

Applying active perception and reinforcement learning to partially observable worlds



Paul A. Crook

paulc@dai.ed.ac.uk

Supervisors: Dr. Gillian Hayes & Prof. Bob Fisher

Institute of Perception Action and Behaviour

School of Informatics

University of Edinburgh

Introduction

- Majority of work applying reinforcement learning (RL) to partially observable tasks assume that it is necessary to use **state estimation** techniques in order to make the task Markovian.



Introduction



- Majority of work applying reinforcement learning (RL) to partially observable tasks assume that it is necessary to use **state estimation** techniques in order to make the task Markovian.
- Unfortunately simultaneous learning of state estimation mappings and RL policies based on those mappings has proved very brittle [Shatkay and Kaelbling 1997, Hartley 2004, Crook 2006], *e.g.* PDA, UDM, U-Tree.

Introduction (cont.)



In using RL to learn policies for embedded and embodied agents who tackle “real world” tasks I’m

- less interested in the ideal of **optimal** policies, especially when reliable convergence cannot be guaranteed and

Introduction (cont.)



In using RL to learn policies for embedded and embodied agents who tackle “real world” tasks I’m

- less interested in the ideal of **optimal** policies, especially when reliable convergence cannot be guaranteed and
- more interested in *reliable* convergence to **satisficing** policies.

satisficing – reaching some minimum level – Simon [1956]

“Evidently, organisms adapt well enough to ‘satisfice’; they do not, in general, ‘optimize’.”

Introduction (cont.)

Thus my PhD work considers an alternative assumption; that it is possible to extend an agent's abilities in such a way that satisficing deterministic policies exist for partially observable tasks.



Introduction (cont.)

- First, I introduce the use of active perception as a method of extending agents' abilities.



Introduction (cont.)

- First, I introduce the use of active perception as a method of extending agents' abilities.
- Second, given the assumption on the previous slide, I consider what design of RL algorithm is required in order to reliably learn satisficing deterministic policies in partially observable worlds.



Active Perception

- I define Active Perception as **any perceptual process that an agent can exert control over.**



Active Perception

- I define Active Perception as **any perceptual process that an agent can exert control over**.
- This includes *covert* (e.g. selection of the filtering done on sensor data) as well as *overt* control (e.g. panning and tilting of a camera).



Active Perception

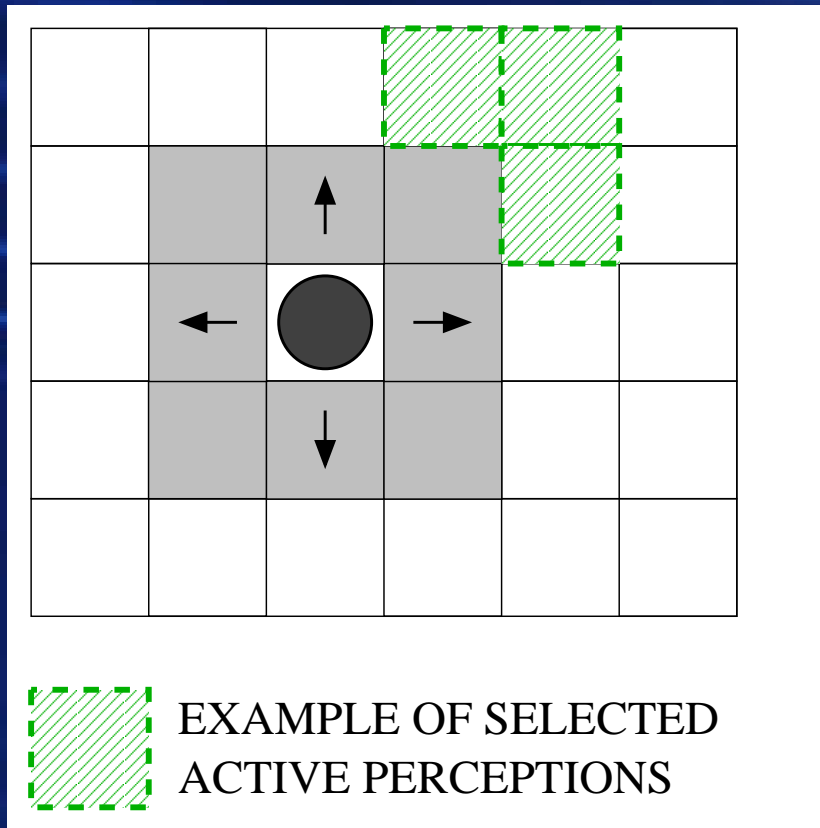


- I define Active Perception as **any perceptual process that an agent can exert control over.**
- This includes *covert* (e.g. selection of the filtering done on sensor data) as well as *overt* control (e.g. panning and tilting of a camera).
- There is a grey area between *perceptual actions* and purely *physical actions* – an issue we'll return to later.

Example Active Perception Agents

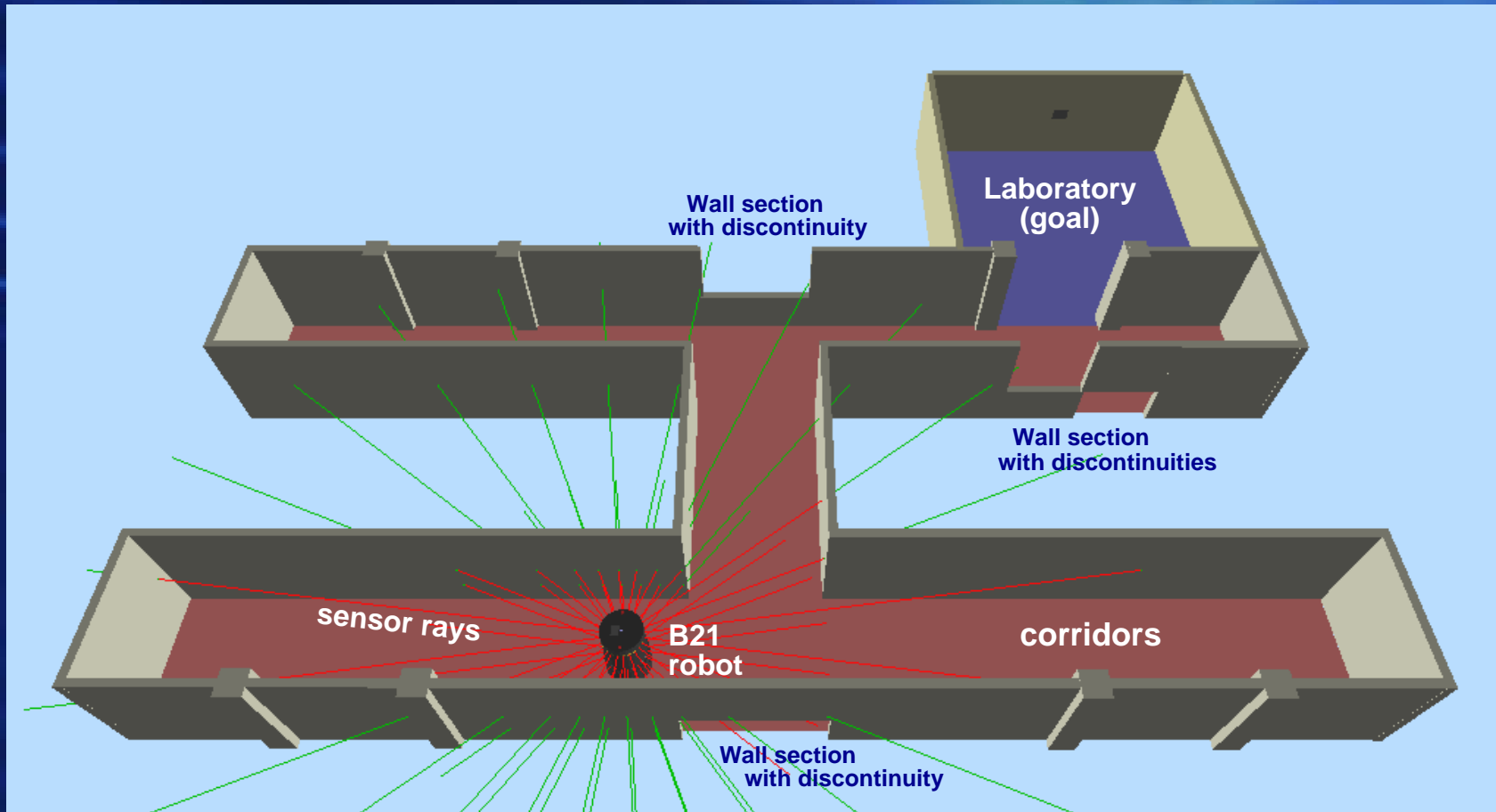


Grid World Agent



Example Active Perception Agents

Corridor Navigation



Why Interest in Active Perception?



To deal with same issues for which state-estimation is used, *i.e.*

- RL learns deterministic reactive policies.

Why Interest in Active Perception?



To deal with same issues for which state-estimation is used, *i.e.*

- RL learns deterministic reactive policies.
- For a partially observable task, no guarantee that a deterministic reactive policy exists.

Why Interest in Active Perception?



To deal with same issues for which state-estimation is used, *i.e.*

- RL learns deterministic reactive policies.
- For a partially observable task, no guarantee that a deterministic reactive policy exists.
- Those policies which do exist might be improved by adding active perception.

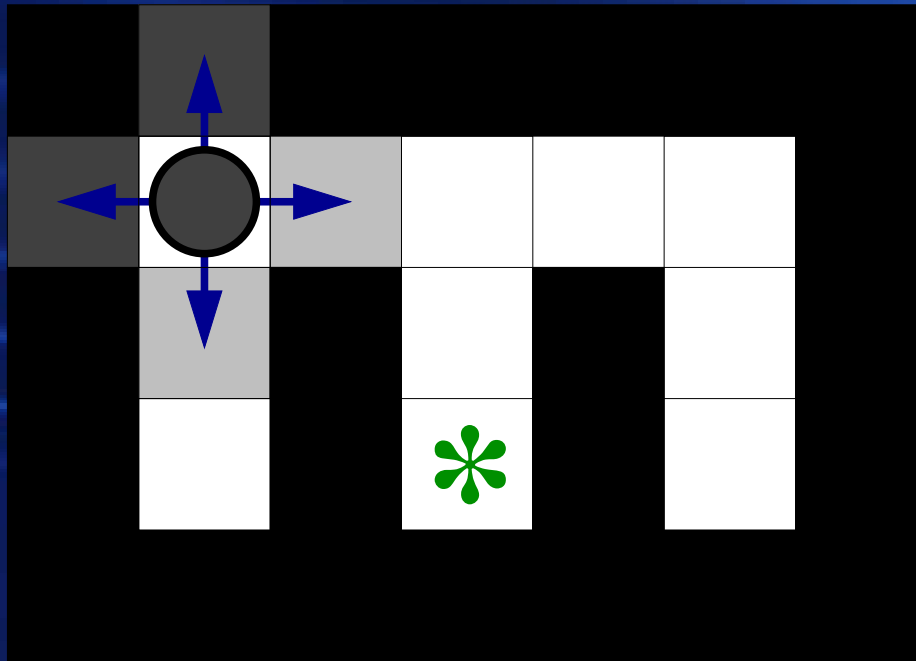
Why Interest in Active Perception?



To deal with same issues for which state-estimation is used, *i.e.*

- RL learns deterministic reactive policies.
- For a partially observable task, no guarantee that a deterministic reactive policy exists.
- Those policies which do exist might be improved by adding active perception.
- Increasing the number of satisficing policies appears to increase the likelihood of convergence to a satisficing policy.

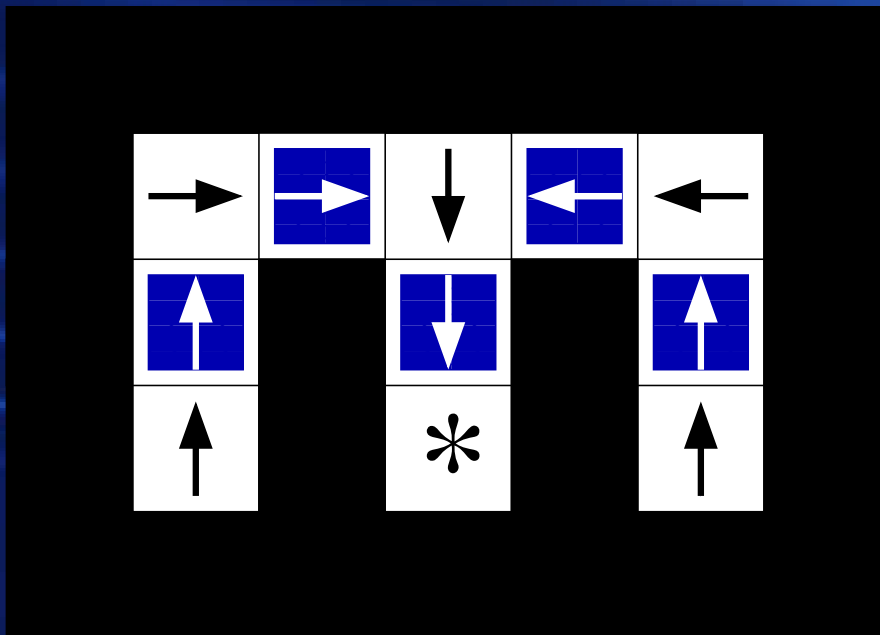
Example - McCallum's M Maze



-  AGENT
-  PERCEPTIONS
-  ACTIONS
-  WALL
-  GOAL STATE

For this agent no reactive policy exists which can reach the goal from all possible locations.

Example - McCallum's M Maze



If an agent with active perception can execute a perceptual action which finds distinguishing features at five locations then a policy can be found.

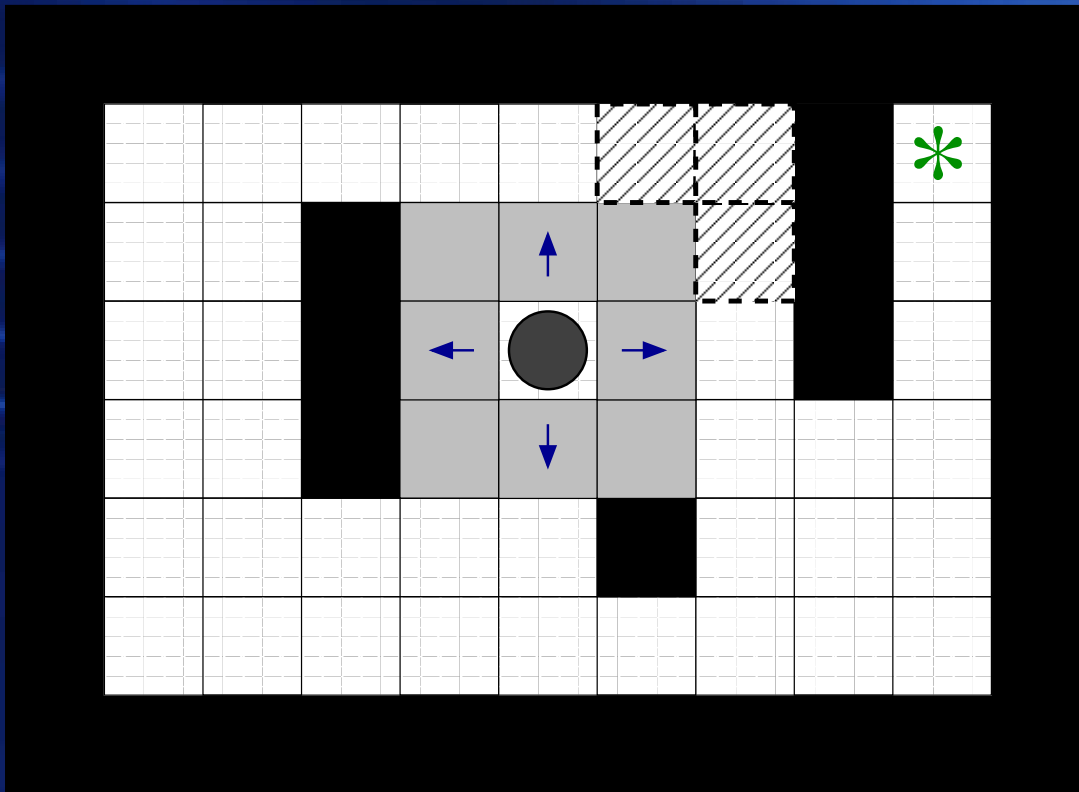
Example - Sutton's Grid World



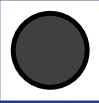





| | | | | | | | | |
|----------|----------|----------|----------|----------|----------|----------|----------|----------|
| → 47 | → 135 | → 71 | → 39 | ↓ 7 | ↓ 7 | ← 151 | | * |
| → 41 | ↑ 144 | | → 40 | ↓ 0 | ↓ 0 | ↑ 148 | | ↑ 189 |
| → 41 | ↑ 148 | | → 41 | ↓ 0 | ↓ 0 | ↓ 20 | | ↑ 157 |
| → 41 | ↓ 20 | | → 9 | → 128 | → 64 | → 36 | → 2 | ↑ 149 |
| → 41 | → 4 | → 2 | → 1 | ↑ 16 | | ↑ 8 | ↓ 0 | ↑ 148 |
| → 233 | → 224 | → 224 | → 224 | ↑ 228 | → 226 | ↑ 225 | → 224 | ↑ 244 |

For the above agent the best satisficing policy takes 416 steps.

Example - Sutton's Grid World



-  OBSTACLE
-  GOAL
-  AGENT
-  FIXED PERCEPTIONS
-  OPTIONAL ACTIVE PERCEPTIONS
-  ACTIONS

Sutton's Grid World – active perception added.

Example - Sutton's Grid World

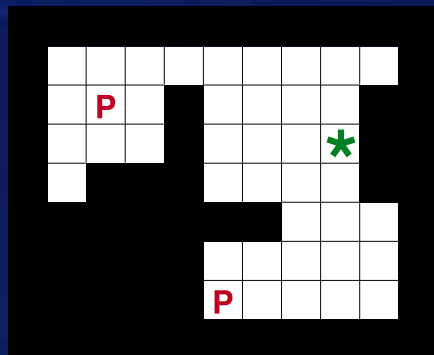
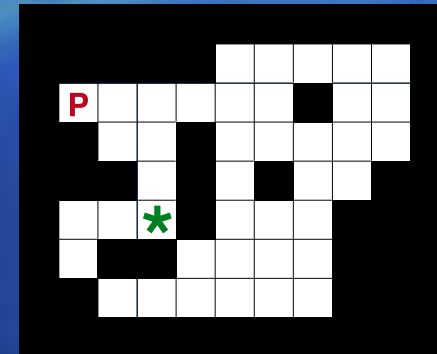
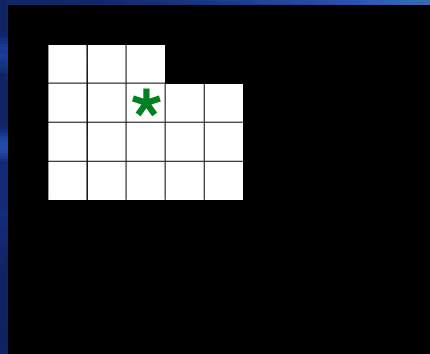
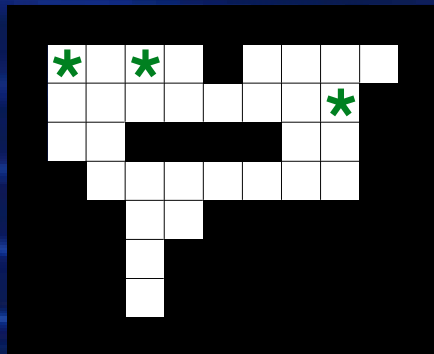


| | | | | | | | | |
|----------|----------|----------|----------|----------|----------|----------|----------|----------|
| → 47 | → 135 | → 71 | ↓ 39 | ↓ 7 | ↓ 7 | ↓ 151 | | * |
| → 41 | ↑ 144 | | → 40 | ↓ 0 | ↓ 0 | ↓ 148 | | ↑ 189 |
| → 41 | ↓ 148 | | → 41 | ↓ 0 | ↓ 0 | ↓ 20 | | ↑ 157 |
| → 41 | ↓ 20 | | → 9 | → 128 | → 64 | → 36 | → 2 | ↑ 149 |
| → 41 | → 4 | → 2 | ↑ 1 | ↑ 16 | | ↑ 8 | ↓ 0 | ↑ 148 |
| ↑ 233 | → 224 | → 224 | → 224 | ↑ 228 | → 226 | ↑ 225 | → 224 | ↑ 244 |

best active perception policy 414 steps (406 physical, 8 perceptual).

Increase in Satisficing Policies

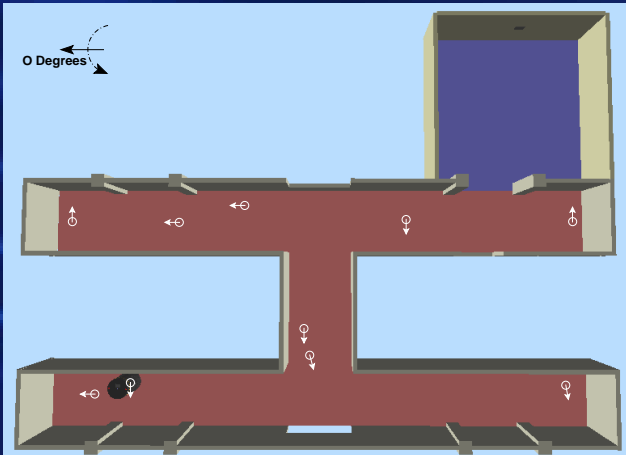
Randomly generated grid worlds



| Agent | Num. Satisficing Policies |
|------------------------|---------------------------|
| Basic | 85% (std. 33%) |
| with Active Perception | 93% (std. 22%) |

Increase in Satisficing Policies

Corridor Navigation



| Agent | Number Satisficing Policies Learnt |
|------------------------|------------------------------------|
| Basic Robot | 20% |
| with Active Perception | 50% |

Summary – Active Perception

Active perception can expand the range of policies that can be constructed such that:

- It can make deterministic reactive policies possible (e.g. McCallum's Maze)



Summary – Active Perception



Active perception can expand the range of policies that can be constructed such that:

- It can make deterministic reactive policies possible (e.g. McCallum's Maze)
- It can allow an agent to learn better deterministic reactive policies.

Summary – Active Perception



Active perception can expand the range of policies that can be constructed such that:

- It can make deterministic reactive policies possible (e.g. McCallum's Maze)
- It can allow an agent to learn better deterministic reactive policies.
- It appears to increase the likelihood of convergence to satisficing deterministic reactive policies.

Active Perception & RL

- In combining active perception with RL I argue that there is no clear distinction between what is a *perceptual action* and what is a *physical action*.



Active Perception & RL



- In combining active perception with RL I argue that there is no clear distinction between what is a *perceptual action* and what is a *physical action*.
- A physical action, such as moving position, will often result in a perceptual change that could reveal hidden information.

Active Perception & RL



- In combining active perception with RL I argue that there is no clear distinction between what is a *perceptual action* and what is a *physical action*.
- A physical action, such as moving position, will often result in a perceptual change that could reveal hidden information.
- Many perceptual actions, e.g. looking around, require physical movement.

Active Perception & RL

- This argument suggest that all actions (physical and perceptual) should be selected by the same process.



Active Perception & RL



- This argument suggest that all actions (physical and perceptual) should be selected by the same process.
- Using the same process to select both action types is also appealing in that it avoids imposing artificial constraints on the policies that can be learnt. (unlike previous RL active perception algorithms; CS-QL [Tan 1991] and the Lion Algorithm [Whitehead 1992]).

Active Perception & RL

- Given the advantages that I presented earlier for active perception – in making policies for partially observable tasks easier to learn – I don't use state-estimation techniques.



Active Perception & RL

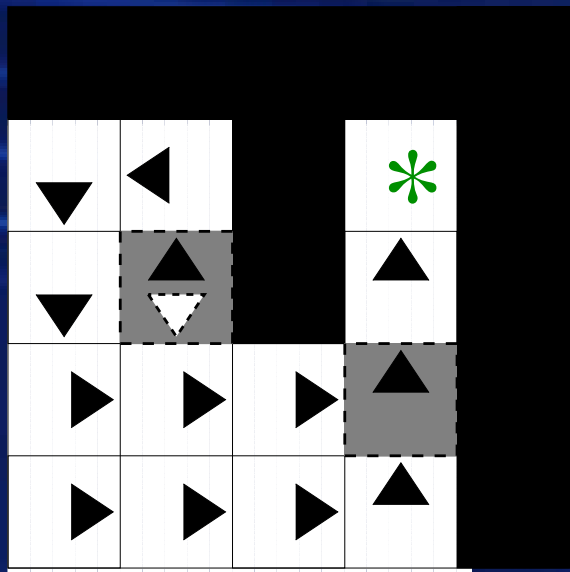


- Given the advantages that I presented earlier for active perception – in making policies for partially observable tasks easier to learn – I don't use state-estimation techniques.
- Instead I consider how to design a RL algorithm that reliably learns deterministic reactive policies in partial observable worlds.

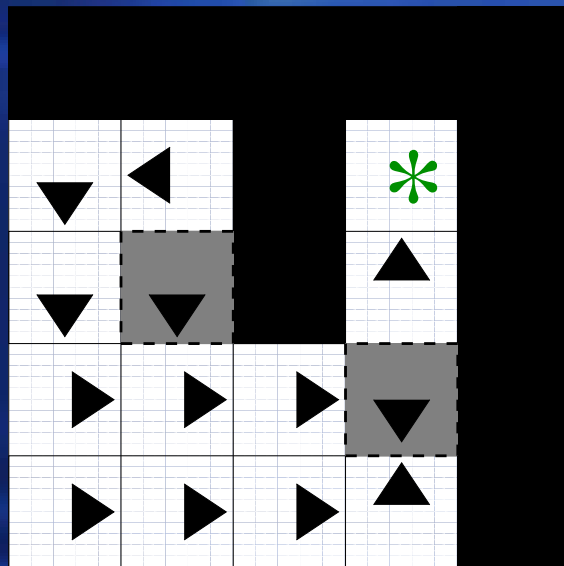
Consistent Exploration

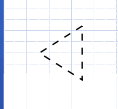



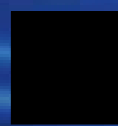


Estimating Value of Explorative Move



Policy Updated



-  Exploratory Move
-  Policy
-  Goal
-  Aliased Location
-  Obstacle

Consistent Exploration $Q(\lambda)$

- CEQ(λ) based on Watkins's $Q(\lambda)$



Consistent Exploration Q(λ)

- CEQ(λ) based on Watkins's Q(λ)
- when exploratory action a_1 is selected for observation o_1 follows modified policy

$$\pi \leftarrow (o_1, a_1)$$



Consistent Exploration Q(λ)

- CEQ(λ) based on Watkins's Q(λ)
- when exploratory action a_1 is selected for observation o_1 follows modified policy
 $\pi \leftarrow (o_1, a_1)$
- when new exploratory action a_2 chosen:



Consistent Exploration Q(λ)



- CEQ(λ) based on Watkins's Q(λ)
- when exploratory action a_1 is selected for observation o_1 follows modified policy
 $\pi \leftarrow (o_1, a_1)$
- when new exploratory action a_2 chosen:
 - clear eligibility trace

Consistent Exploration Q(λ)



- CEQ(λ) based on Watkins's Q(λ)
- when exploratory action a_1 is selected for observation o_1 follows modified policy $\pi \leftarrow (o_1, a_1)$
- when new exploratory action a_2 chosen:
 - clear eligibility trace
 - change policy followed to $\pi \leftarrow (o_2, a_2)$

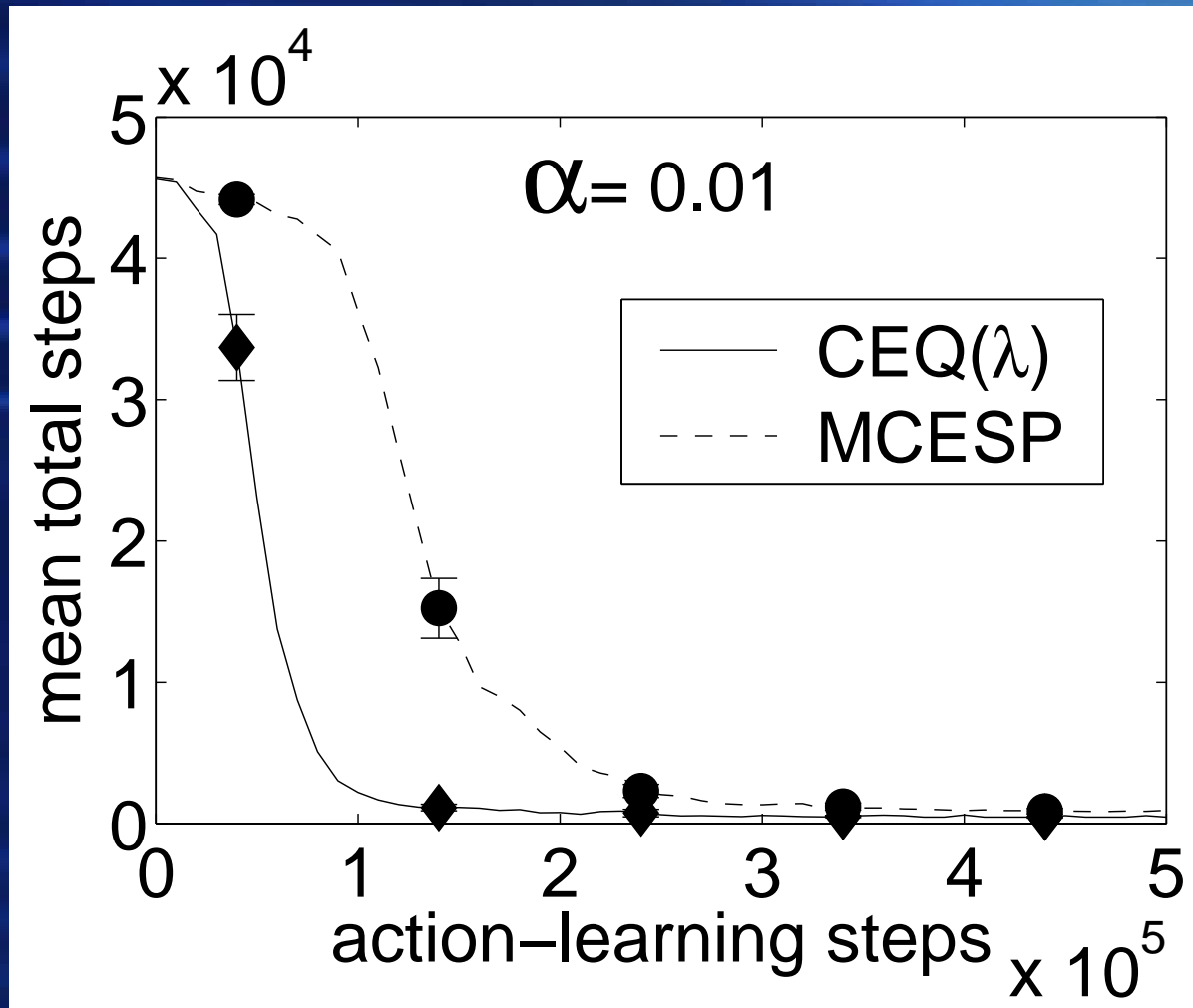
Comparison with SARSA(λ)



Mean number of satisficing policies for each agent-algorithm combination as learnt on 10 random grid worlds

| | | Algorithm | | |
|-------|-------------------|------------------|--------------------|---------------|
| | | CEQ(λ) | SARSA(λ) | U-Tree |
| Agent | | | | |
| | Basic | 90% \pm 28% | 80% \pm 37% | 48% \pm 45% |
| | Active Perception | 99% \pm 6% | 88% \pm 29% | |

Comparison with MCESP



Conclusions

- Demonstrated that adding active perception to agents is a useful approach when applying RL in partially observable environments.



Conclusions

- Demonstrated that adding active perception to agents is a useful approach when applying RL in partially observable environments.
- That the concept of Consistent Exploration can result in algorithms which learn deterministic reactive policies more reliably in partially observable environments.



Future Directions

- Development of a theory of learning deterministic policies in partially observable domains.



Future Directions

- Development of a theory of learning deterministic policies in partially observable domains.
 - Is it possible to identify tasks where a deterministic policy exists?



Future Directions



- Development of a theory of learning deterministic policies in partially observable domains.
 - Is it possible to identify tasks where a deterministic policy exists?
- Comparison of approach with PSR, recurrent-networks (LSTM), etc.

Future Directions



- Development of a theory of learning deterministic policies in partially observable domains.
 - Is it possible to identify tasks where a deterministic policy exists?
- Comparison of approach with PSR, recurrent-networks (LSTM), etc.
- Look at a larger range of tasks (only considered minimum. cost to goal so far).

Future Directions



- Development of a theory of learning deterministic policies in partially observable domains.
 - Is it possible to identify tasks where a deterministic policy exists?
- Comparison of approach with PSR, recurrent-networks (LSTM), etc.
- Look at a larger range of tasks (only considered minimum. cost to goal so far).

Thank You

Reinforcement Learning (RL)



Reinforcement Learning (RL)



RL algorithms are distinguished from other machine learning techniques by:

- the use of trial and error to search the action space
- reward signals can be delay and thus attributable to earlier actions (or action sequences).