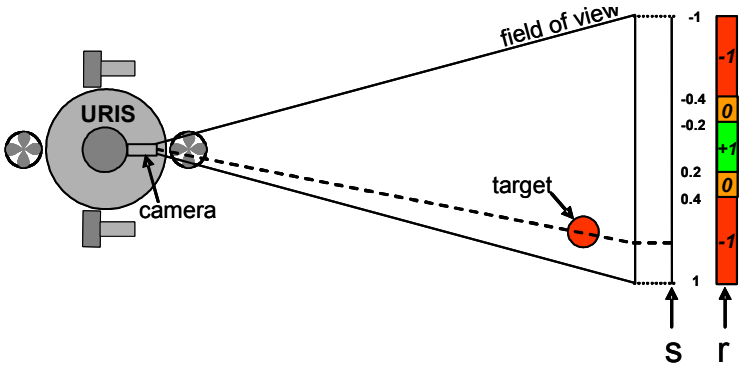
0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000
0,000	-9,286	-9,210	-9,123	-9,025	-8,917	0,000	0,000	-8,512	-8,349	-8,165	-7,962	-7,735	-7,484	0,000	0,000	-0,100	1,000	-0,100	0,000
0,000	-9,210	-9,123	-9,025	-8,917	-8,796	0,000	0,000	-8,349	-8,166	-7,962	-7,735	-7,484	-7,204	0,000	0,000	1,000	0,000	1,000	0,000
0,000	-9,123	-9,025	-8,917	-8,796	-8,663	-8,514	-8,349	-8,166	-7,962	-7,735	-7,484	-7,204	-6,893	0,000	0,000	-0,100	1,000	-0,100	0,000
0,000	-9,025	-8,917	-8,796	-8,663	-8,514	-8,349	-8,166	-7,962	-7,735	-7,484	-7,204	-6,893	-6,548	0,000	0,000	-1,090	-0,100	-1,090	0,000
0,000	-8,917	-8,796	-8,663	-8,514	-8,349	-8,166	-7,962	-7,735	-7,484	-7,204	-6,893	-6,548	-6,165	0,000	0,000	-1,981	-1,090	-1,981	0,000
0,000	-9,025	-8,917	-8,796	0,000	0,000	0,000	0,000	0,000	-7,204	-6,893	-6,548	-6,165	-5,738	0,000	0,000	-2,783	-1,981	-2,783	0,000
0,000	-9,122	-9,025	-8,917	0,000	0,000	0,000	0,000	0,000	-6,893	-6,548	-6,165	-5,738	-5,265	-4,739	-4,154	-3,505	-2,783	-3,505	0,000
0,000	-9,204	-9,122	-9,025	-8,917	-8,796	-8,663	0,000	0,000	-7,204	-6,893	-6,548	-6,165	-5,738	-5,265	-4,739	-4,154	-3,505	-4,154	0,000
0,000	-9,122	-9,025	-8,917	-8,796	-8,663	-8,514	0,000	0,000	-7,484	-7,204	-6,893	-6,548	-6,165	-5,738	-5,265	-4,739	-4,154	-4,739	0,000
0,000	-9,025	-8,917	-8,796	-8,663	-8,514	-8,349	-8,166	-7,962	-7,735	-7,484	-7,204	-6,893	-6,548	0,000	-5,738	-5,265	-4,739	-5,265	0,000
0,000	-9,123	-9,025	-8,917	-8,796	-8,663	-8,514	-8,349	-8,166	-7,962	-7,735	-7,484	-7,204	0,000	0,000	-6,164	-5,738	-5,265	-5,738	0,000
0,000	-9,209	-9,122	-9,025	-8,917	-8,796	-8,663	-8,514	-8,349	-8,166	-7,962	-7,735	0,000	0,000	0,000	-6,539	-6,164	-5,738	-6,164	0,000
0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000

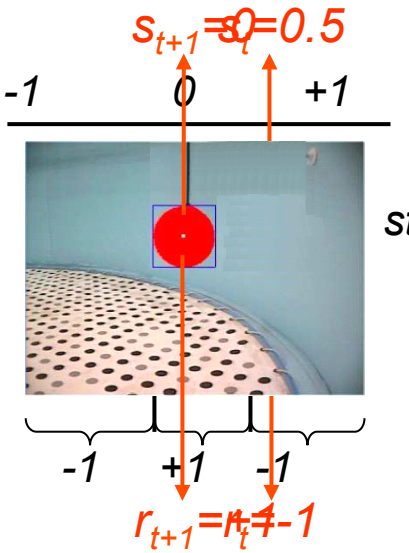
Example: Path planning

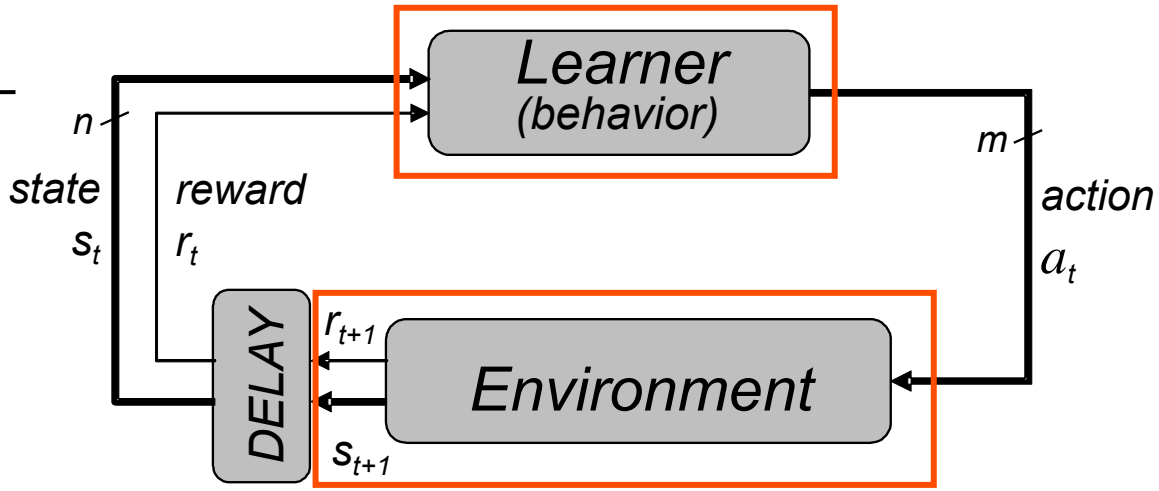
- V function

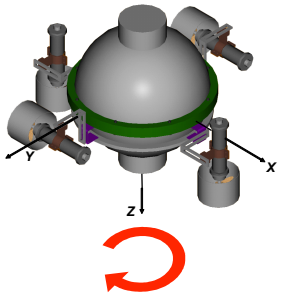


Case study: behaviour learning







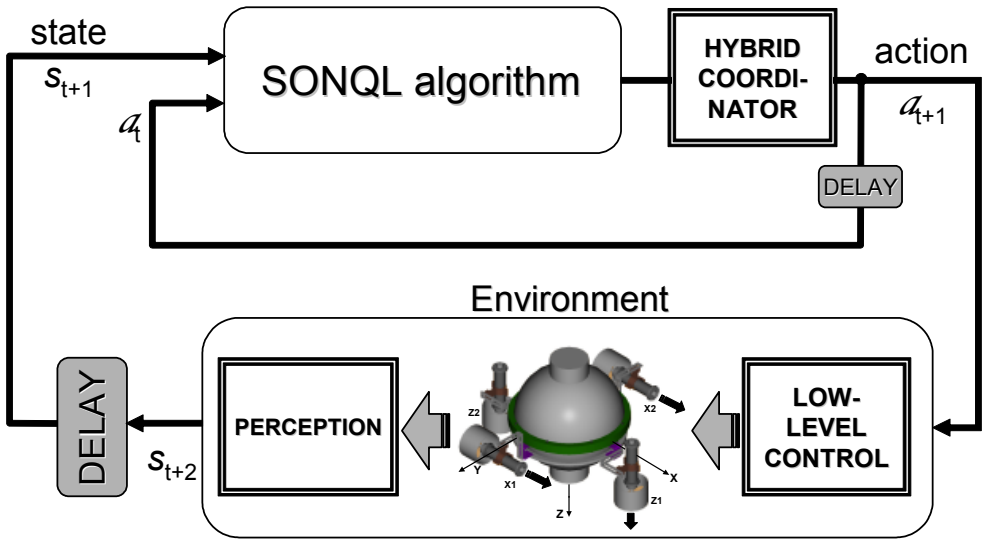


Generalization problem

Autonomous Robots

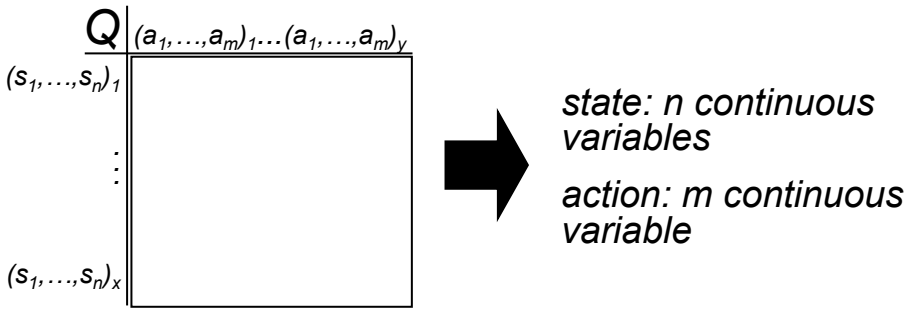
Generalization problem

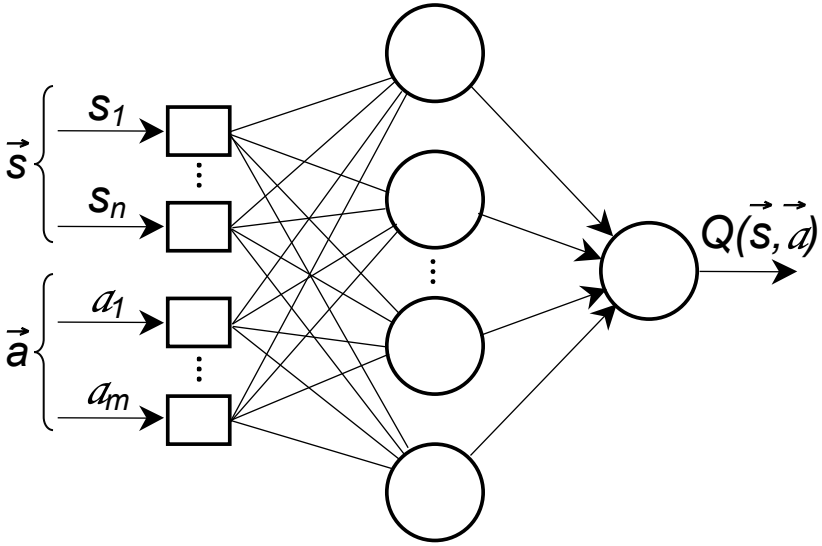
Semi Online Neural Q_learning



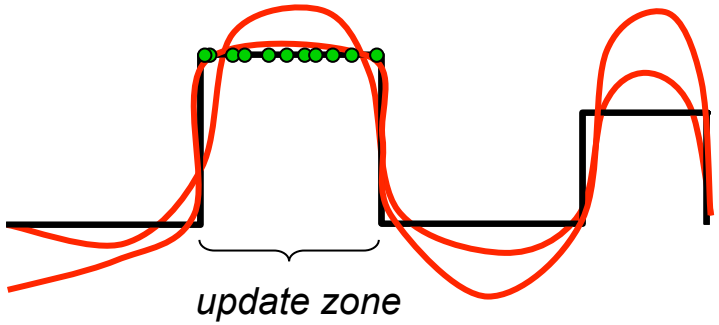
- SONQL goal: **RL algorithm for learning in real-time robotics tasks**
- Components:
  - online, off-policy → Q_learning
  - high generalization capability → Neural Networks
  - Interference Problem → Learning Samples Database

Semi Online Neural Q_learning



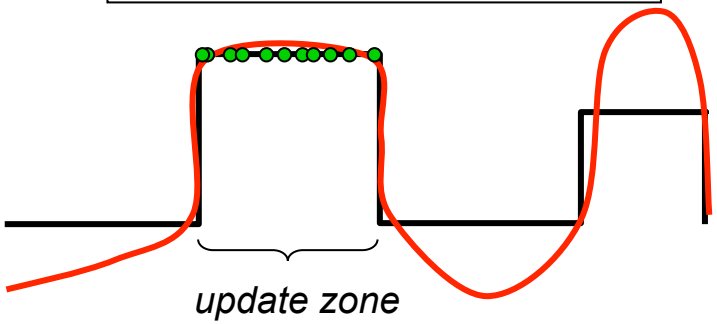


- **NN features:**
  - Multilayer
  - Back-propagation
  - Hidden layers: 1 or 2, hyperbolic tangent function
  - Output layer: linear function
- **Interference problem:**
  - Non-local function approximation

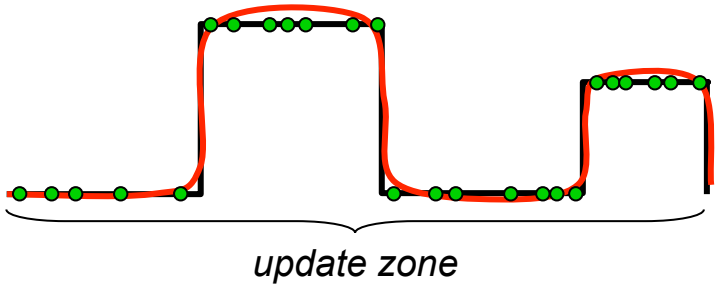


Semi Online Neural Q_learning

Updating the NN with the current samples



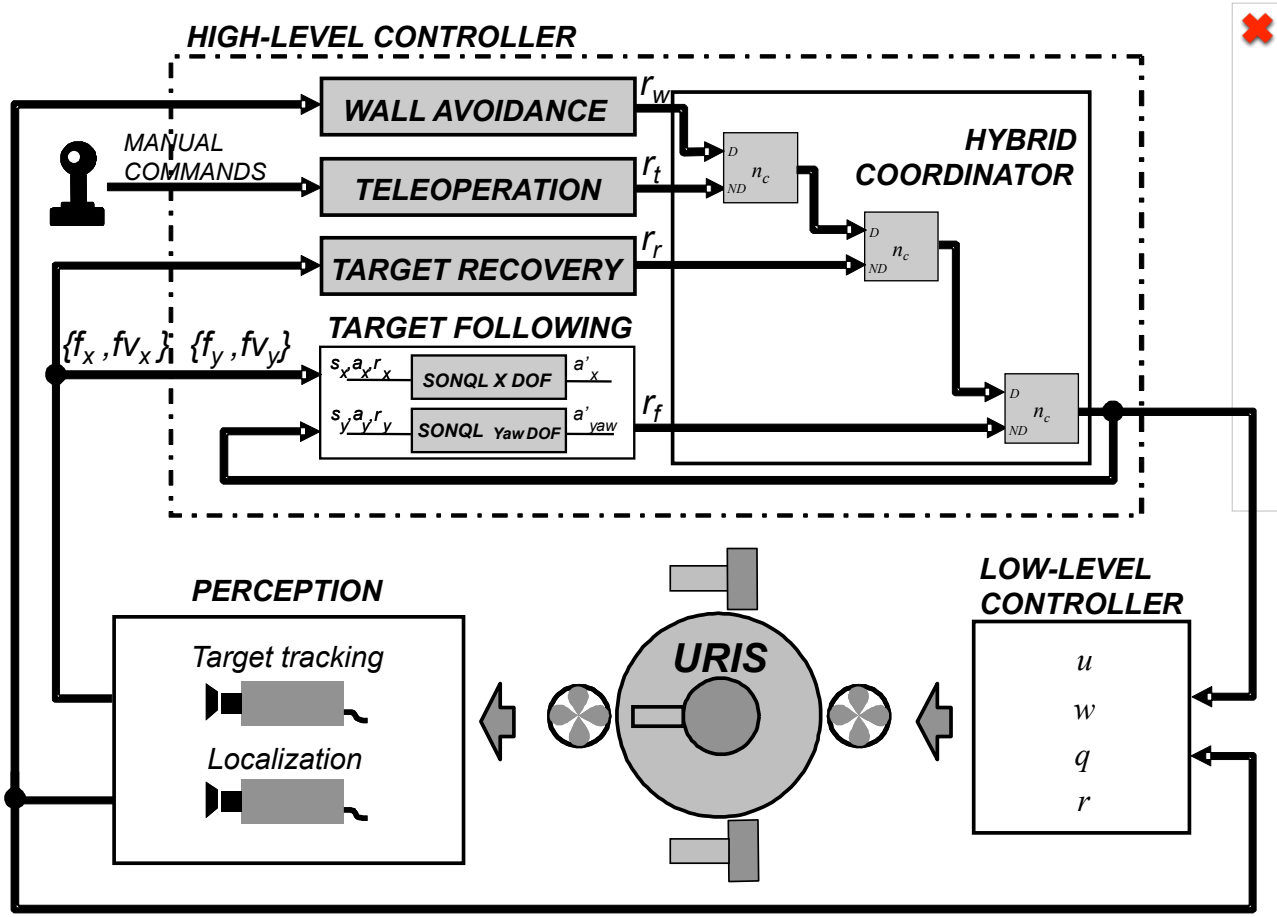
Updating the NN with a representative set of samples



- **Interference problem → Uniformly NN updating**

- Database of Learning Samples
  {st,at,rt+1,st+1}

Example SONQL: Behaviour learning

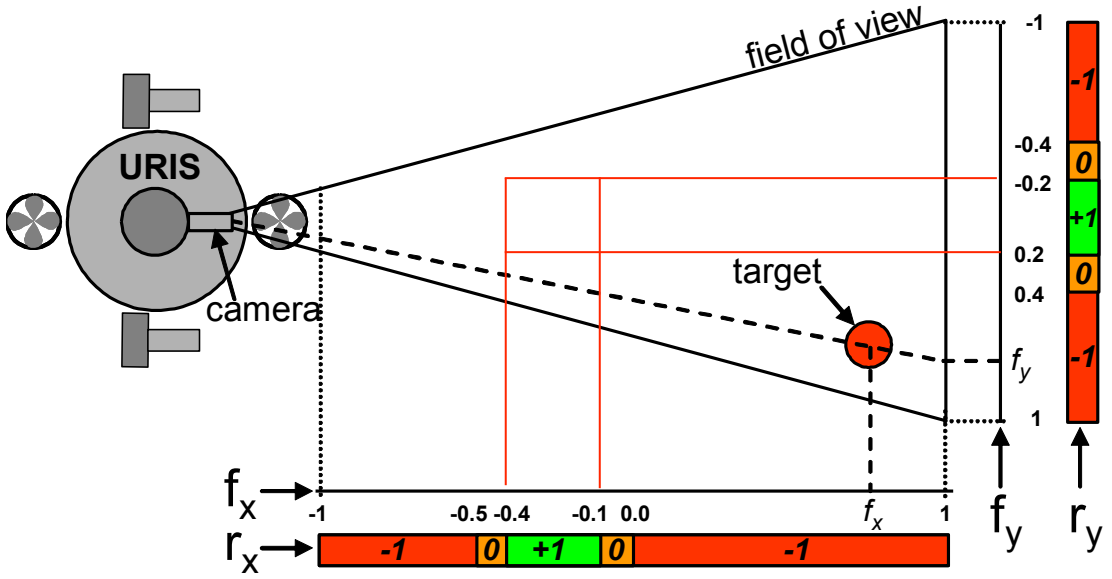


- Task: ***"To follow a target avoiding obstacles"***

Example SONQL: Behaviour learning

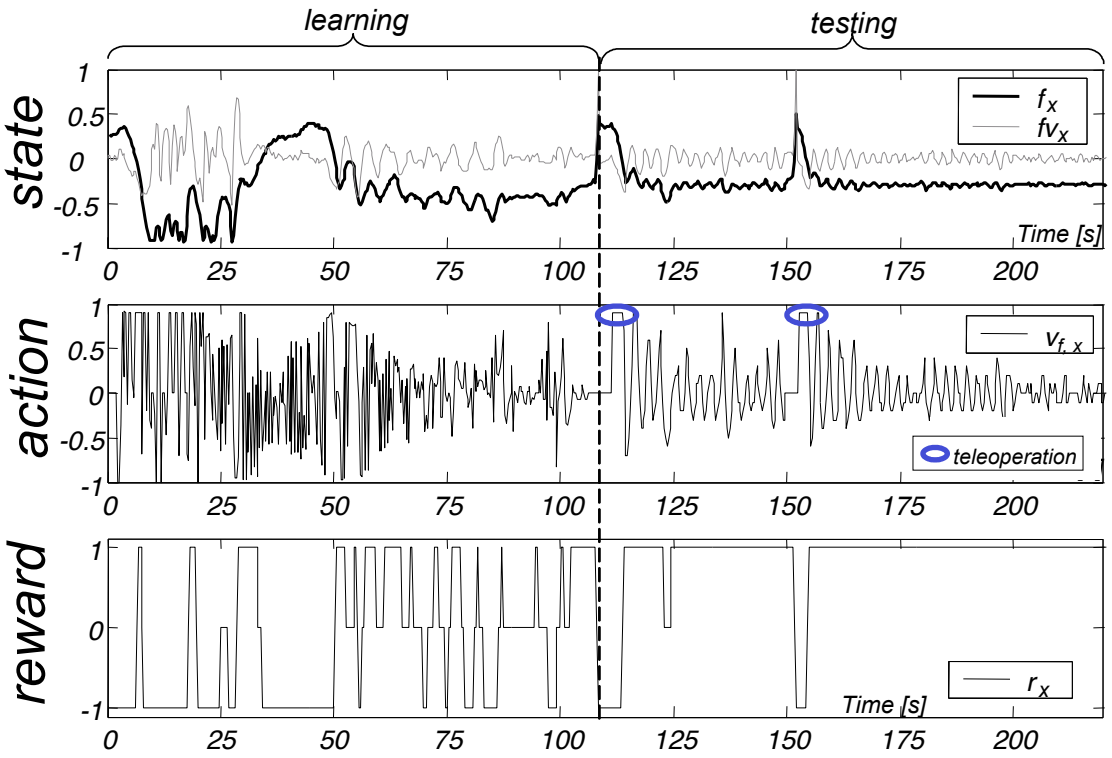
Target Following behavior with SONQL

- *Environment State*
  - *X DOF: f_x ,fv_x*
  - *Y DOF: f_y ,fv_y*
- *Reinforcement Function*
  - *X DOF: r_x*
  - *Y DOF: r_y*



Example SONQL: Behaviour learning

Learning the X DOF



action

X DOF

$\pi(s)=a$

state

action

Yaw DOF

$\pi(s)=a$

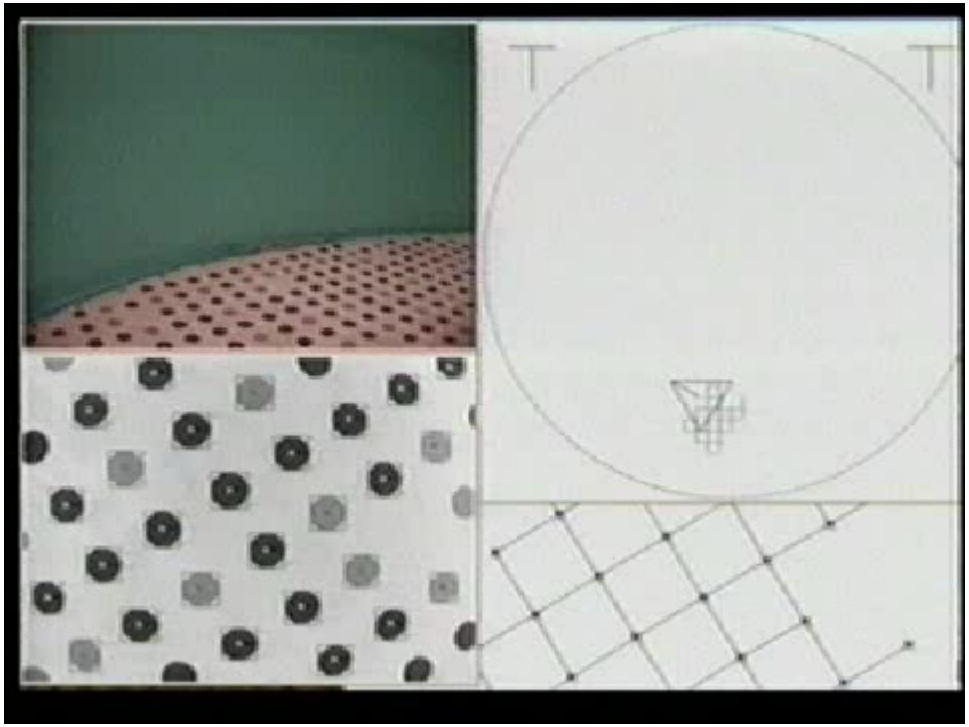
state

Example SONQL: Behaviour learning

Learning the X DOF

Learning the Yaw DOF

Example SONQL: Behaviour learning



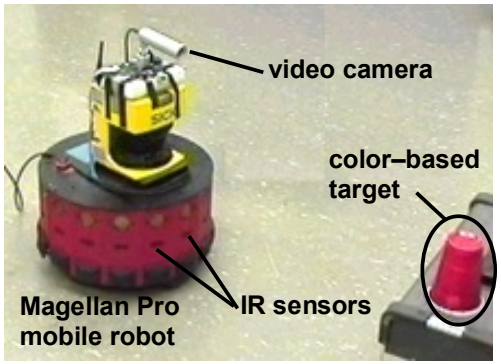
Example SONQL: Behaviour learning

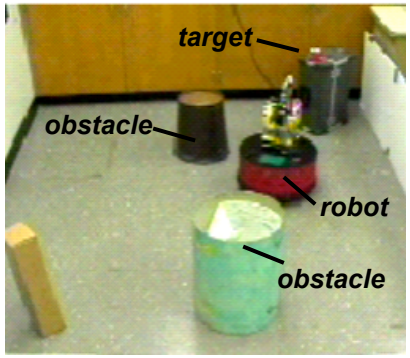
Efficient Learning of Reactive Robot Behaviors with a Neural-Q_learning approach

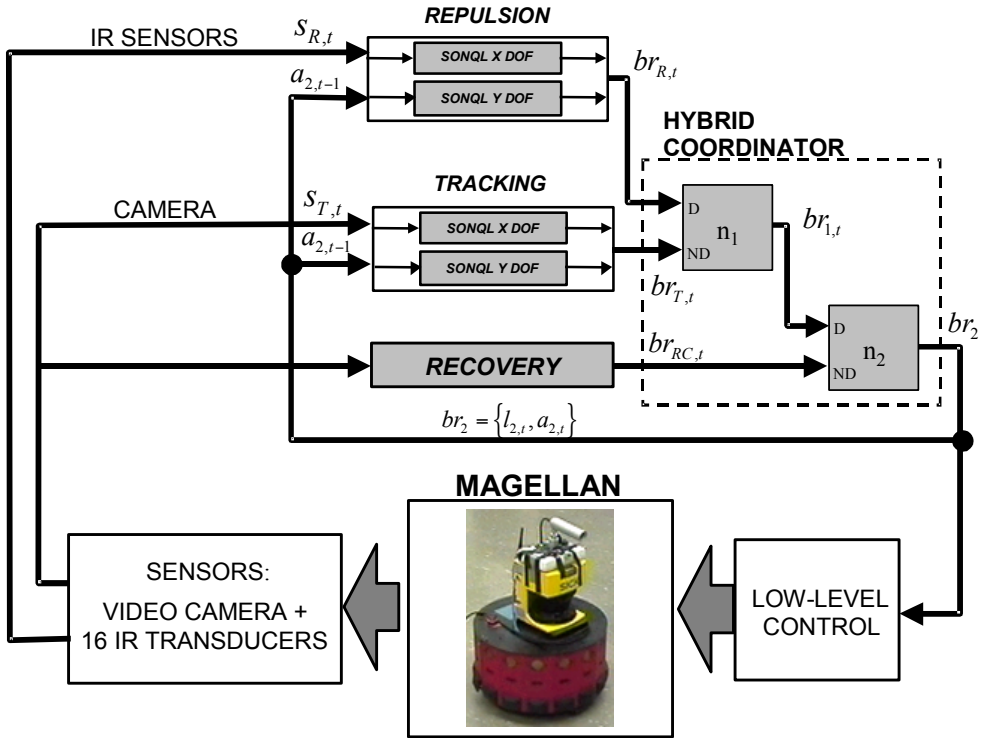
University of Girona
Spain

IROS 2002, Switzerland

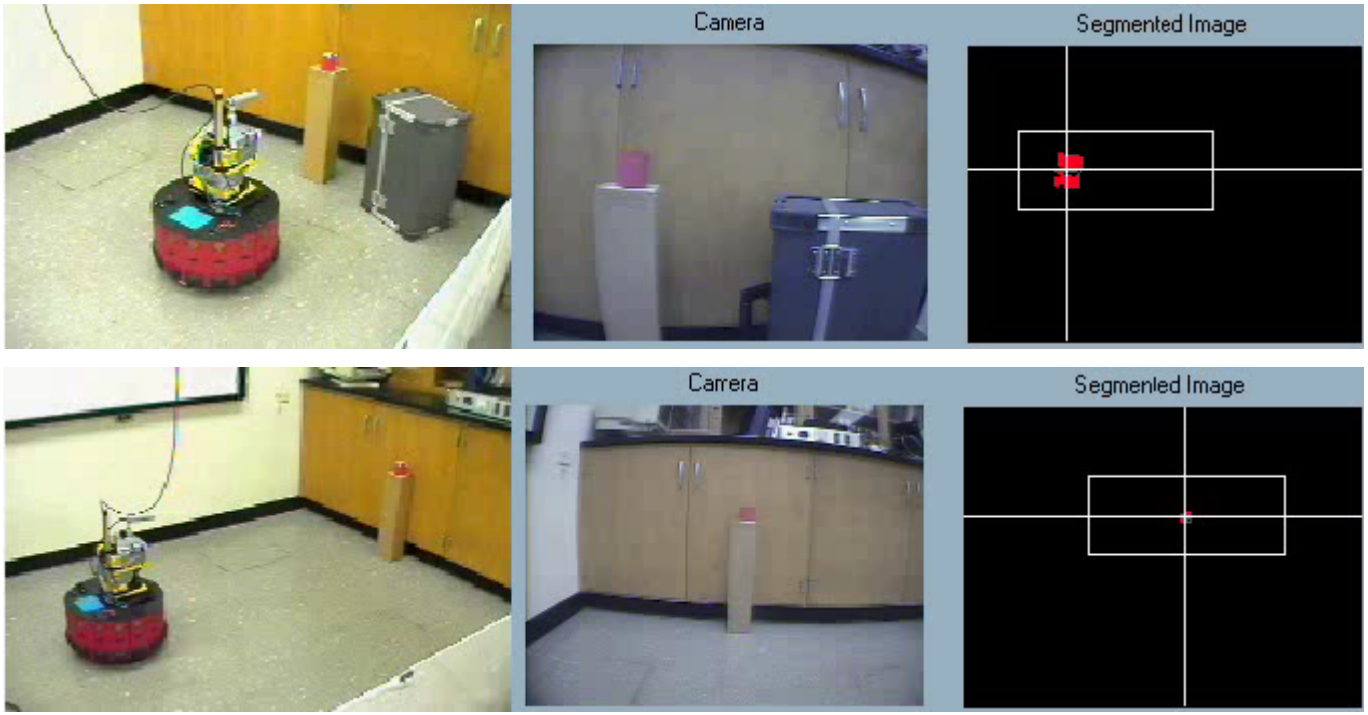
Example SONQL: Behaviour learning 2







Example SONQL: Behaviour learning 2



Example SONQL: Behaviour learning 2