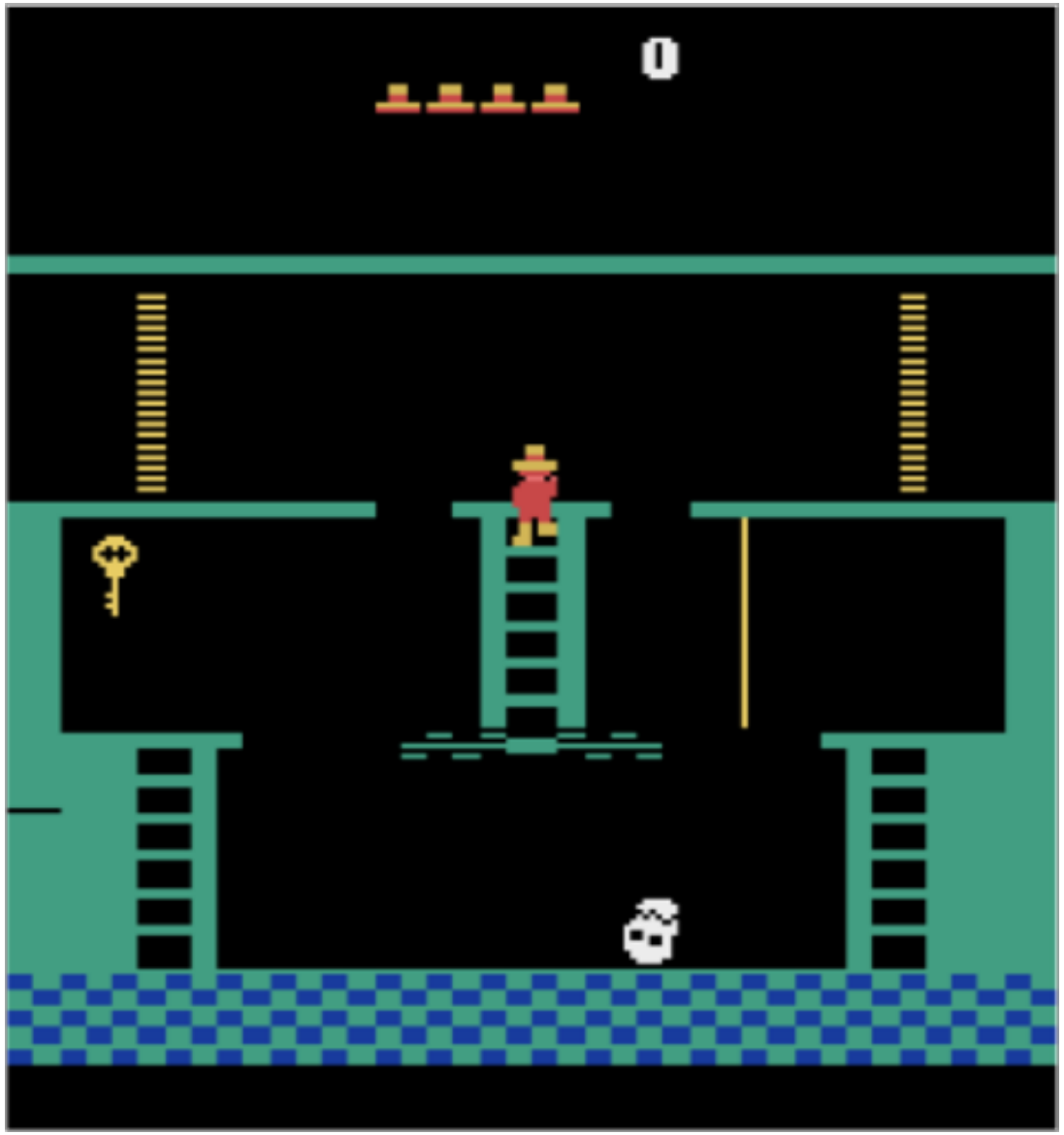
ETH zürich

Hierarchical Reinforcement Learning

Francesco Saverio Varini

Motivation: Why Hierarchical Reinforcement Learning?

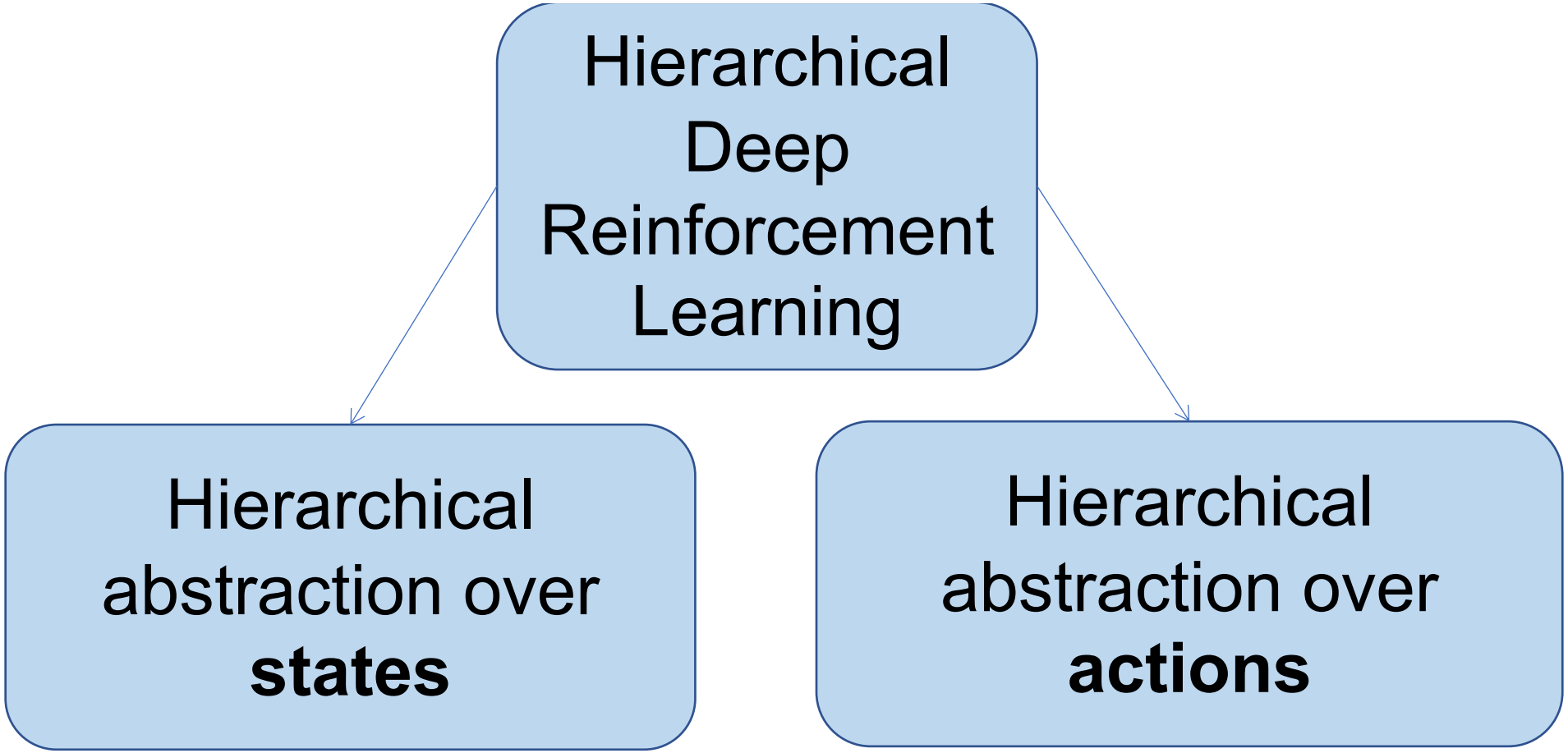
- **Sparse** and **long-term** rewards!
- Not **efficient** long-term planning!
- Training is **inefficient**
- **Local minima**



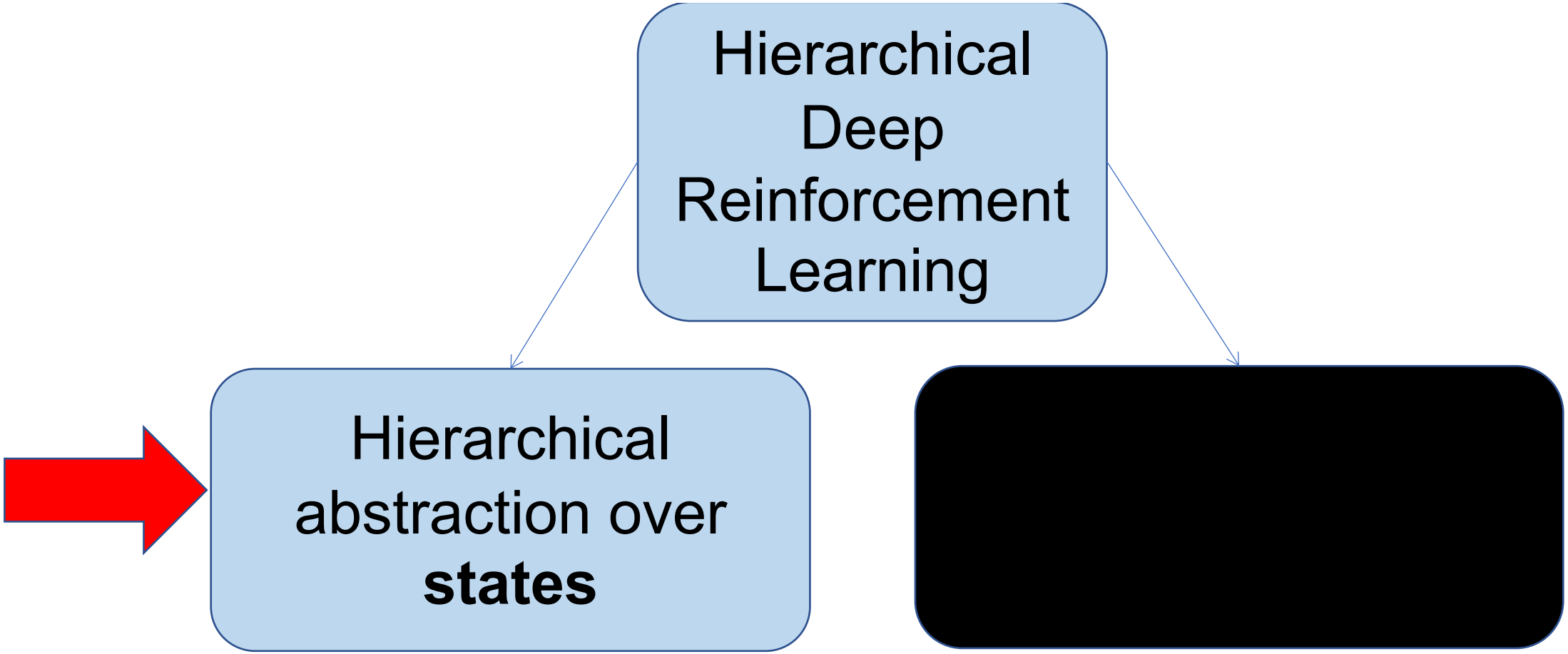
1) **Hierarchical Deep Reinforcement Learning: Integrating Temporal Abstraction and Intrinsic Motivation** (Tejas D., Karthik et al.) (April 2016)

2) **The Option-Critic Architecture** (Pierre-Luc Bacon, Jean Harb, Doina Precup) (September 2016)

Two papers – Two hierarchical abstractions



Two papers – Two hierarchical abstractions

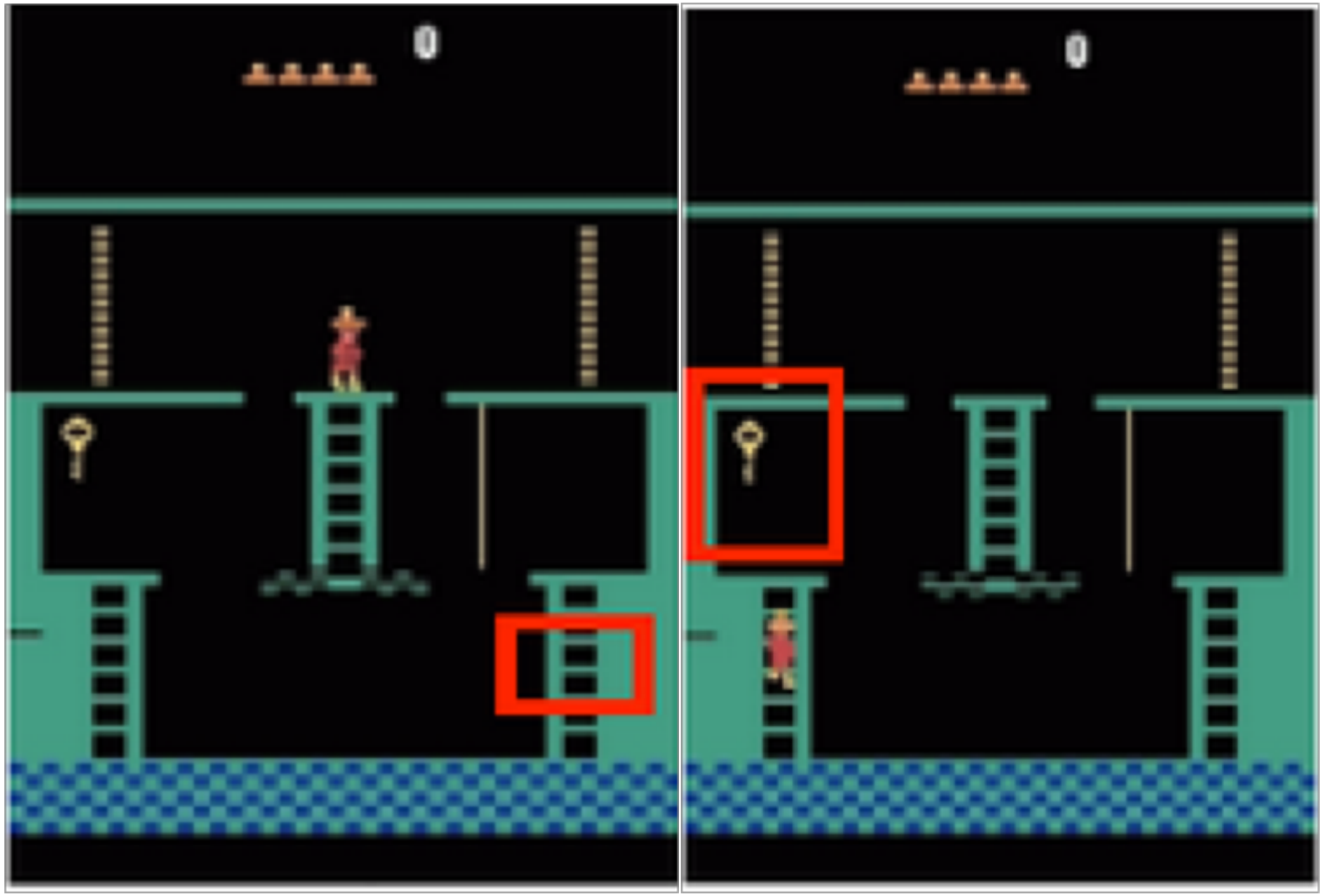


Hierarchical abstraction over states

Sparse and **long-term** rewards!

Solution:

Introduce **intrinsic motivation**
(sub-goals + sub-rewards)



Hierarchical abstraction over states

- «Hierarchical Value functions»

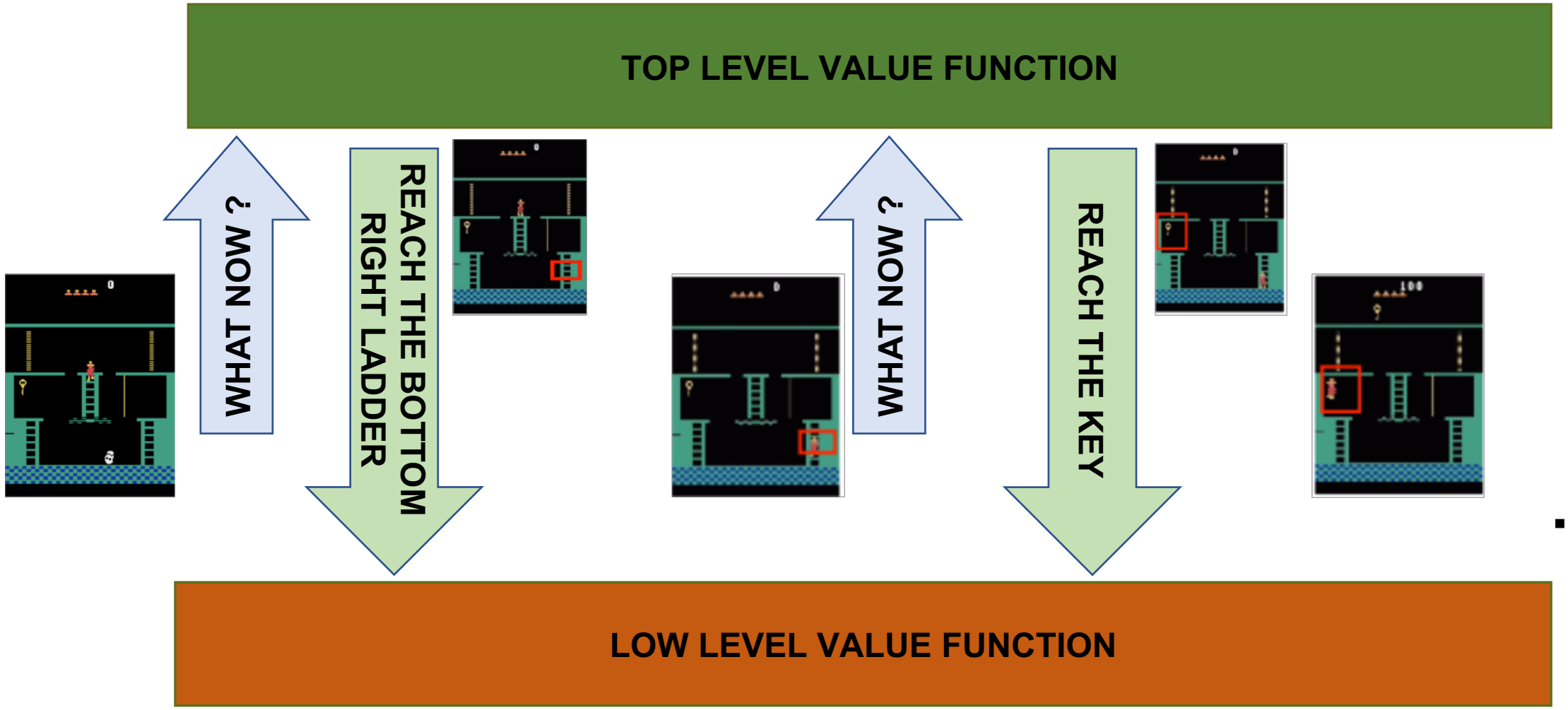
TOP LEVEL VALUE FUNCTION

policy over intrinsic **sub-goals**

LOW LEVEL VALUE FUNCTION

policy over **atomic actions (w.r.t the sub-goal)**

Hierarchical abstraction over states



Hierarchical Value Function

Meta-Controller

Input : state *s*

$$Q_2^*(s, g) = \max_{\pi_g} \mathrm{E}[\sum_{t'=t}^{t+N} f_{t'} + \gamma \max_{g'} Q_2^*(s_{t+N}, g') \mid s_t = s, g_t = g, \pi_g]$$

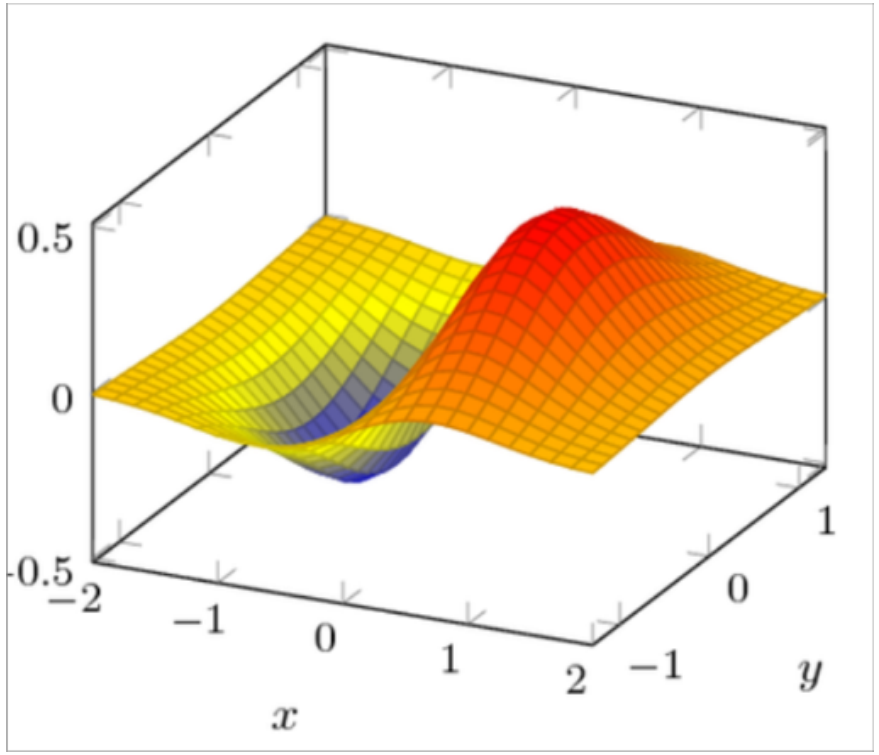
Controller (as standard DQN + sub-goal)

Input : state *s* + sub-goal *g*

$$Q_1^*(s, a; g) = \max_{\pi_{ag}} \mathrm{E}[r_t + \gamma \max_{a_{t+1}} Q_1^*(s_{t+1}, a_{t+1}; g) \mid s_t = s, a_t = a, g_t = g, \pi_{ag}]$$

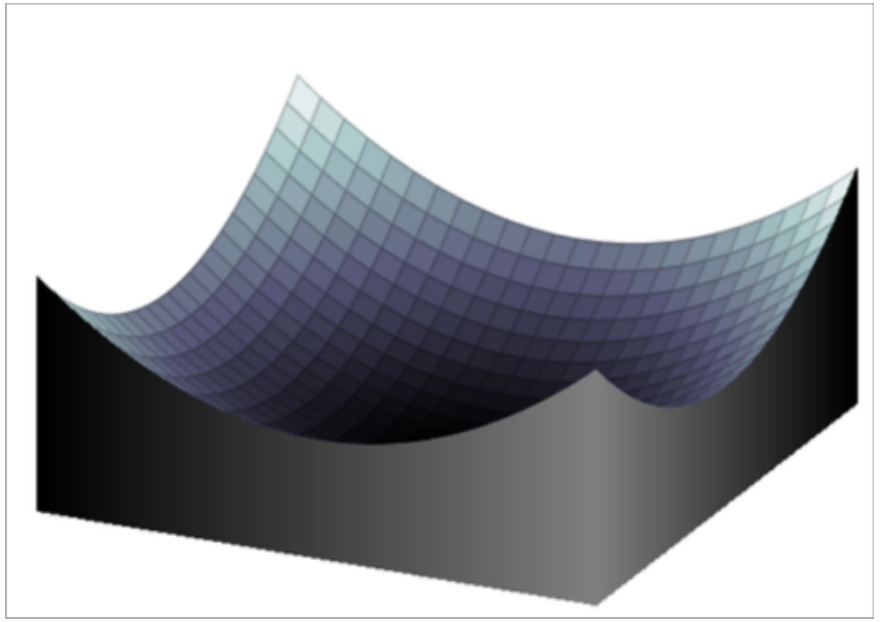
Hierarchical abstraction over states

- **Complicated** original solution space (meta-controller)

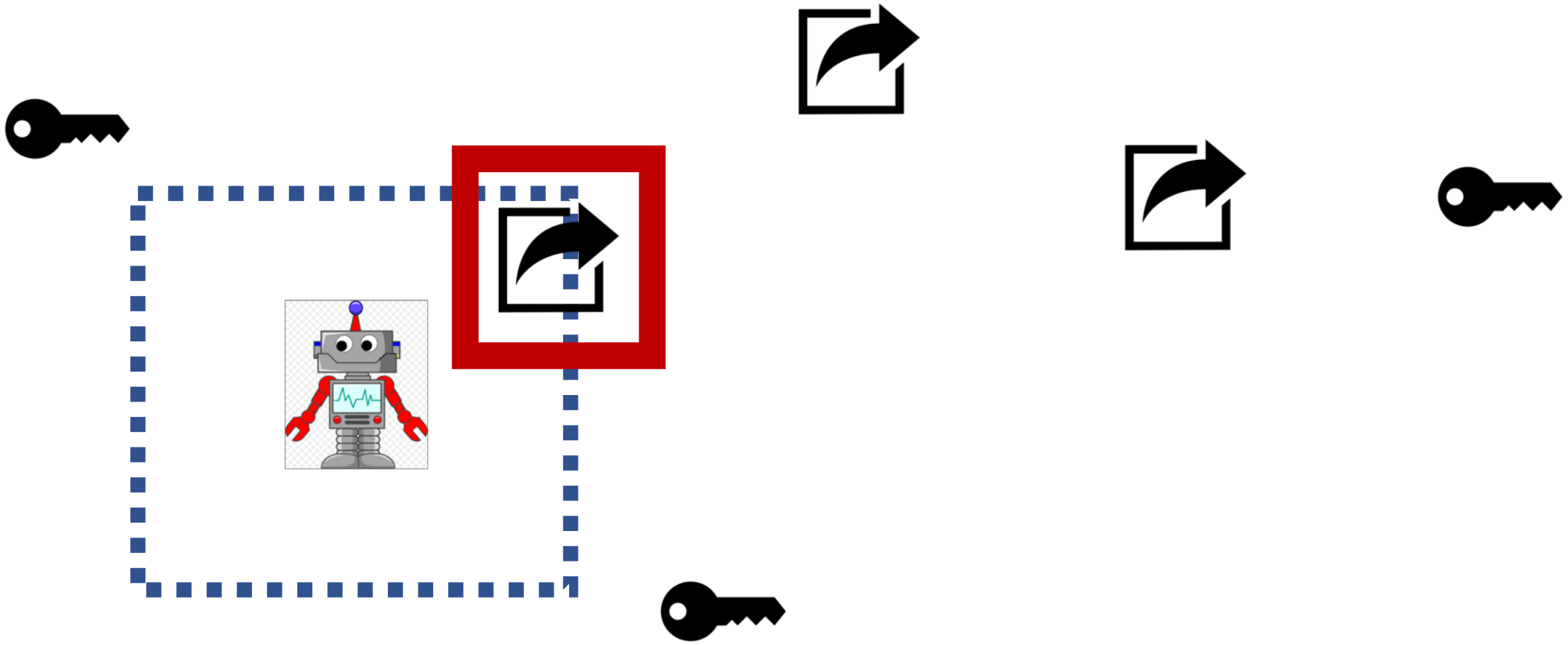


GOAL

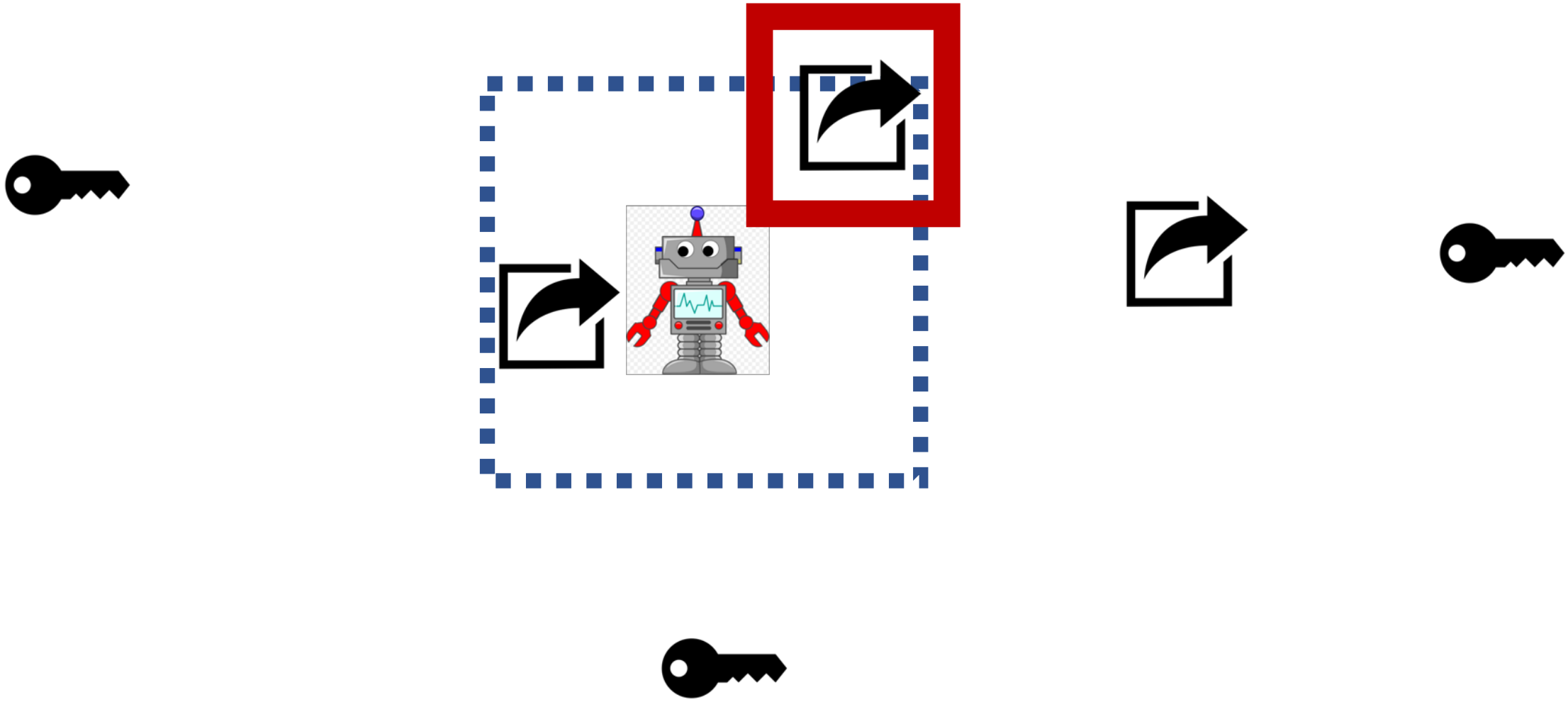
Shape the solution space towards the specific goal (controller)



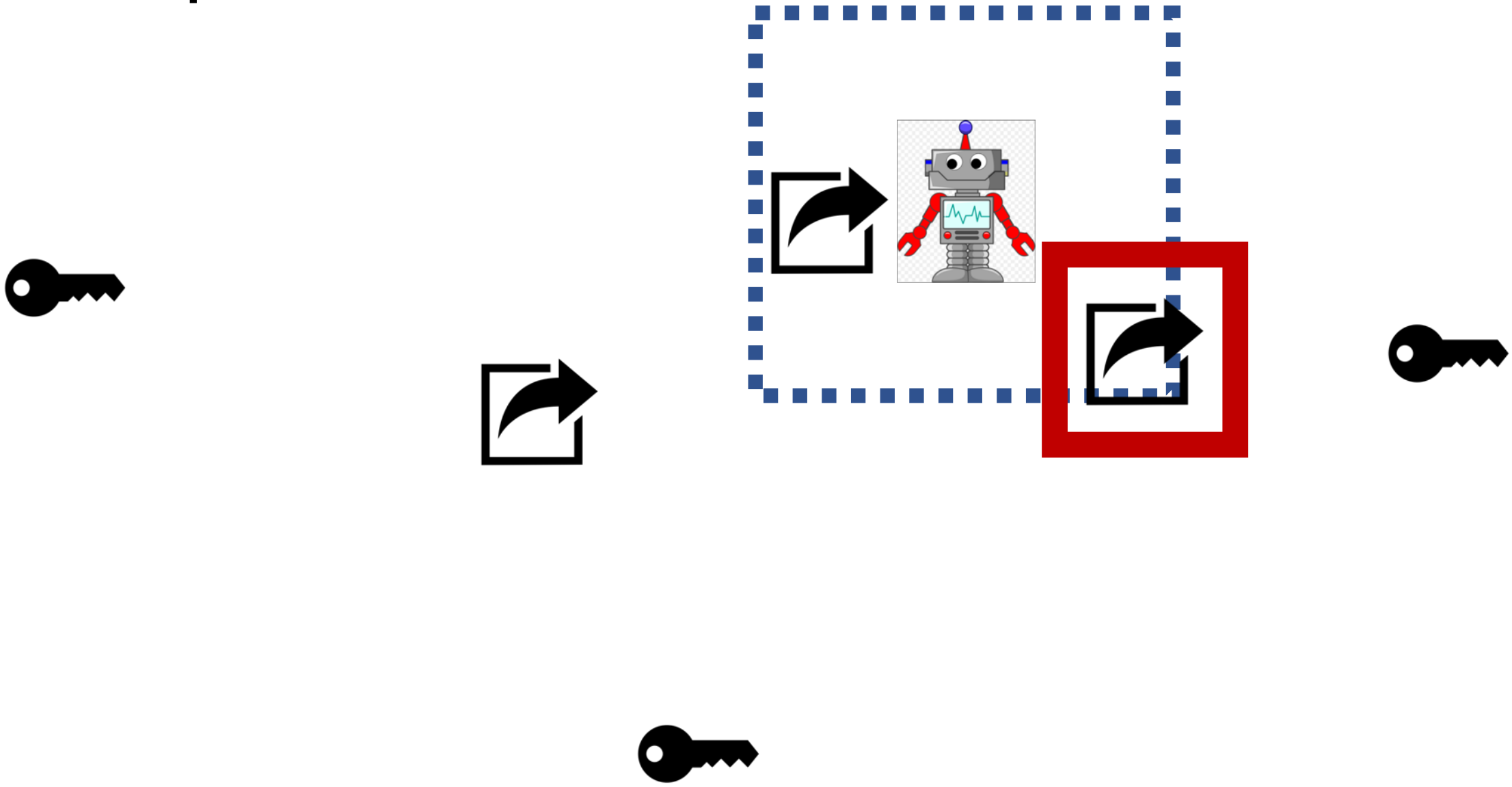
Example



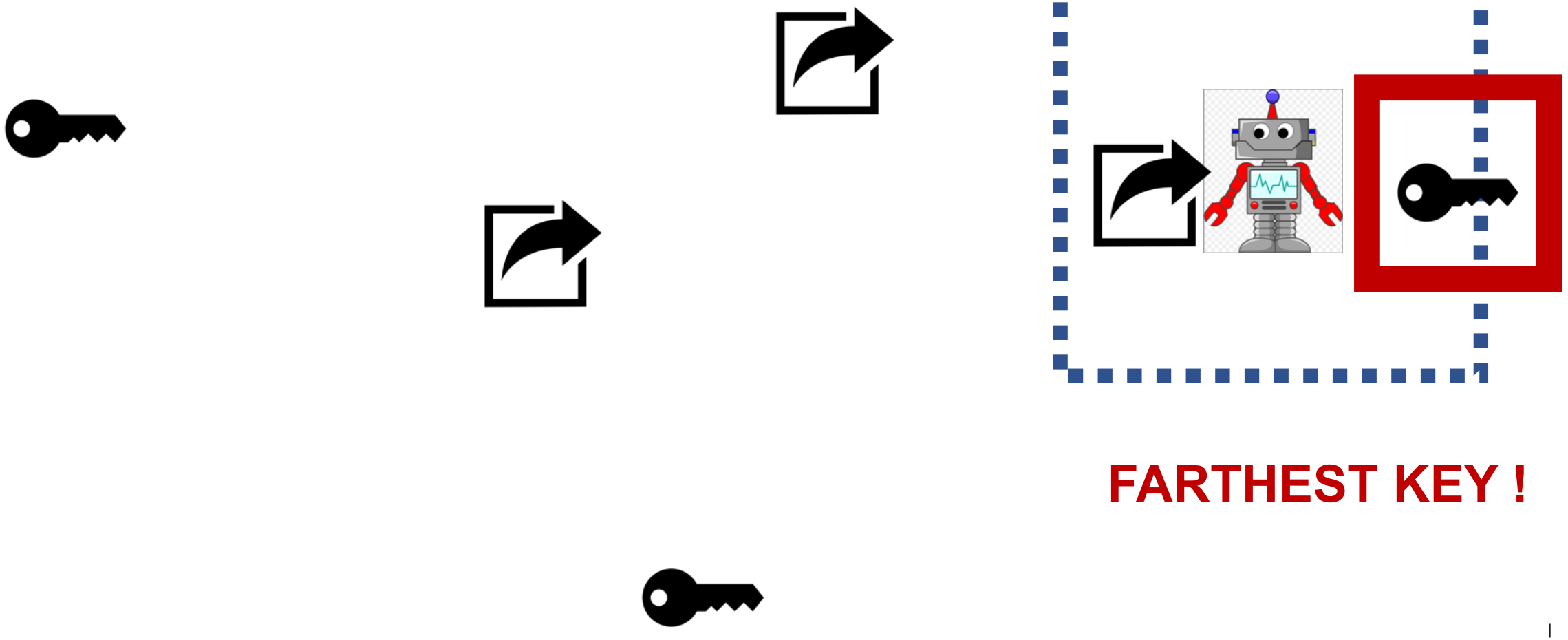
Example



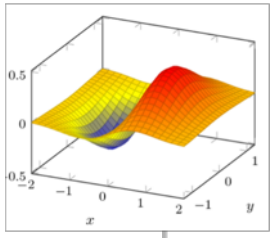
Example

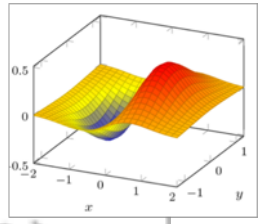


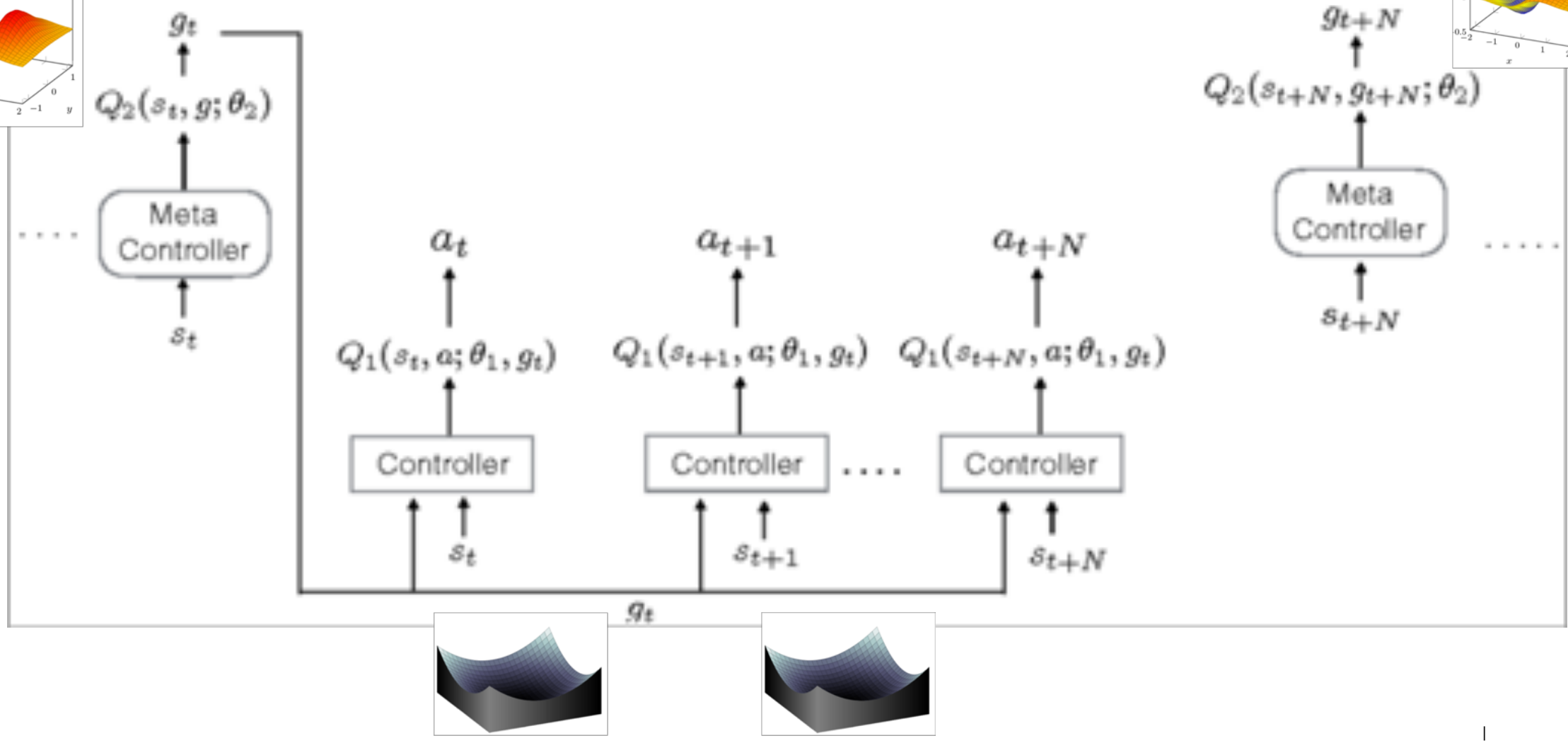
Example



Hierarchical abstraction over states

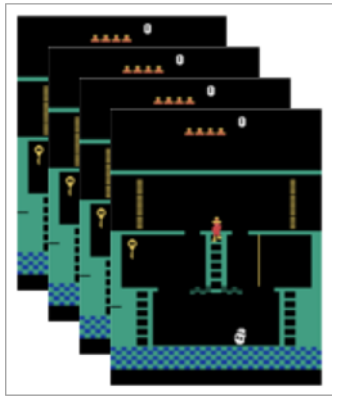


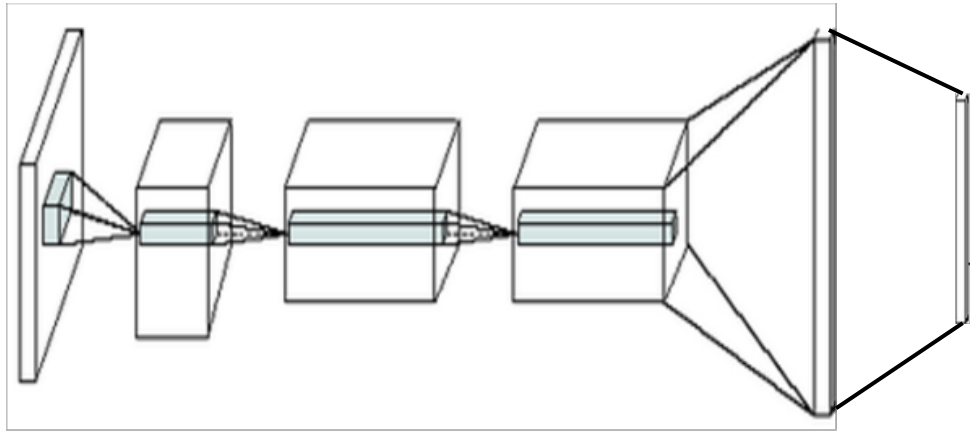




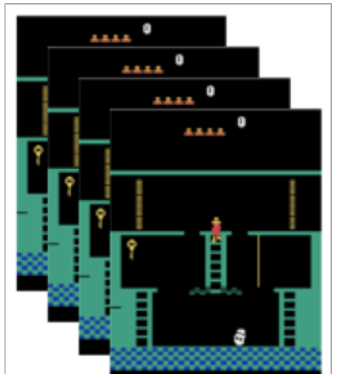
Hierarchical abstraction over states

- The Meta-controller (CNN + dense)

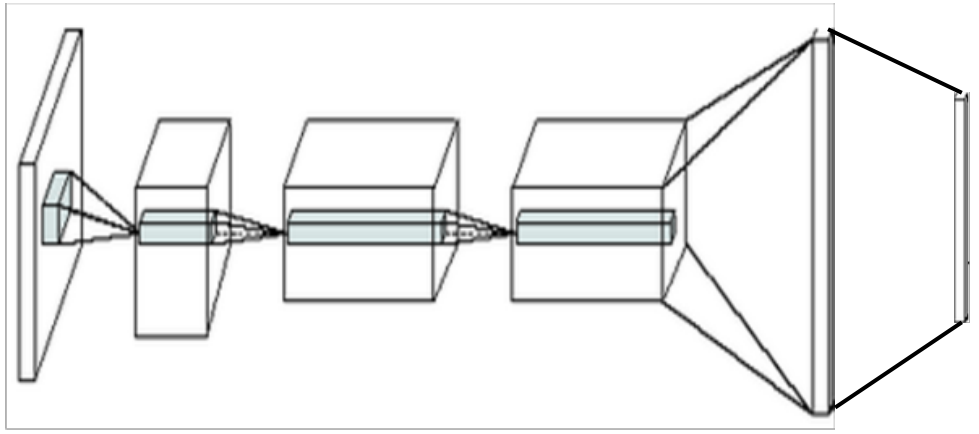




- The Controller (CNN + dense)

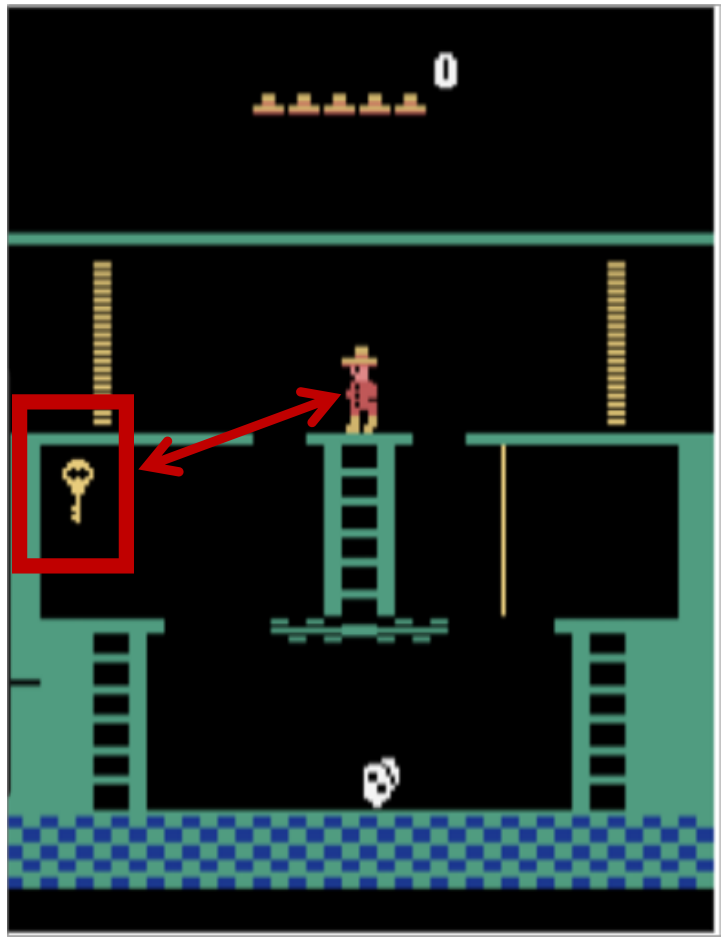


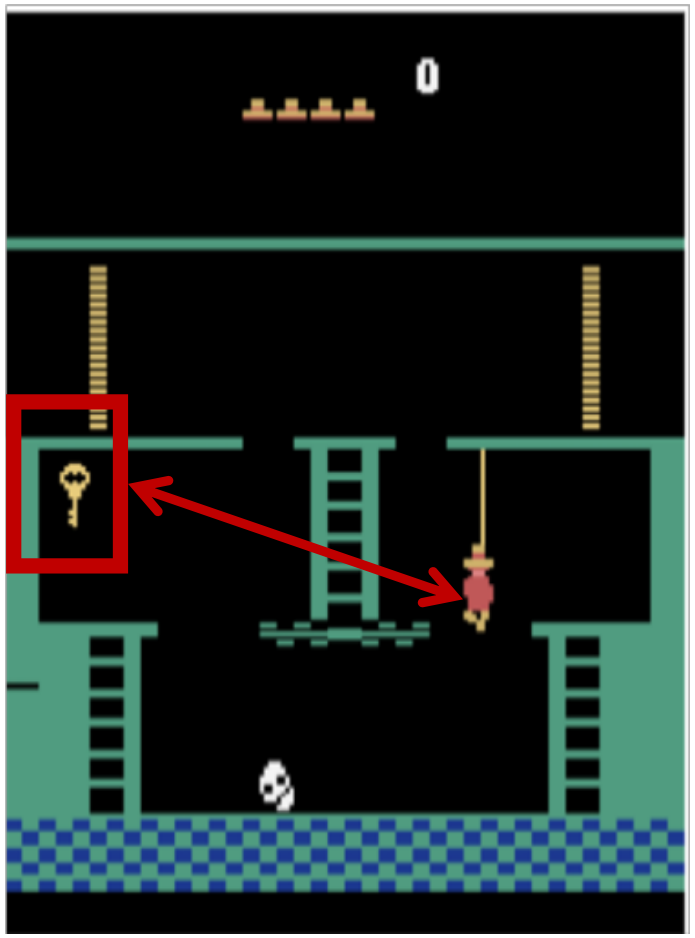
+

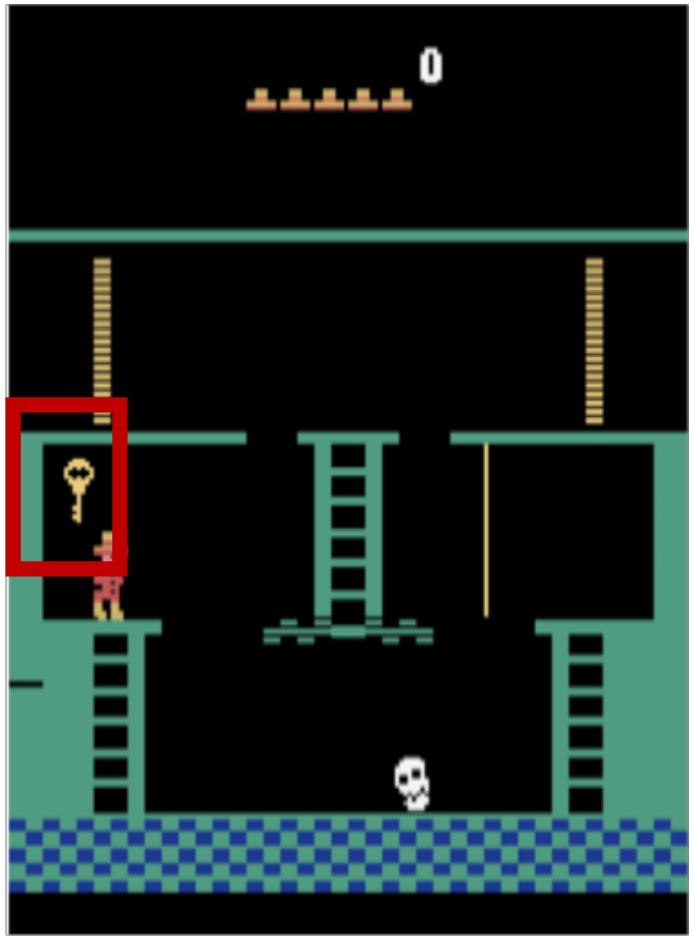


Hierarchical abstraction over states

- The **Internal Critic** (<$entity_1$, *relation*, $entity_2$>)







Training

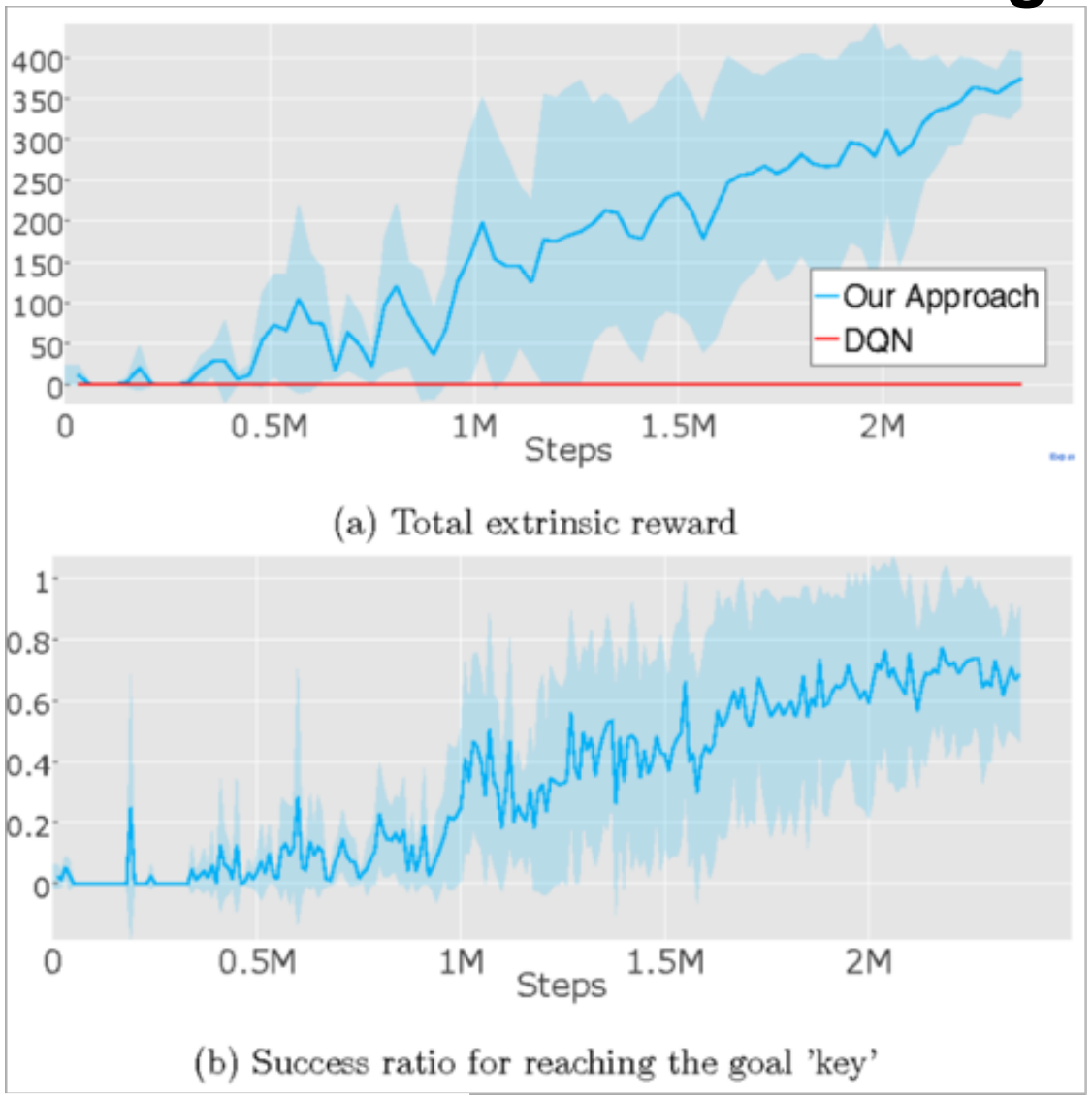
- The algorithm

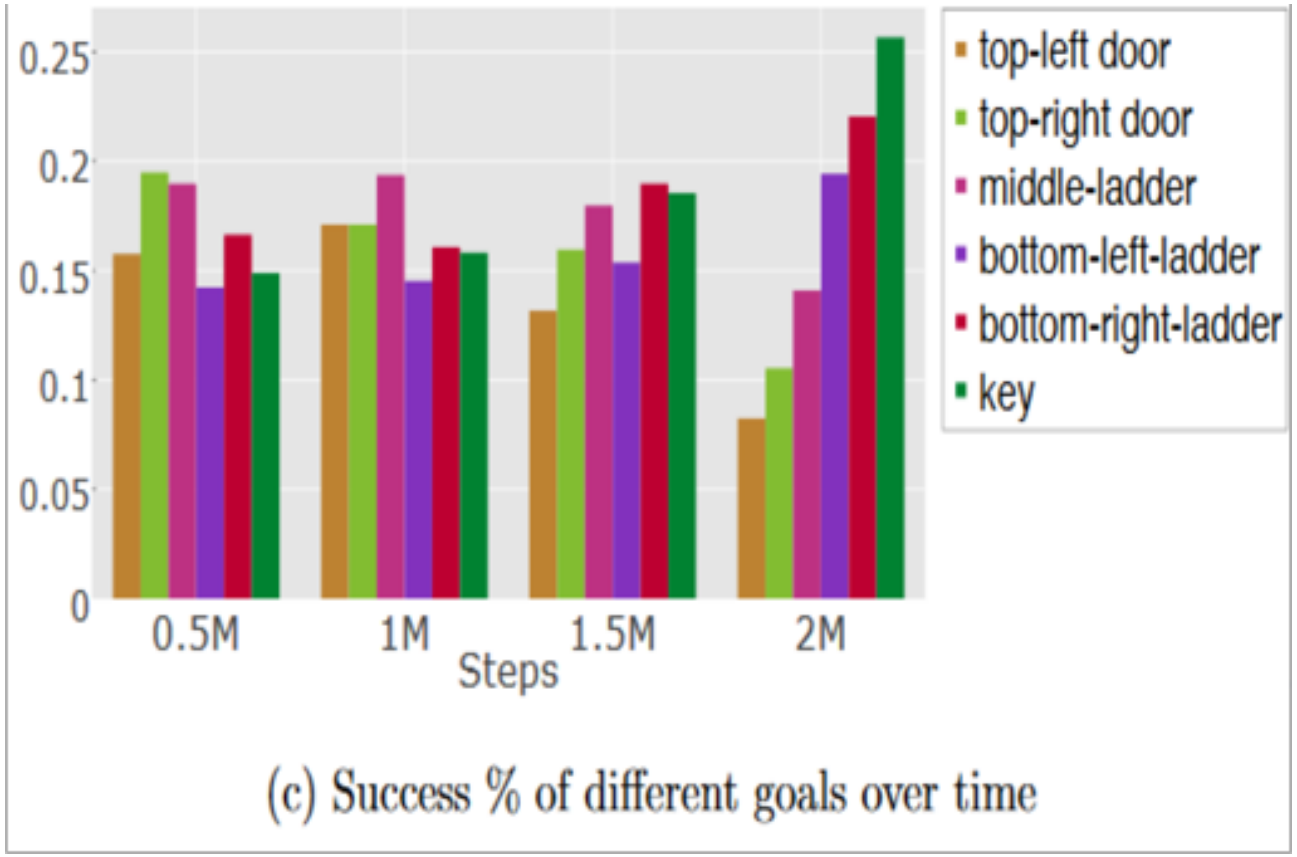
Controller

Meta-controller

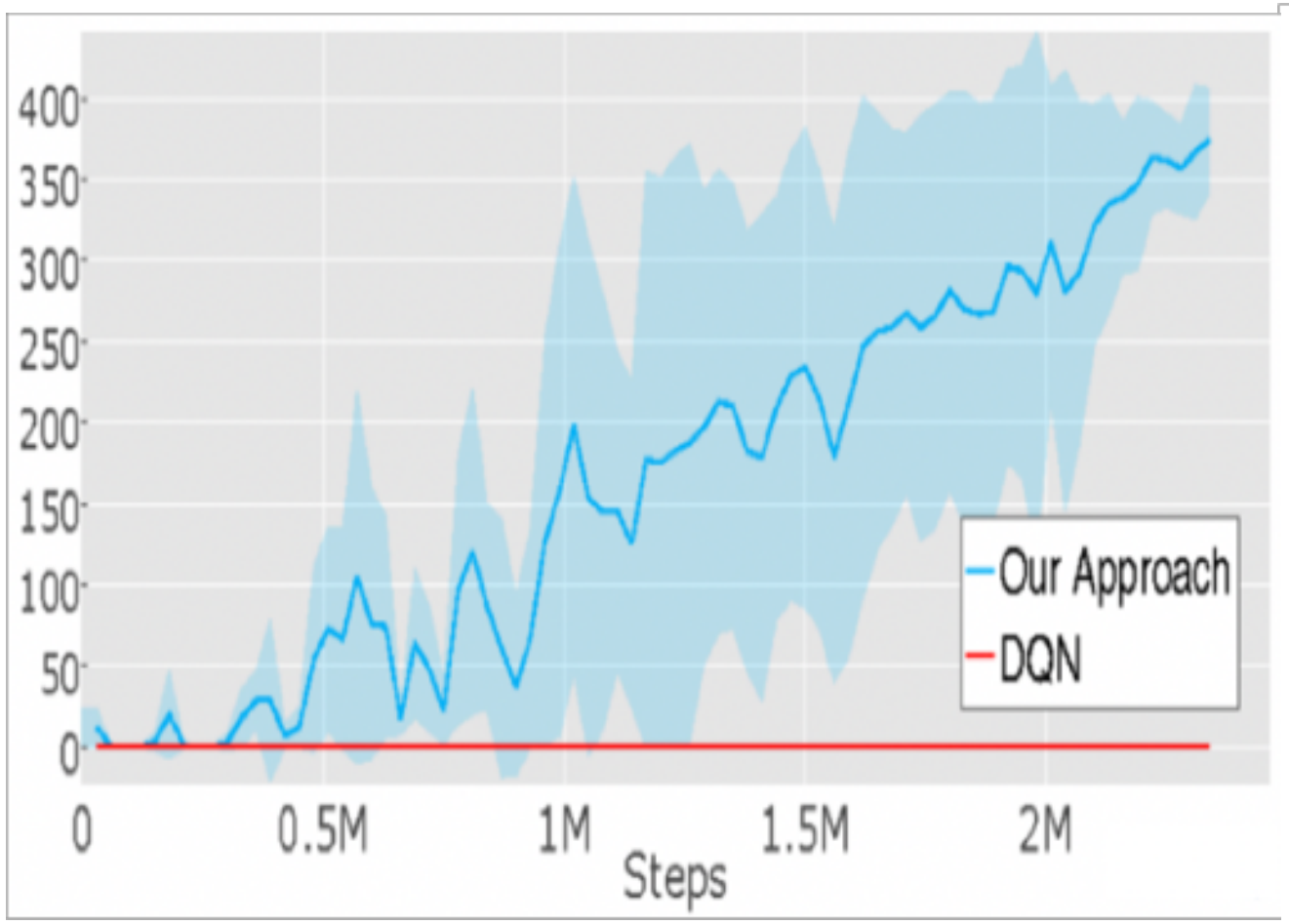
for $i = 1, num\_episodes$ **do**
  Initialize game and get start state description s
  $g \leftarrow \text{EPSGREEDY}(s, \mathcal{G}, \epsilon_2, Q_2)$
  while s is **not** terminal **do**
    $F \leftarrow 0$
    $s_0 \leftarrow s$
    while not (s is terminal **or** goal g reached) **do**
      $a \leftarrow \text{EPSGREEDY}(\{s, g\}, \mathcal{A}, \epsilon_{1,g}, Q_1)$
      Execute a and obtain next state s' and extrinsic reward f from environment
      Obtain intrinsic reward $r(s, a, s')$ from internal critic
      Store transition $(\{s, g\}, a, r, \{s', g\})$ in $\mathcal{D}_1$
      $\text{UPDATEPARAMS}(\mathcal{L}_1(\theta_{1,i}), \mathcal{D}_1)$
      $\text{UPDATEPARAMS}(\mathcal{L}_2(\theta_{2,i}), \mathcal{D}_2)$
      $F \leftarrow F + f$
      $s \leftarrow s'$
    end while
    Store transition (s_0, g, F, s') in $\mathcal{D}_2$
    if s is **not** terminal **then**
      $g \leftarrow \text{EPSGREEDY}(s, \mathcal{G}, \epsilon_2, Q_2)$
    end if
  end while
  Anneal ϵ_2 and adaptively anneal $\epsilon_{1,g}$ using average success rate of reaching goal g.
end for

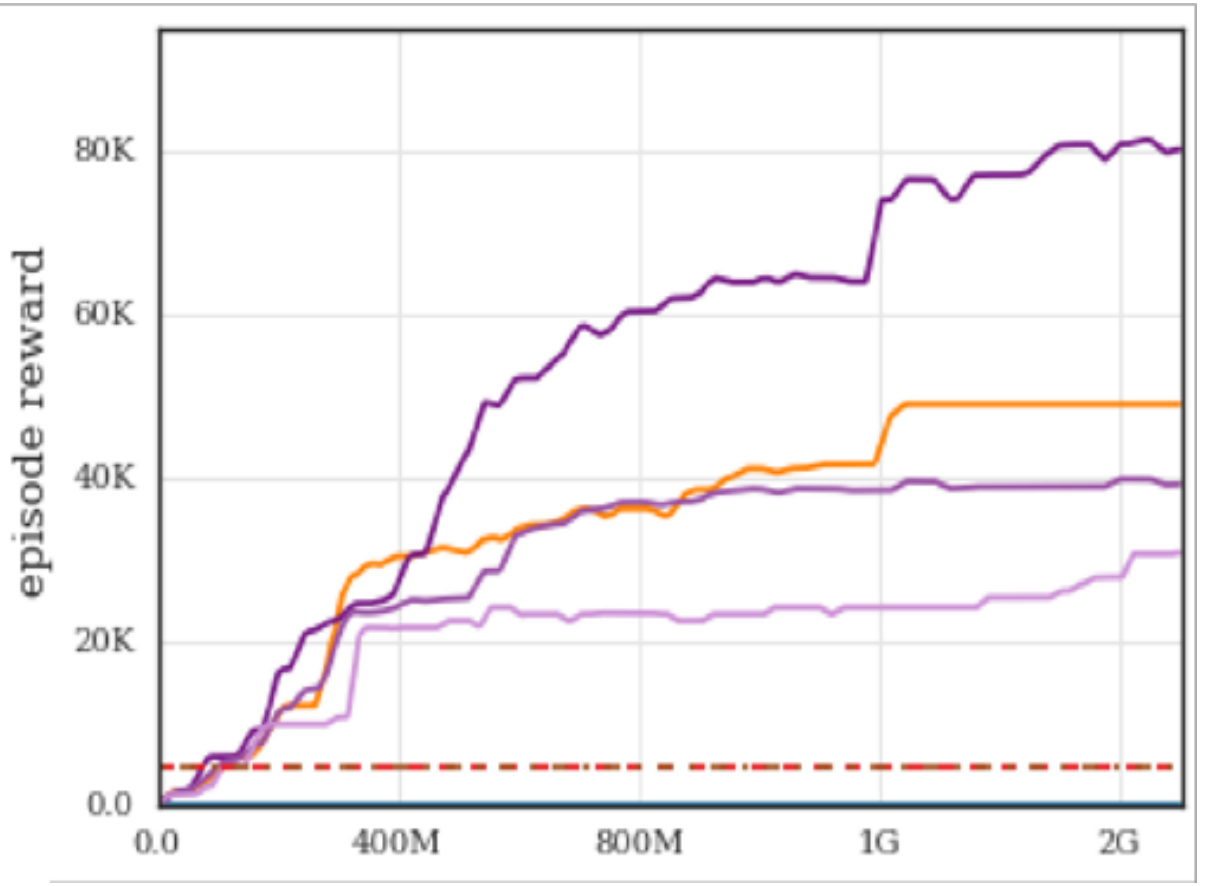
Results: Montezuma's Revenge





Results: Montezuma's Revenge





Learning from Artificial Demonstrations

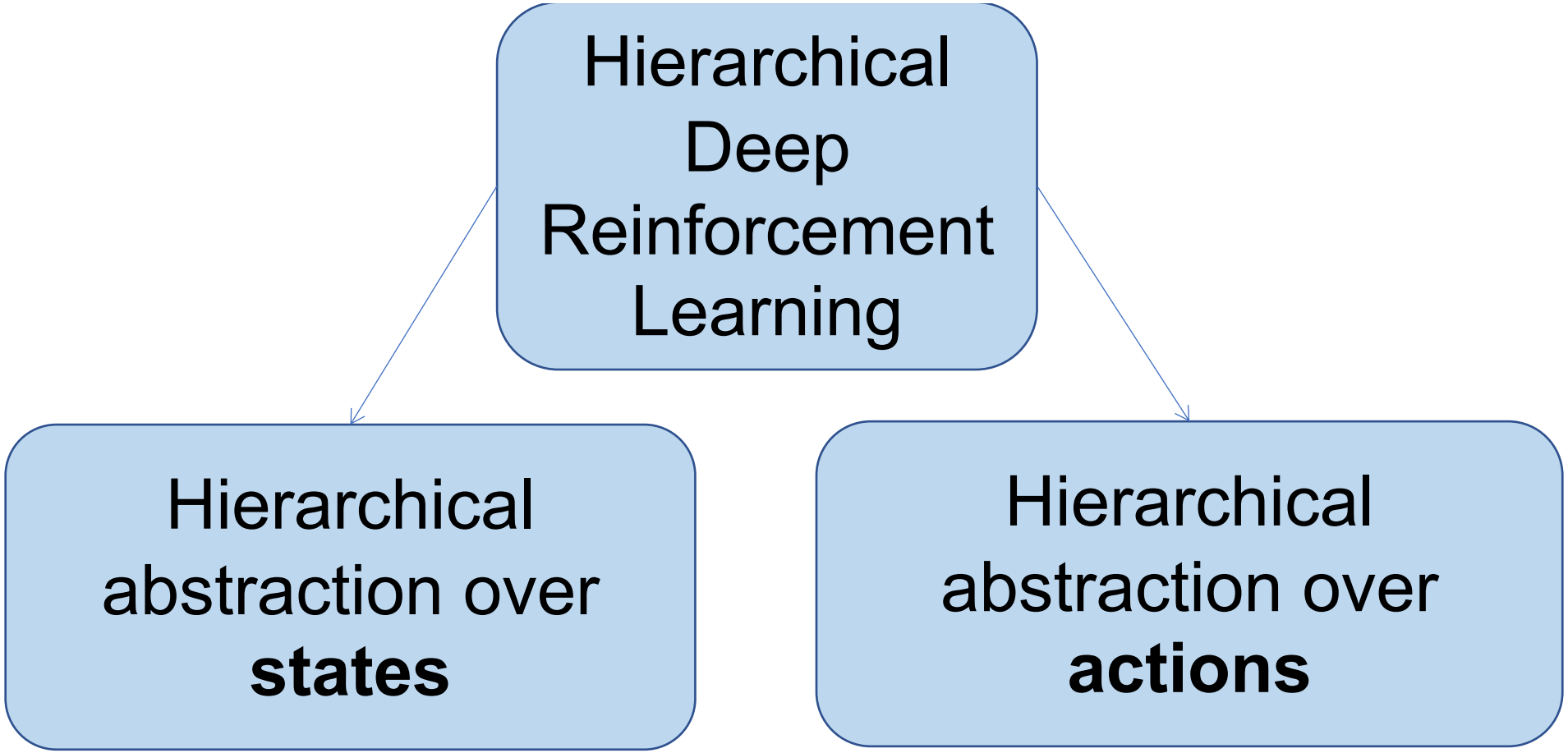
https://arxiv.org/pdf/1805.11592.pdf

Hierarchical abstraction over states

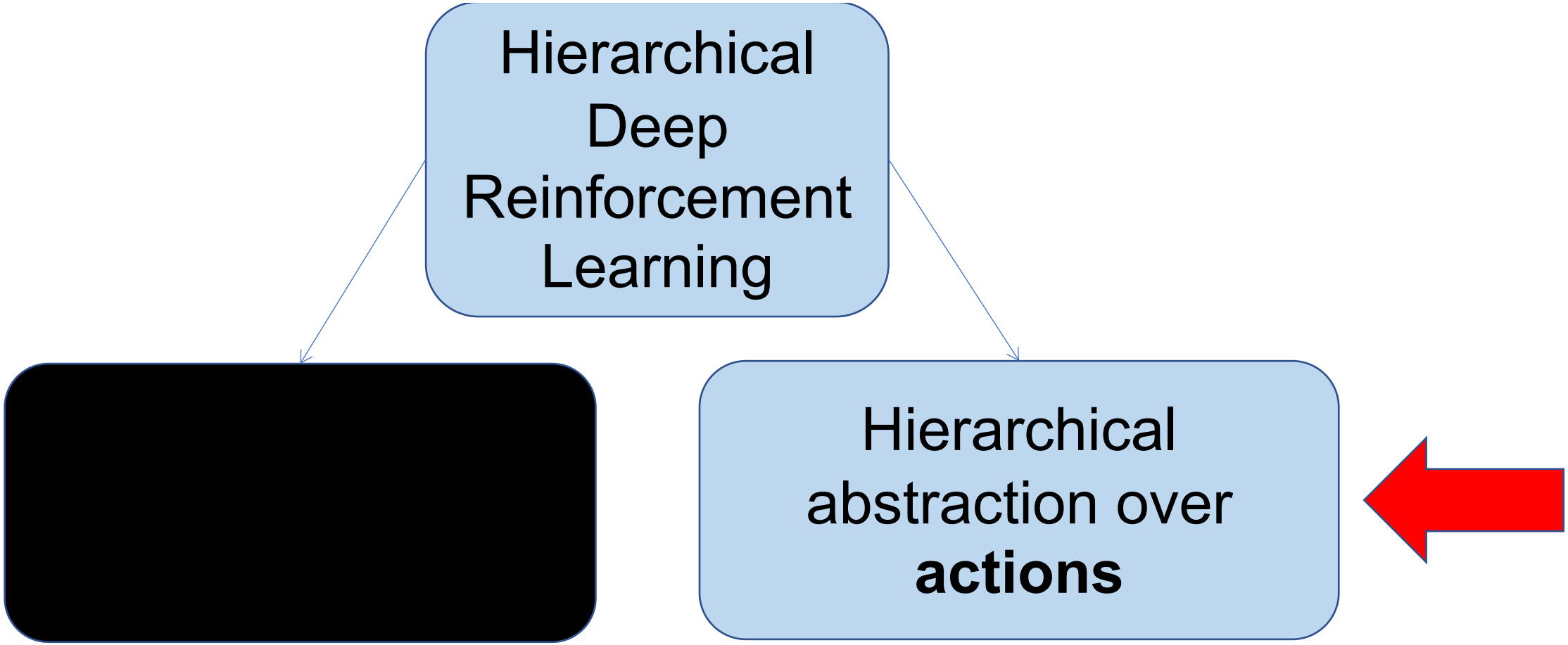
Rewards over **sub-goals**:

+ Reduce sparsity
+ Rewards are less long-term
+ Training is more efficient (**?**)
+ Local minima (**?**)

Two papers – Two hierarchical abstractions



Two papers – Two hierarchical abstractions



Hierarchical abstraction over actions

- Not **efficient** long-term planning!

Solution:

- **Long-term planning** w.r.t actions (options)

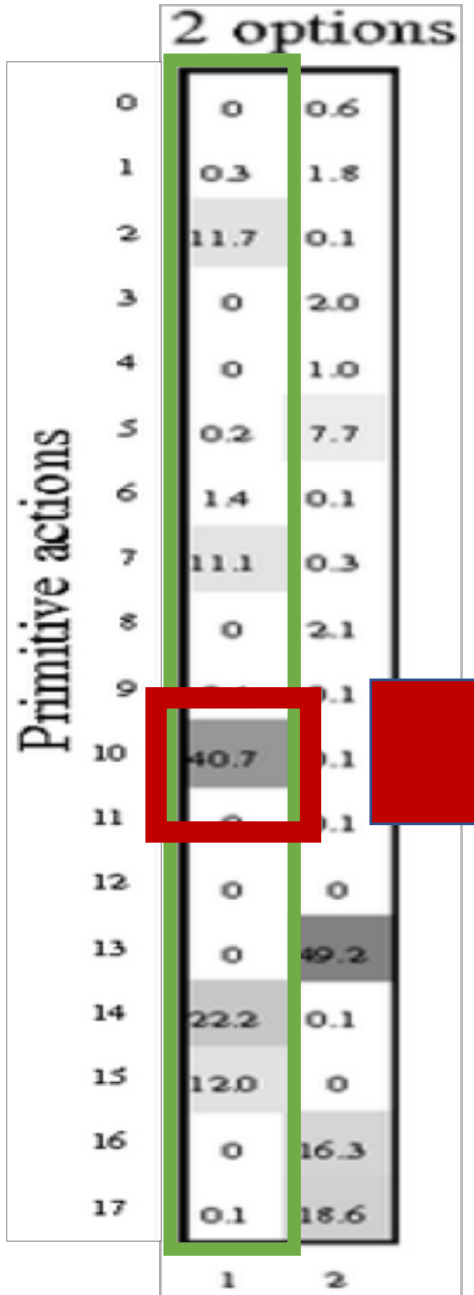
Hierarchical abstraction over actions

Key points:

- The action space may be **sparse** given the current game state.
- **Strategies** to solve the current task. **Learn** them !

Hierarchical abstraction over actions

s'



a_{10}

The Option-Critic architecture

- Markovian option ω in Ω is a triple $(I_\omega, \pi_\omega, \beta_\omega)$

- I_ω is an initiation set ($I_\omega \subseteq S$)
- Π_ω is an intra-option policy
- β_ω is a termination function ($\beta_\omega : S \rightarrow [0,1]$)

The Option-Critic Q-learning formula

- Option value function

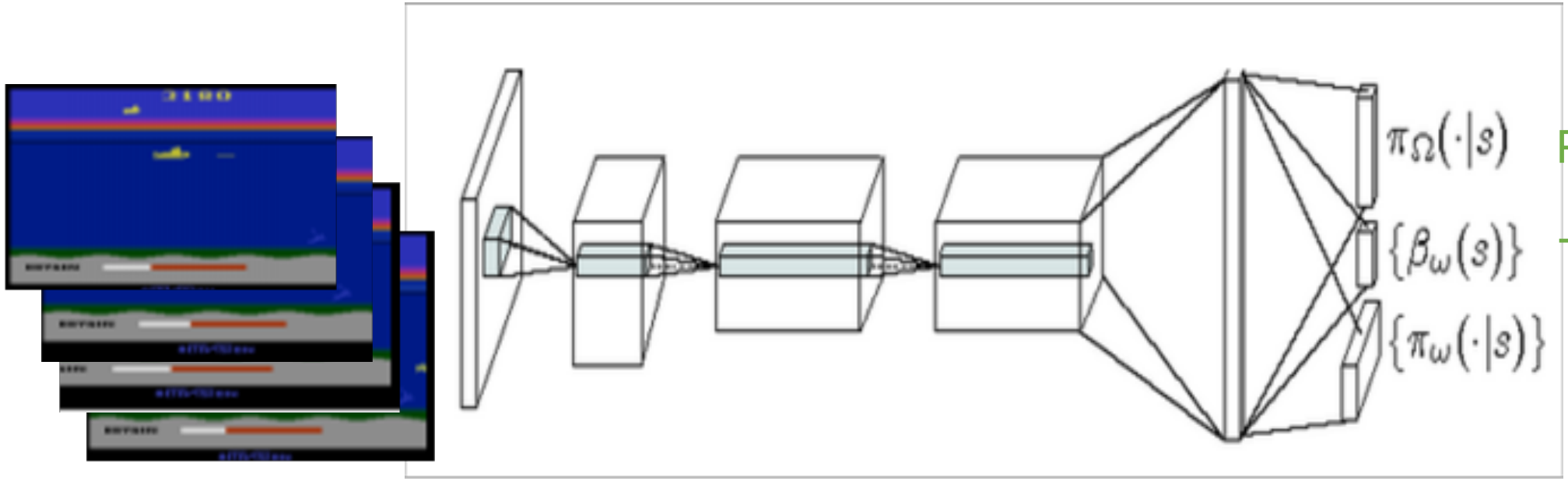
$$Q_\Omega(s,\omega) = \sum_a \pi_{\omega,\theta}(a \mid s) \Big(r(s,a) + \gamma \sum_{s'} \mathrm{P}(s' \mid s,a) \Big((1-\beta_{\omega,\vartheta}(s')) Q_\Omega(s',\omega) + \beta_{\omega,\vartheta}(s') \max_{\bar{\omega}} Q_\Omega(s',\bar{\omega}) \Big) \Big)$$

Go on with ω

Terminate ω $V_\Omega(s')$

Option critic architecture

- Arcade Learning Environment



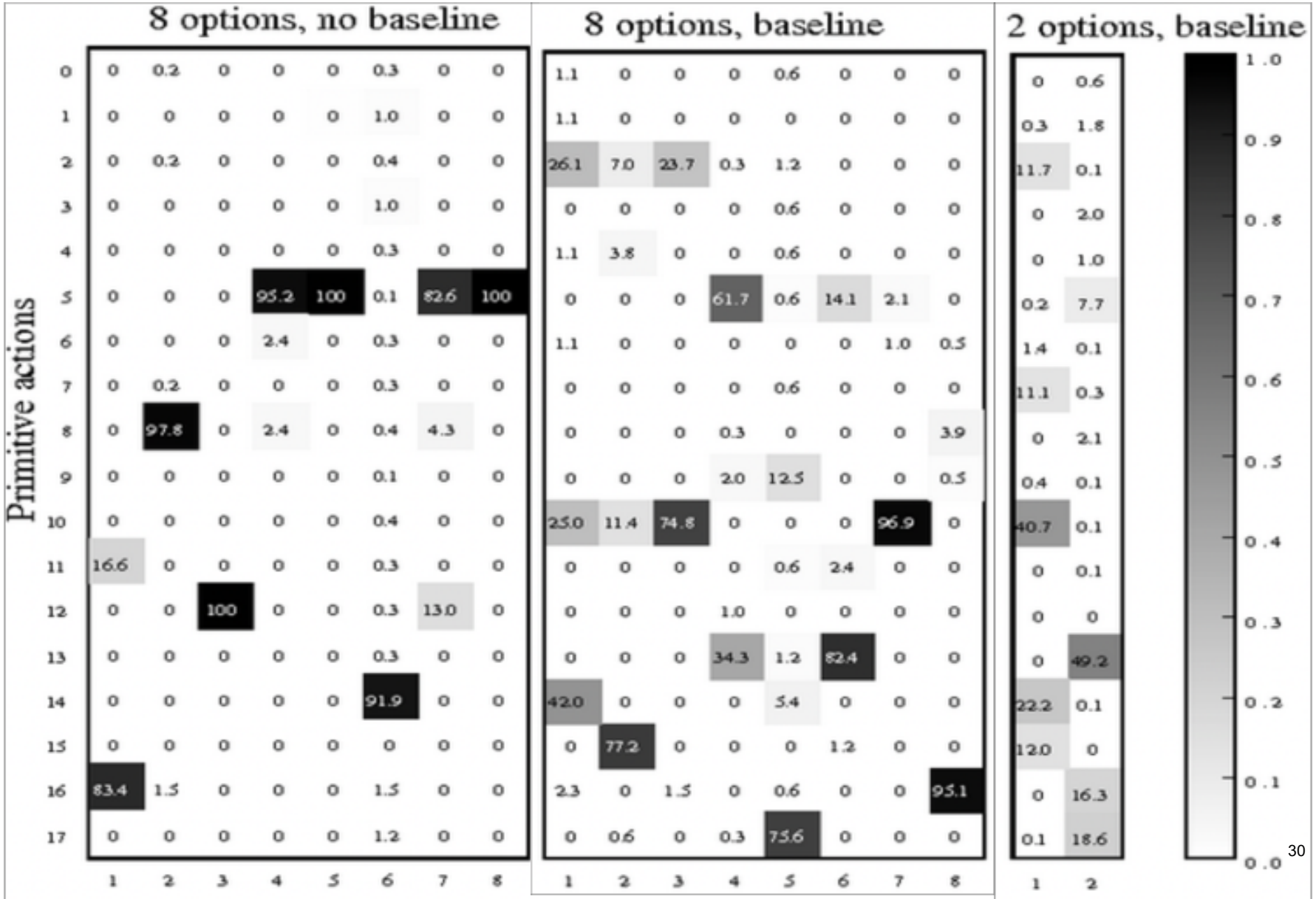
Policy over options

Termination function

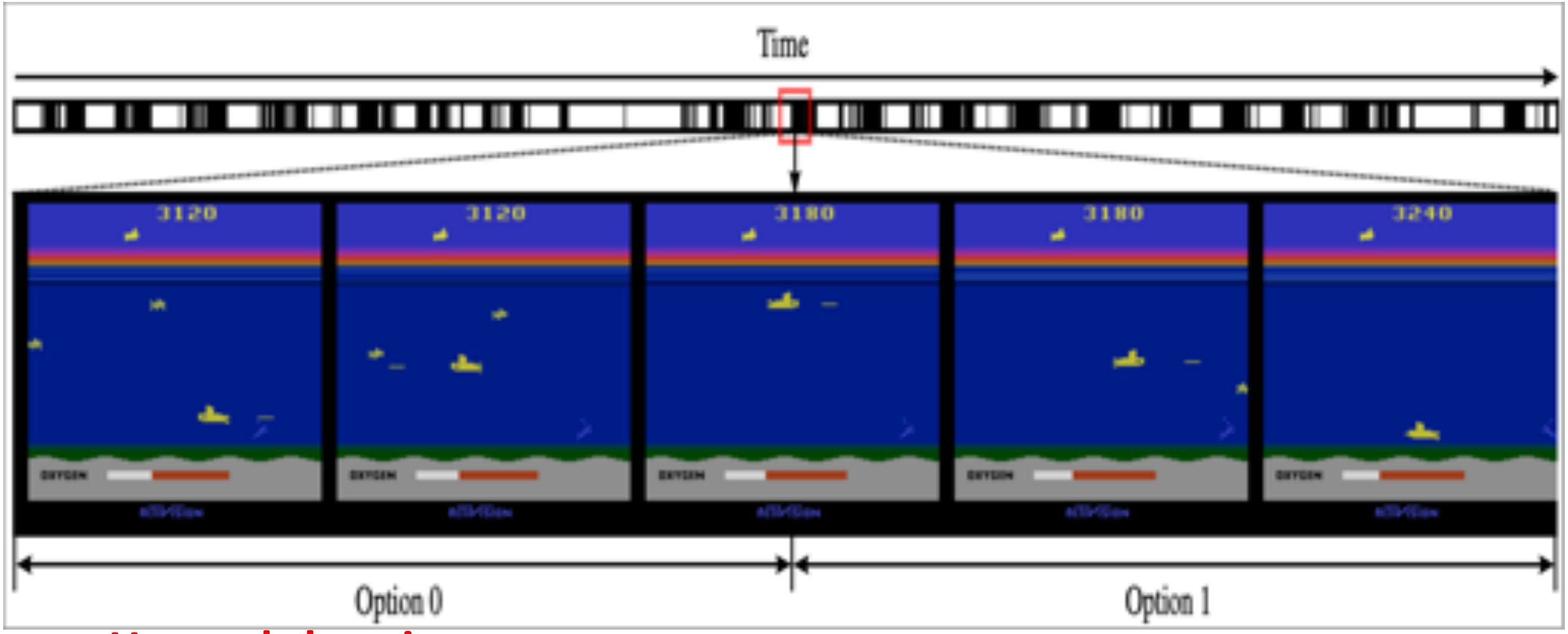
Intra-option policy

Seaquest options

- Options: **distributions** over **primitive actions**



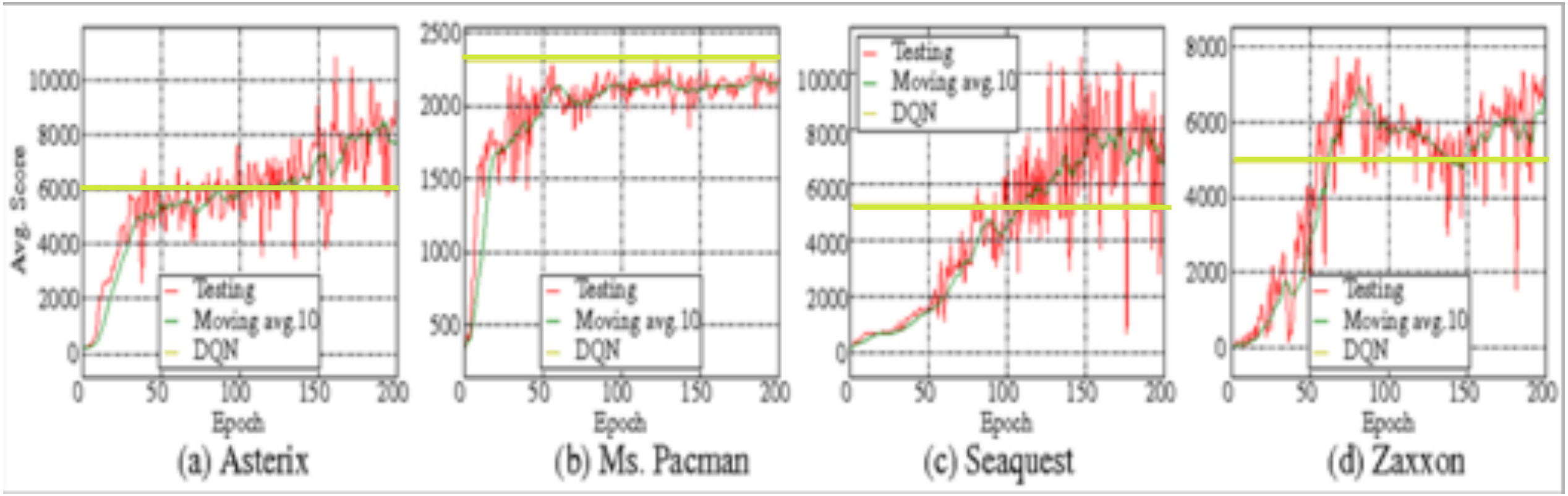
Hierarchical abstraction over actions



Upward shooting sequences

Downward shooting sequences

Results - ATARI



Rainbow

428200

https://arxiv.org/pdf/1710.02298.pdf

Microsoft AI

1 M (game completion)

https://www.youtube.com/watch?v=TpB1B9Tr_ck

Dueling DDQN

50254

https://arxiv.org/pdf/1710.02298.pdf

A3C

24622

https://arxiv.org/pdf/1602.01783.pdf

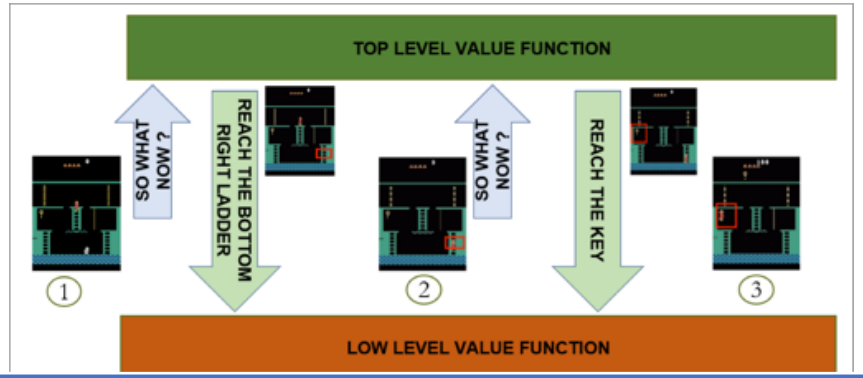
Hierarchical abstraction over actions

Abstraction over **action space**:

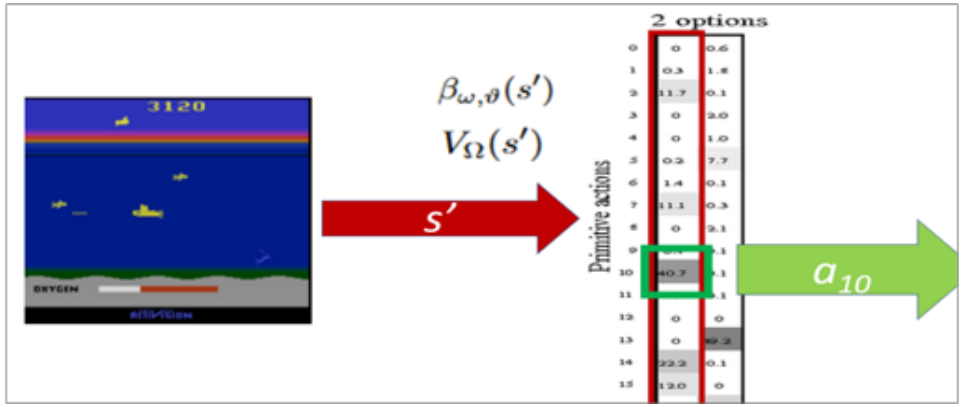
+ Long-term planning
+ How many options (?)
+ Training is more efficient (?)
+ Local minima (?)

Summary and conclusions

- Abstraction over states



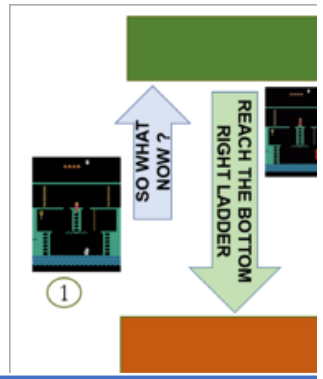
Abstraction over actions



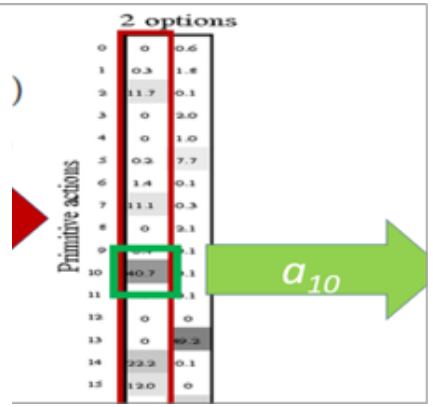
1° paper	2° paper
Adopted temporal abstraction (Sutton et al.)	Adopted temporal abstraction (Sutton et al.)
Sub-goals (meta-controller)	Options
Transfer learning (among sub-goals)	Transfer learning (among options)
Specify sub-goals. Establish extra rewards	Specify the **number** of options. Also, no extra rewards.
Scalable to multiple sub-goals (same DQN controller)	Scalable to multiple options (same DQN)

Summary and conclusions

- Abstraction over states



Abstraction over actions



NO MEMORY W.R.T THE HIERARCHICAL ABSTRACTION (sub-goals/options)

→ Next talk

1° paper	
Adopted temporal ab	ion (Sutton et al.)
Sub-goals (meta-controller)	Options
Transfer learning (among sub-goals)	Transfer learning (among options)
Specify sub-goals. Establish extra rewards	Specify the **number** of options. Also, no extra rewards.
Scalable to multiple sub-goals (same DQN controller)	Scalable to multiple options (same DQN)

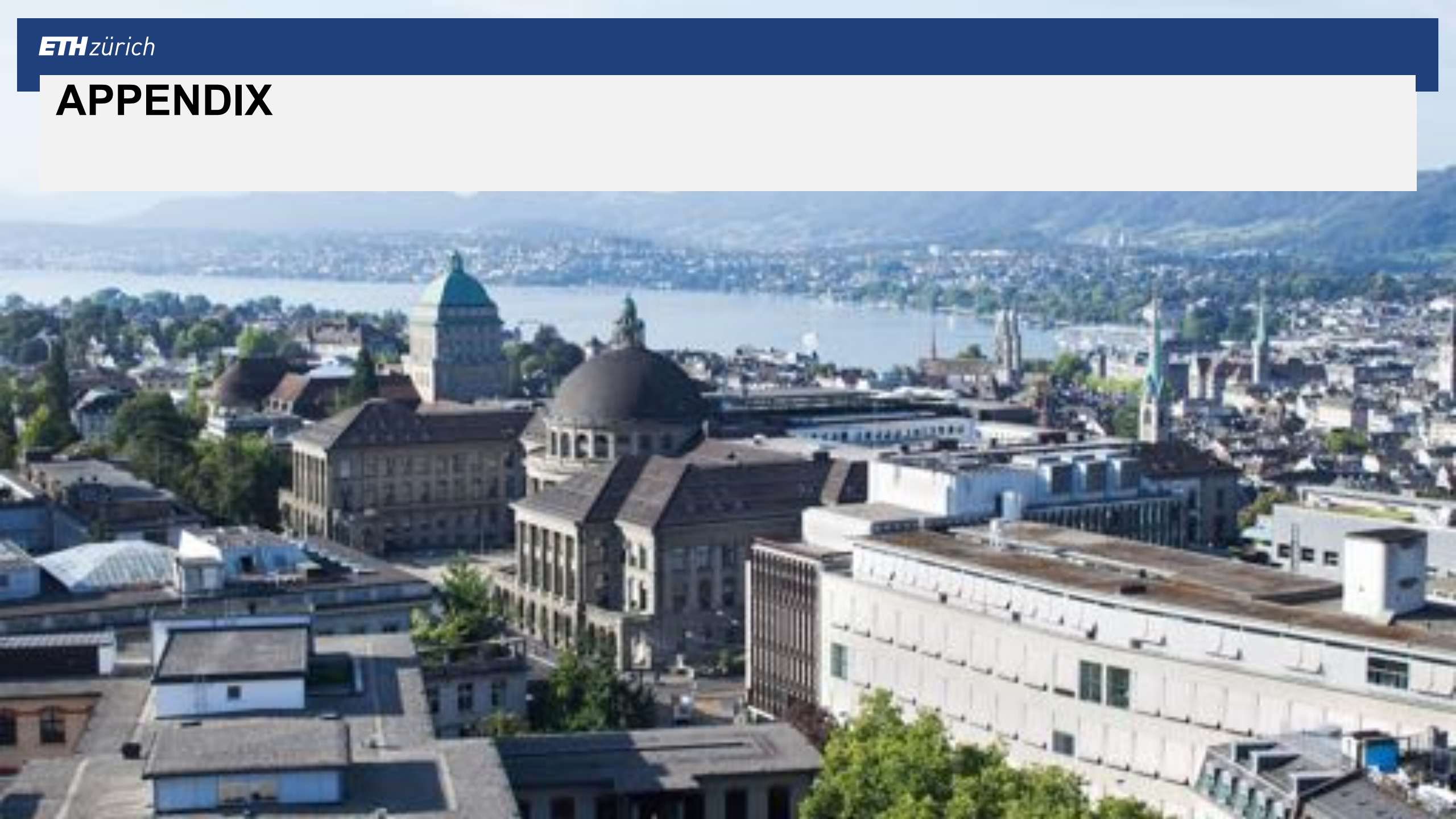
End

- Thanks !

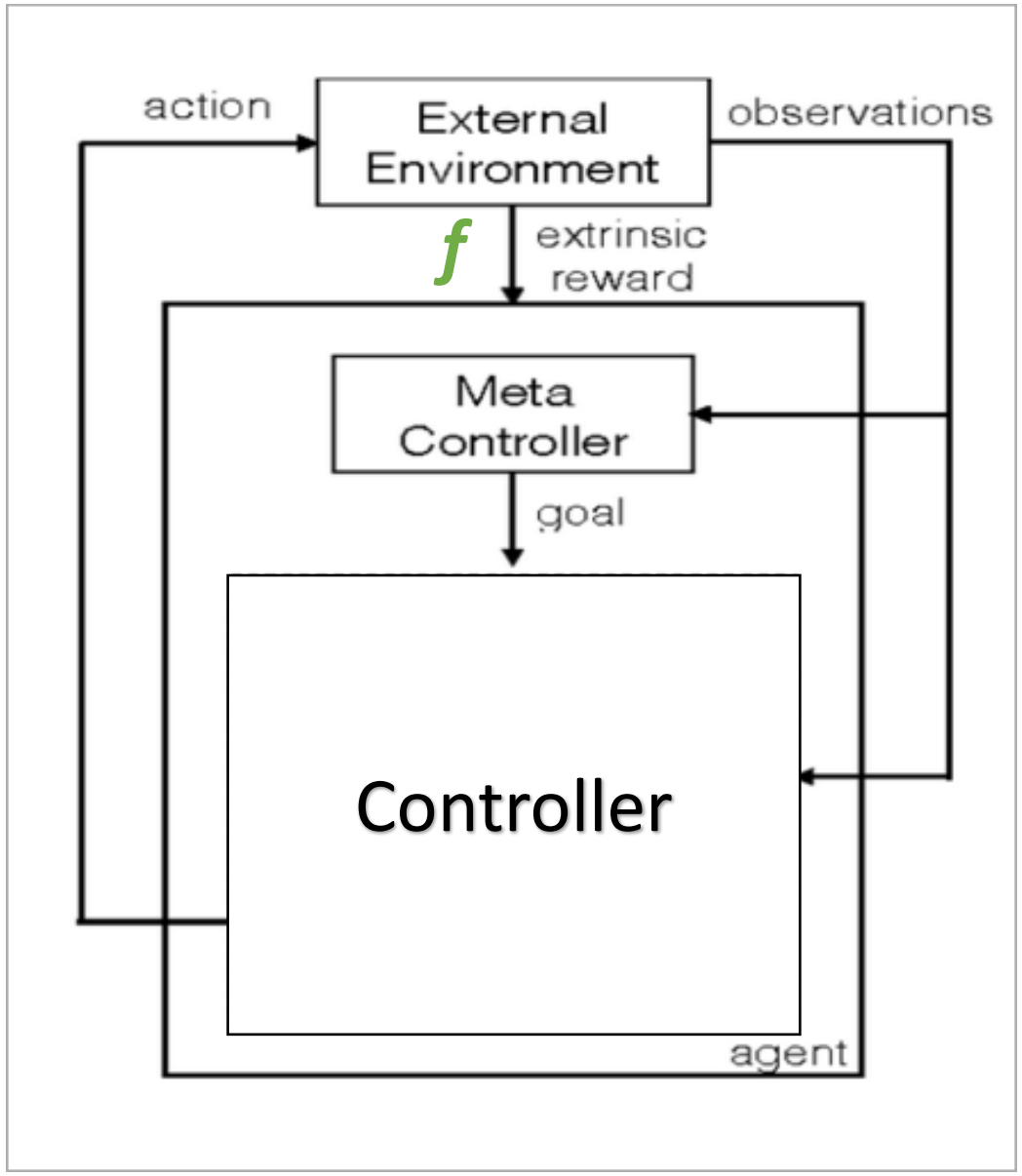
QUESTIONS ?

ETH zürich

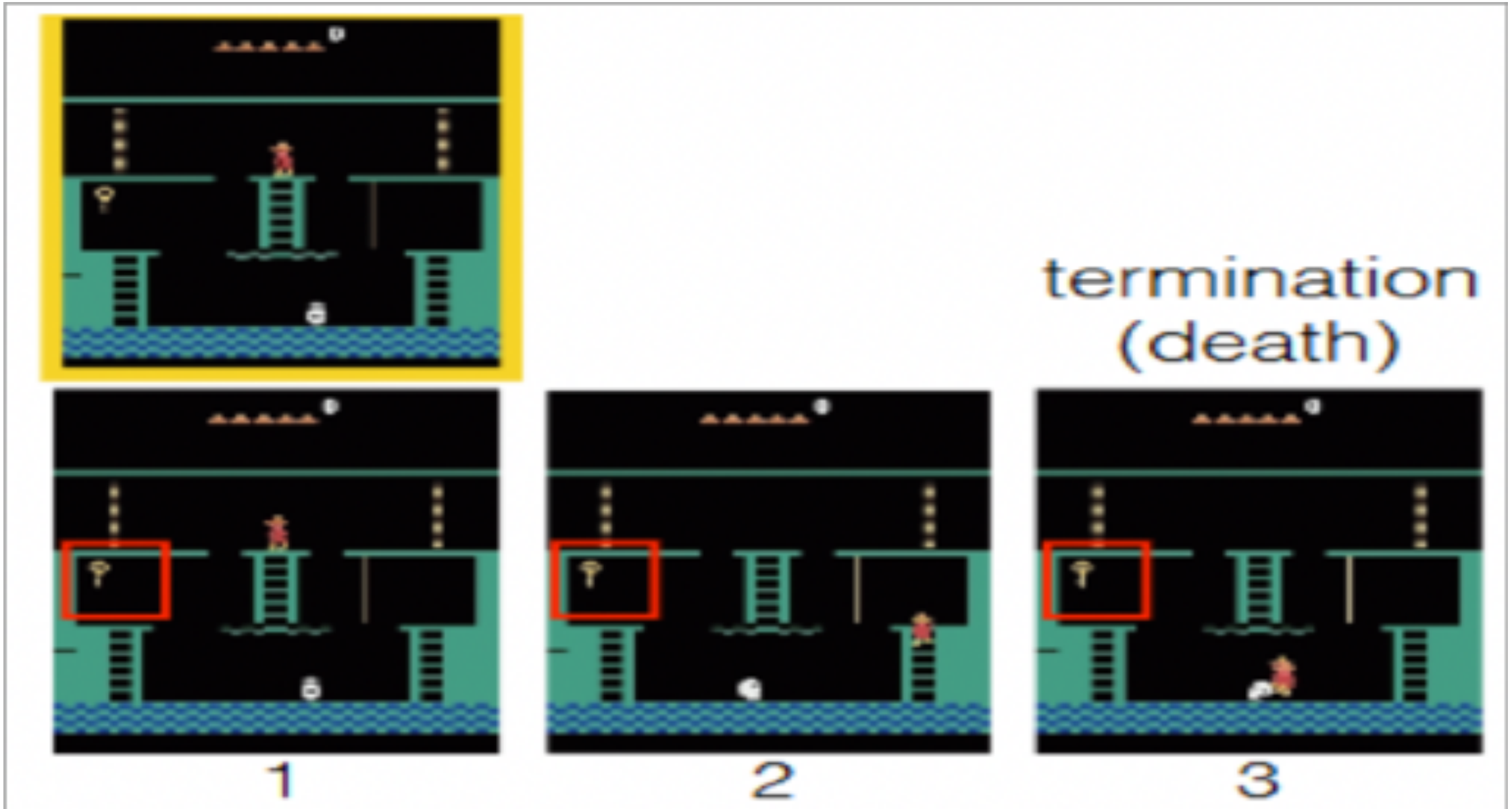
APPENDIX



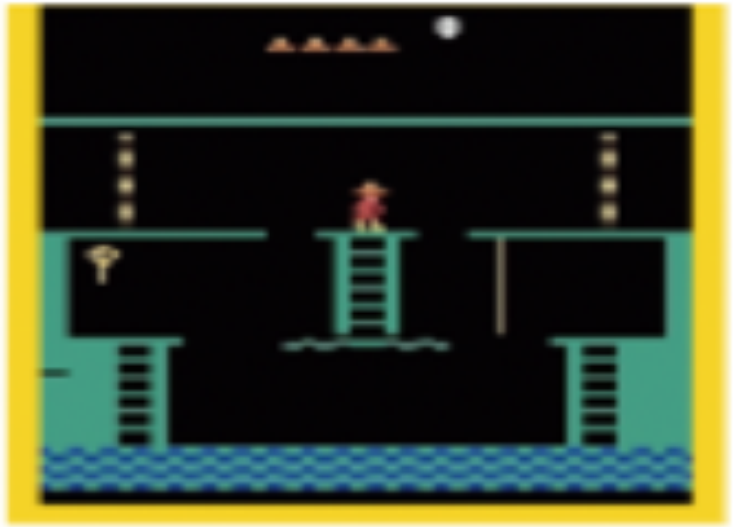
Hierarchical abstraction over states

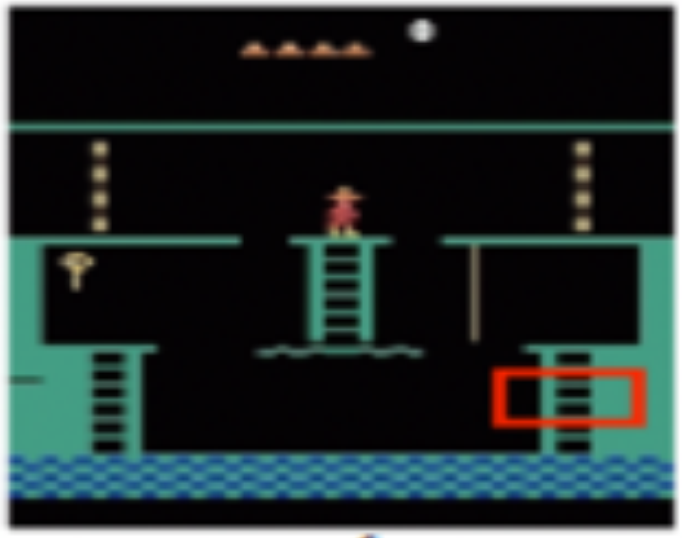


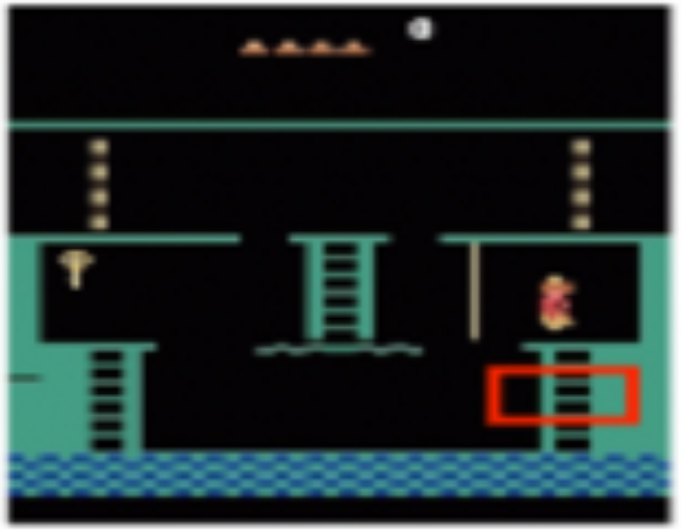
Hierarchical abstraction over states – Test screenshot



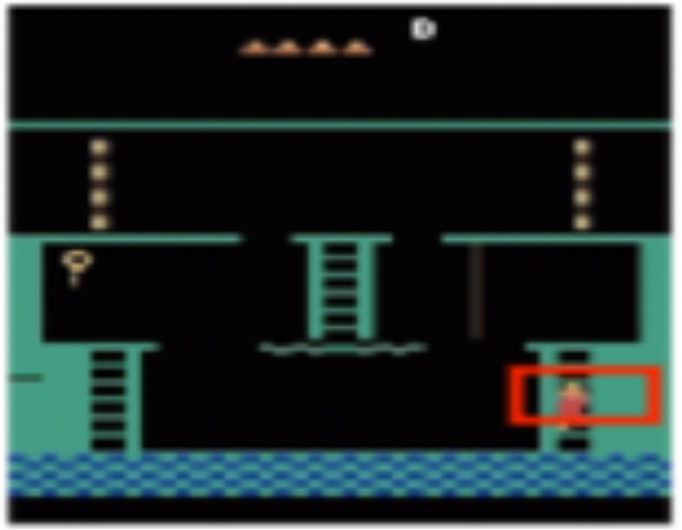
Hierarchical abstraction over states – Test screenshots



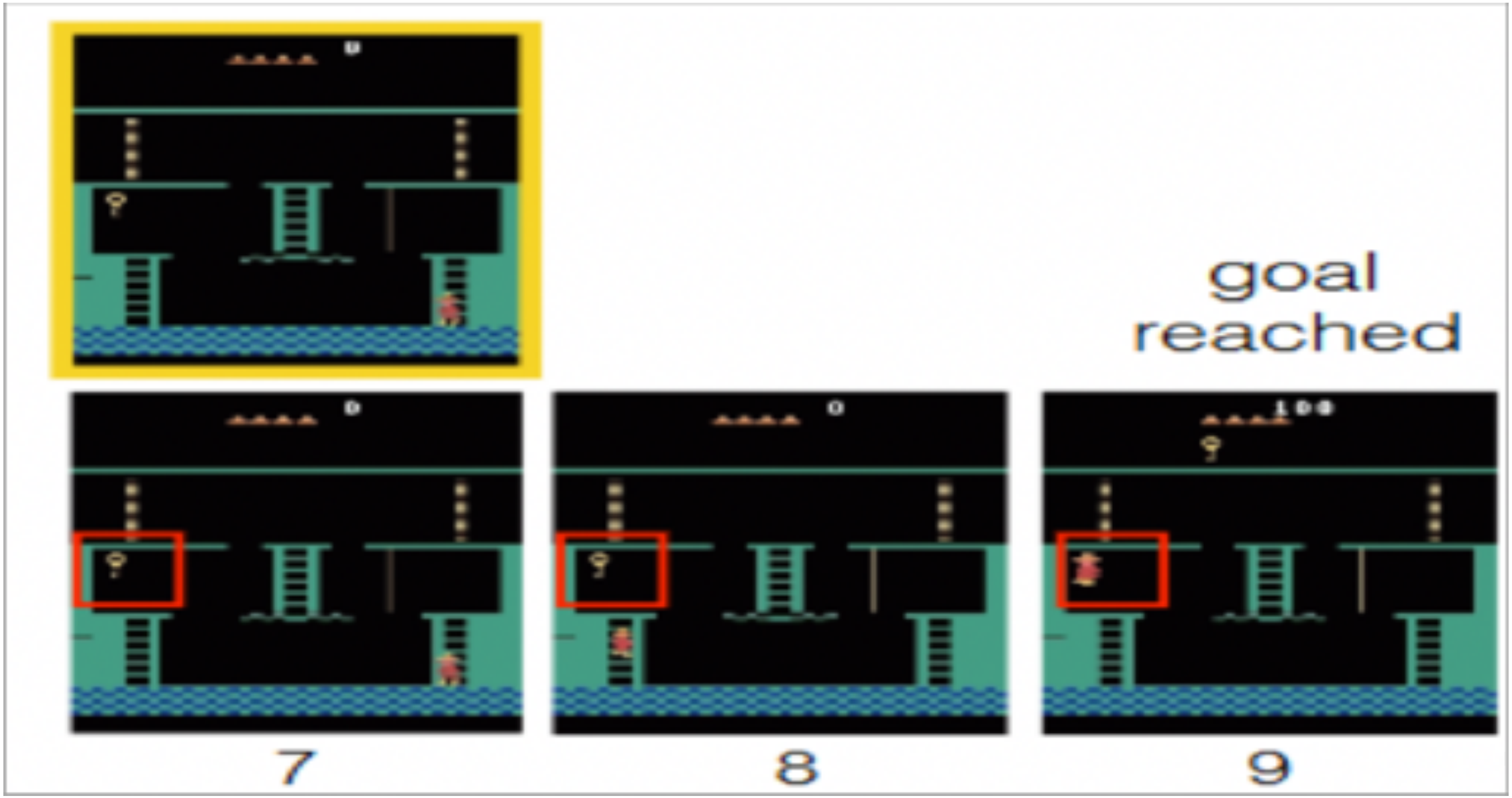




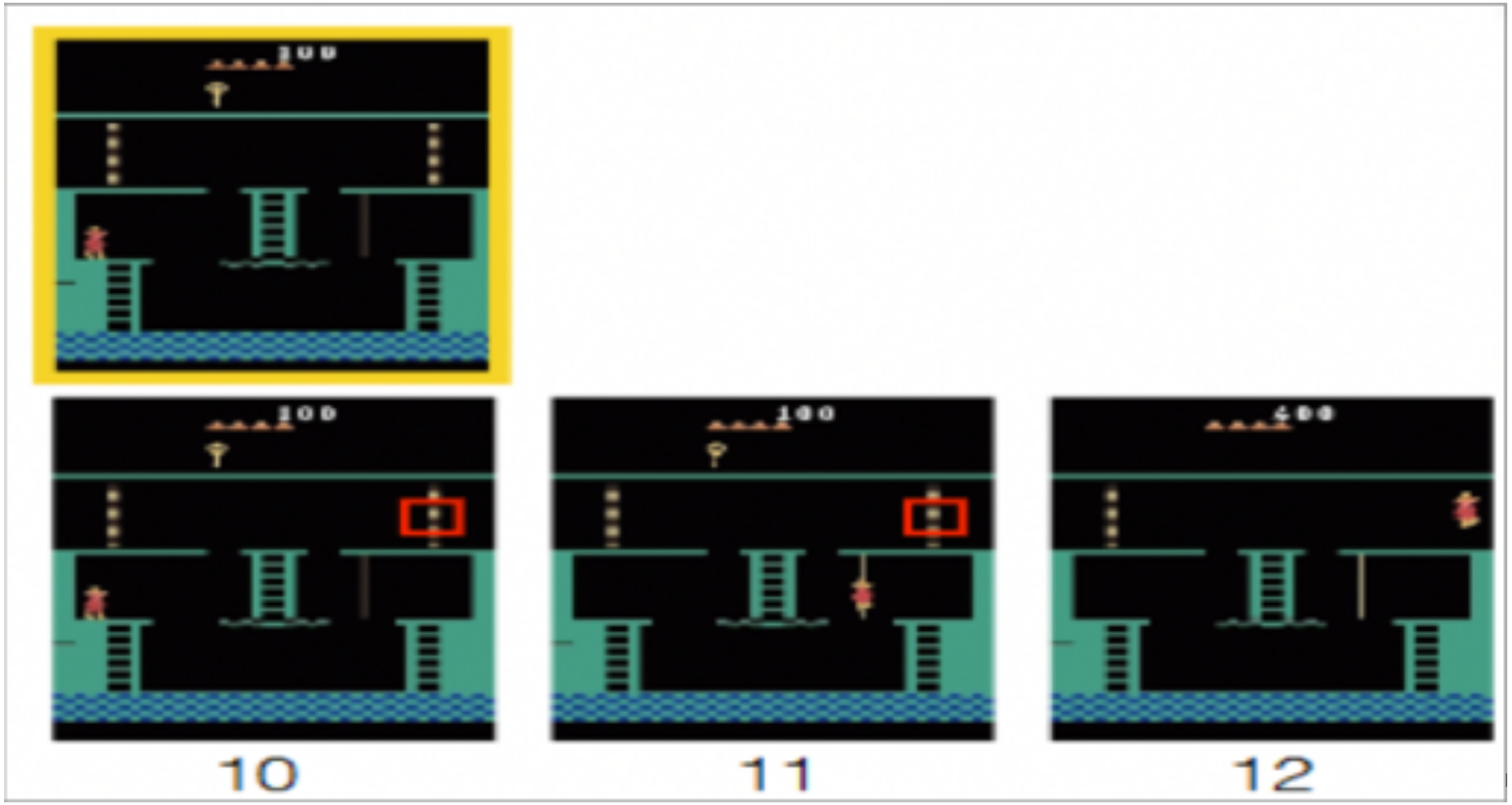
goal reached



Hierarchical abstraction over states – Test screenshots



Hierarchical abstraction over states – Test screenshots



Hierarchical abstraction over states – Loss functions

- There are two loss functions associated to the controller and meta-controller respectively.

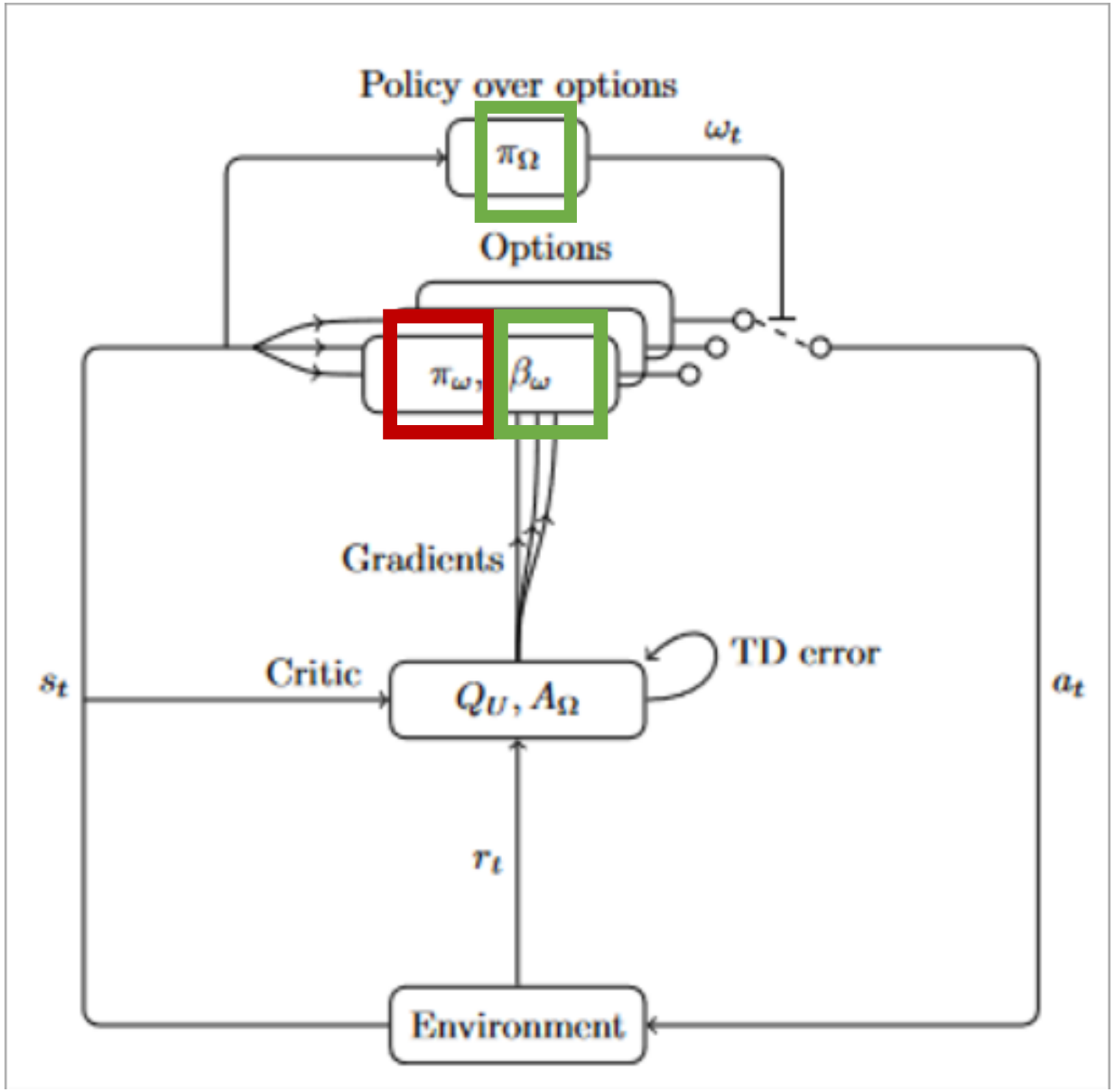
$$L_1(\theta_{1,i}) = \mathrm{E}_{(s,a,g,r,s')\sim D_1}[(y_{1,i} - Q_1(s,a;\theta_{1,i},g))^2],$$

- Its gradient is

$$\nabla_{\theta_{1,i}} L_1(\theta_{1,i}) = \mathrm{E}_{(s,a,r,s'\sim D_1)}\left[\left(r + \gamma \max_{a'} Q_1(s',a';\theta_{1,i-1},g) - Q_1(s,a;\theta_{1,i},g)\right)\nabla_{\theta_{1,i}} Q_1(s,a;\theta_{1,i},g)\right)\right]$$

- Similarly the gradient for L_2 *can be derived*

Hierarchical abstraction over actions



Hierarchical abstraction over actions

- Option-critic with tabular intra-option Q-learning

$s \leftarrow s_0$
Choose ω according to an ϵ-soft policy over options $\pi_\Omega(s)$
repeat
  Choose a according to $\pi_{\omega,\theta}(a \mid s)$
  Take action a in s, observe s', r
  1. Options evaluation:
  $\delta \leftarrow r - Q_U(s,\omega,a)$
  if s' *is non-terminal* **then**
    $\delta \leftarrow \delta + \gamma(1-\beta_{\omega,\vartheta}(s'))Q_\Omega(s',\omega) + \gamma\beta_{\omega,\vartheta}(s')\max_{\bar{\omega}} Q_\Omega(s',\bar{\omega})$
  end
  $Q_U(s,\omega,a) \leftarrow Q_U(s,\omega,a) + \alpha\delta$
  2. Options improvement:
  $\theta \leftarrow \theta + \alpha_\theta \frac{\partial \log \pi_{\omega,\theta}(a \mid s)}{\partial\theta} Q_U(s,\omega,a)$
  $\vartheta \leftarrow \vartheta - \alpha_\vartheta \frac{\partial \beta_{\omega,\vartheta}(s')}{\partial\vartheta}(Q_\Omega(s',\omega) - V_\Omega(s'))$
  if $\beta_{\omega,\vartheta}$ *terminates in* s' **then**
  choose new ω according to ϵ-soft$(\pi_\Omega(s'))$
  $s \leftarrow s'$
until s' *is terminal*

Hierarchical abstraction over actions

- Advantage function + Regularizer

“As a consequence of optimizing for the return, the termination gradient tends to shrink options over time. This is expected since in theory primitive actions are sufficient for solving any MDP.”

$$A_\Omega(s,\omega)+\xi = Q_\Omega(s,\omega) - V_\Omega(s)+\xi$$

“This makes the advantage function positive
if the value of an option is near the optimal one,
thereby stretching it."