Partially Observable Markov Decision Processes

Chapter 17

Mausam

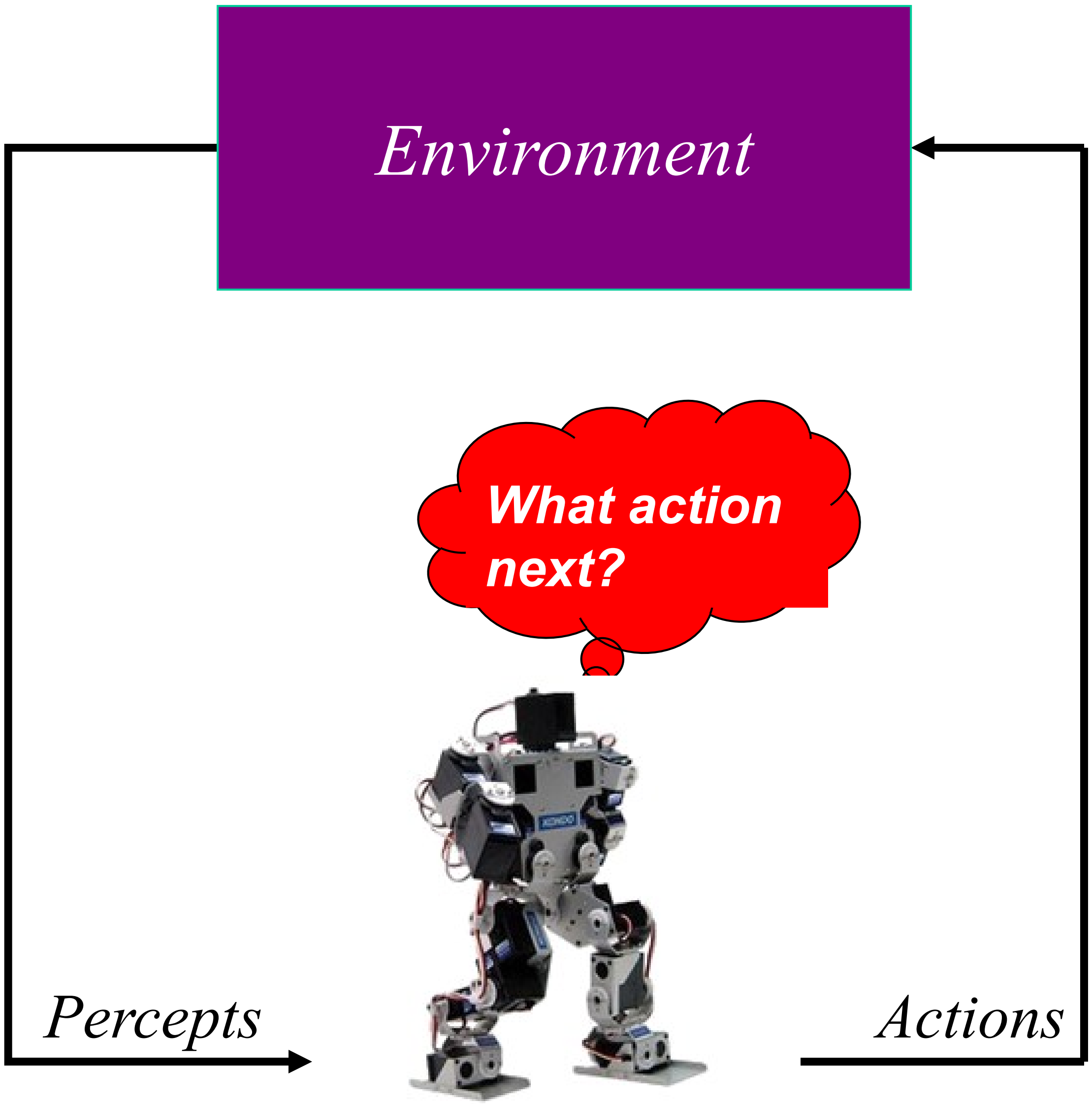
(Based on slides by Dieter Fox, Lee Wee Sun)

Stochastic Planning: MDPs

Static

Fully Observable

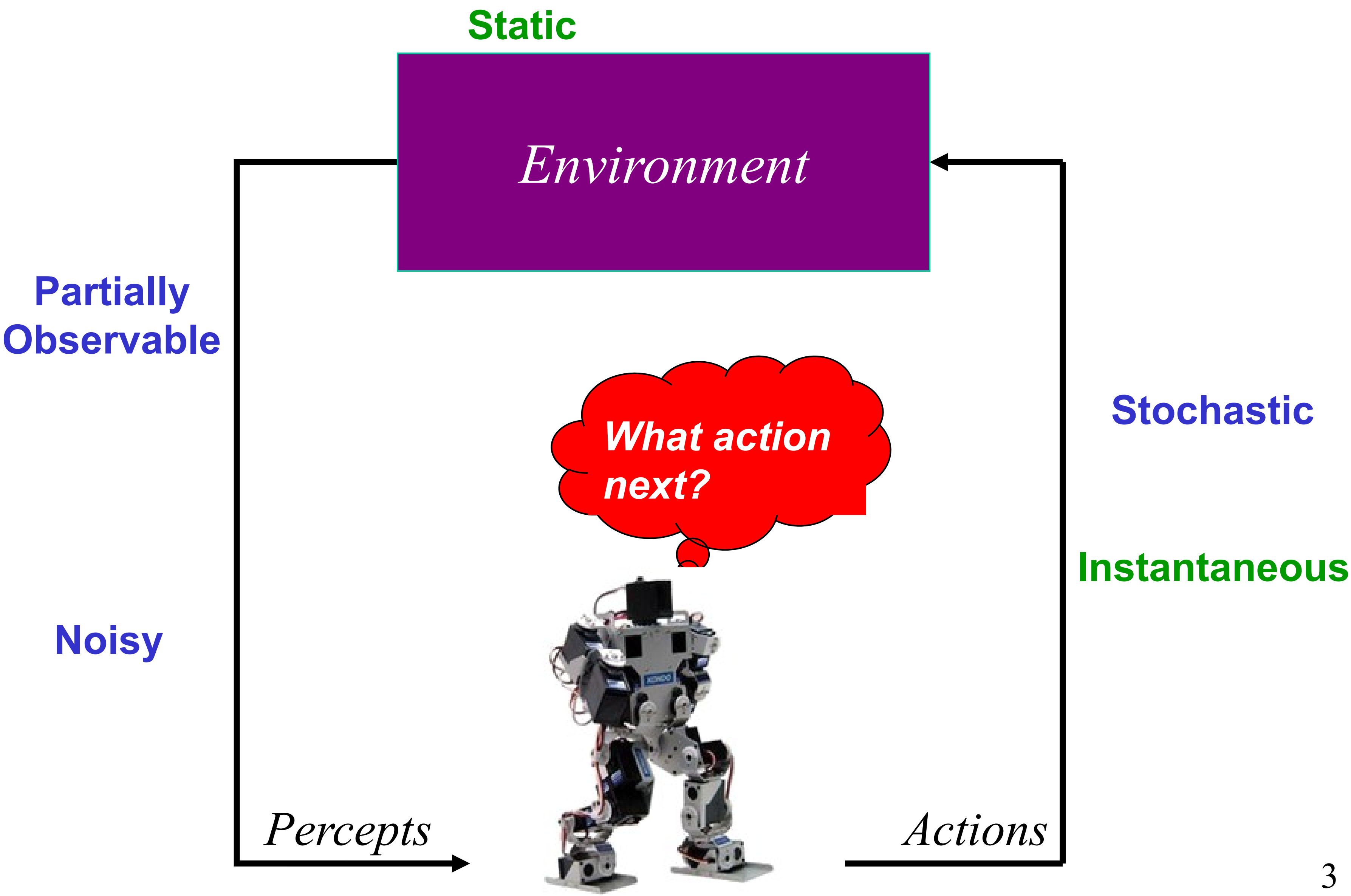
Perfect



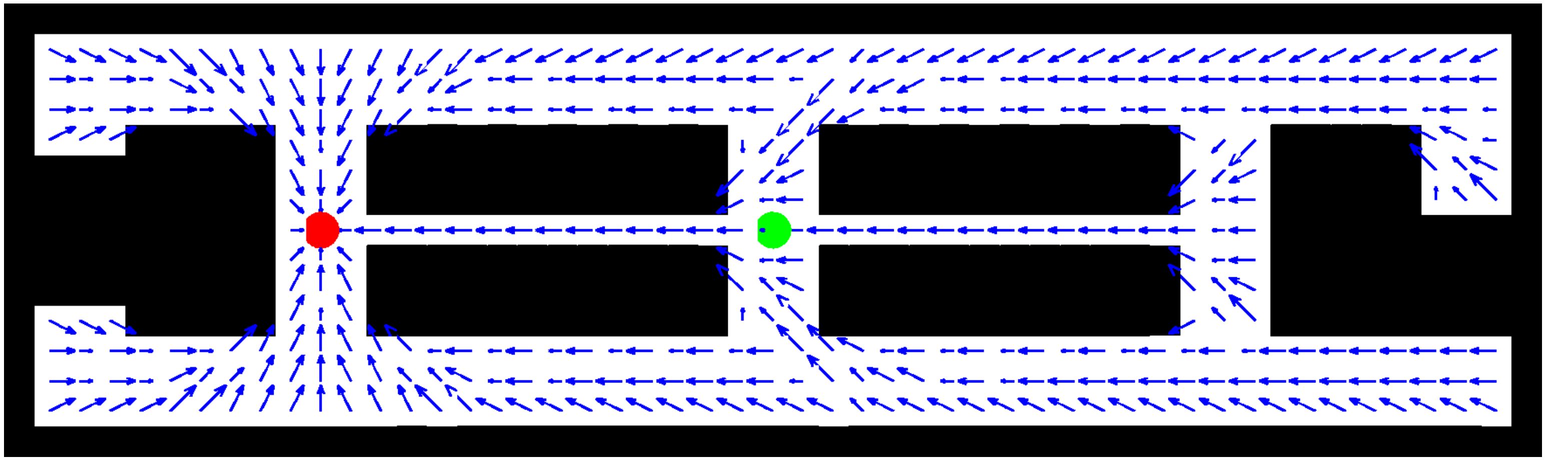
Stochastic

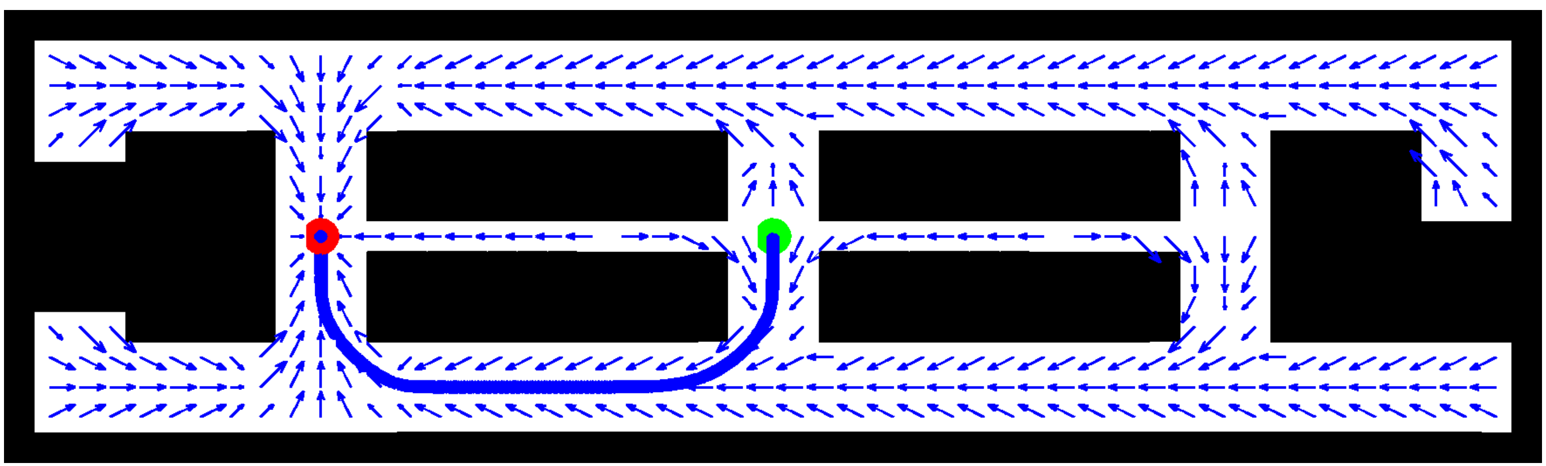
Instantaneous

Partially Observable MDPs

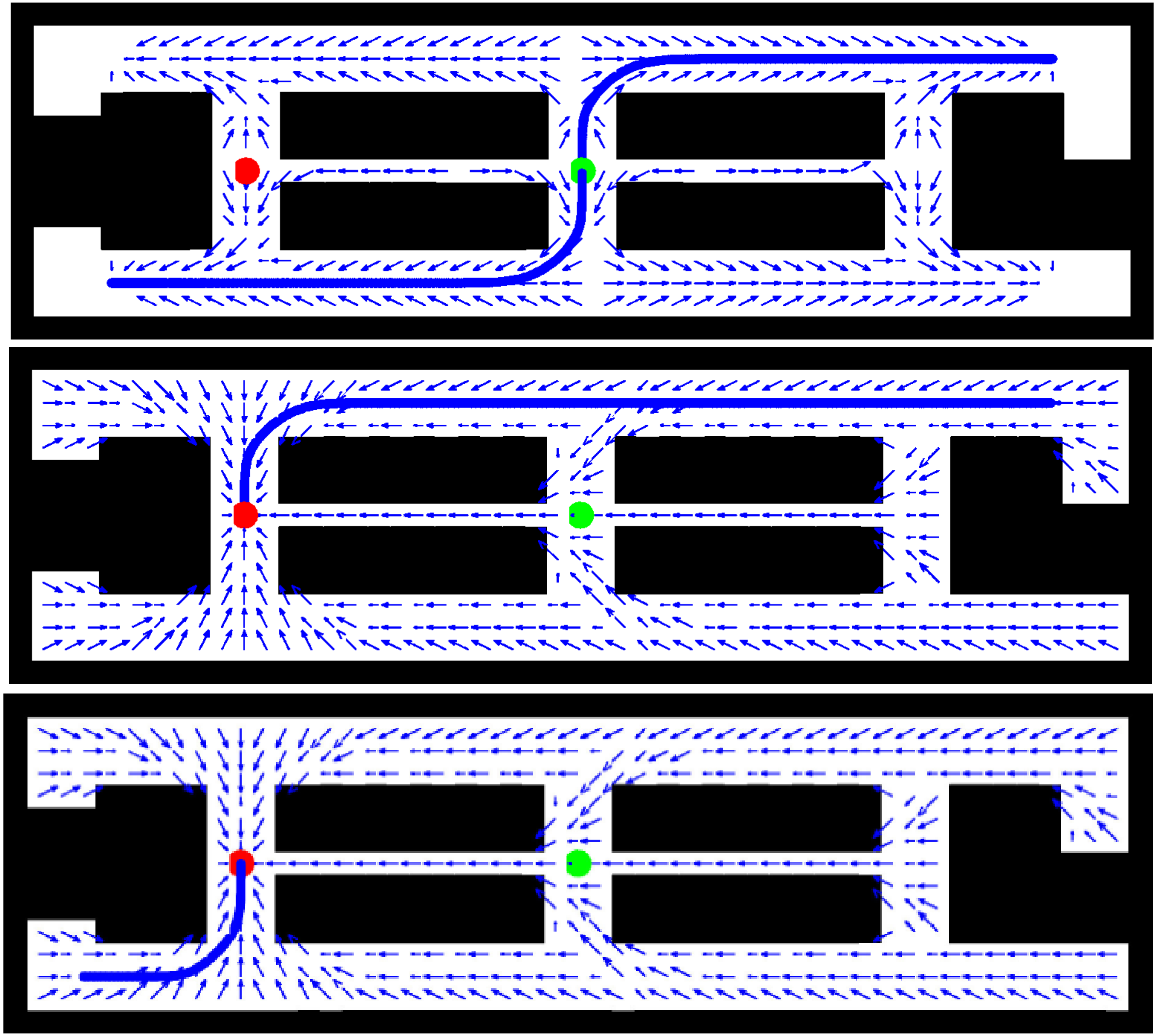


Stochastic, Fully Observable





Stochastic, Partially Observable



POMDPs

- In POMDPs we apply the very same idea as in MDPs.

- Since the state is not observable,

  the agent has to make its decisions based on the belief state which is a posterior distribution over states.

- Let b be the belief of the agent about the current state
- POMDPs compute a value function over belief space:

$$V_T(b) \quad = \quad \max_a \left[r(b,a) + \gamma \int V_{T-1}(b') p(b' \mid b, a) \, db' \right]$$

POMDPs

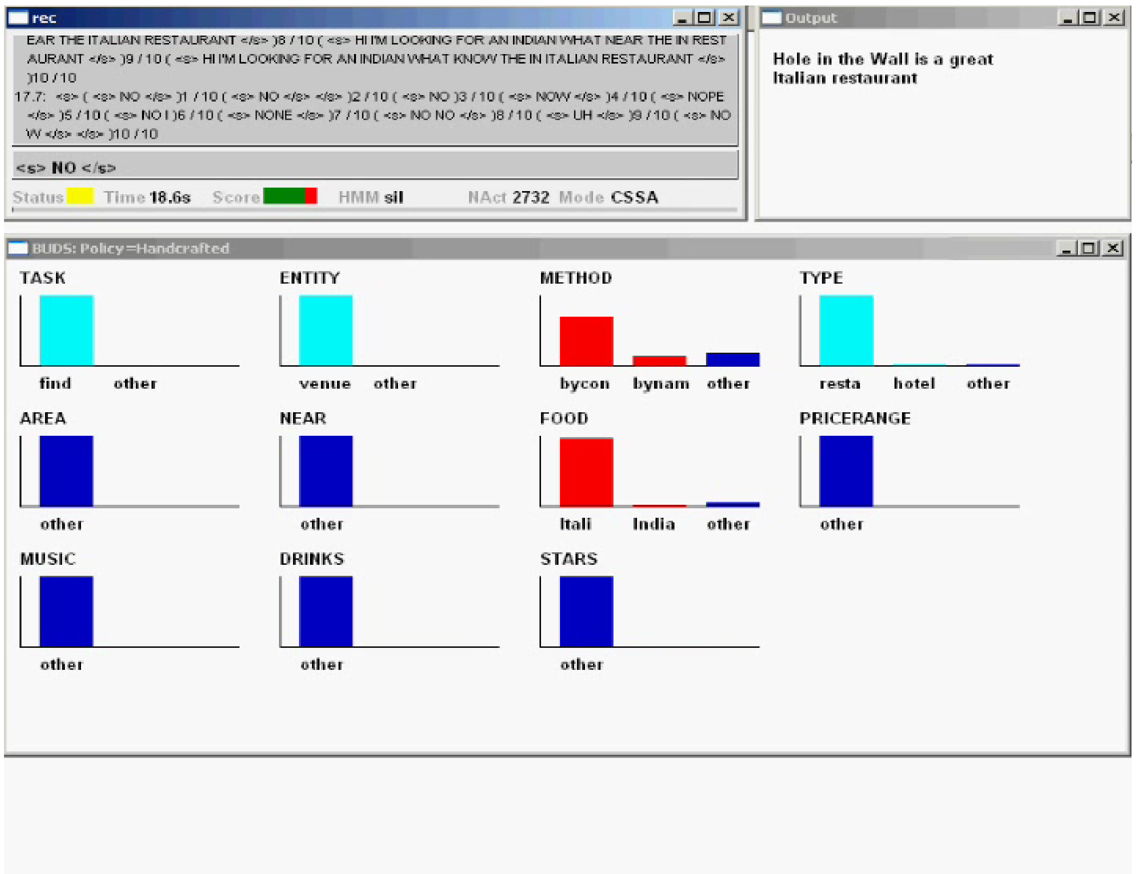
- Each belief is a probability distribution,
  - value fn is a function of an entire probability distribution.
- Problematic, since probability distributions are continuous.
- Also, we have to deal with huge complexity of belief spaces.

- For finite worlds with finite state, action, and observation spaces and finite horizons,
  - we can represent the value functions by piecewise linear functions.

Applications

- Robotic control
  - helicopter maneuvering, autonomous vehicles
  - Mars rover - path planning, oversubscription planning
  - elevator planning
- Game playing - backgammon, tetris, checkers
- Neuroscience
- Computational Finance, Sequential Auctions
- Assisting elderly in simple tasks
- Spoken dialog management
- Communication Networks – switching, routing, flow control
- War planning, evacuation planning

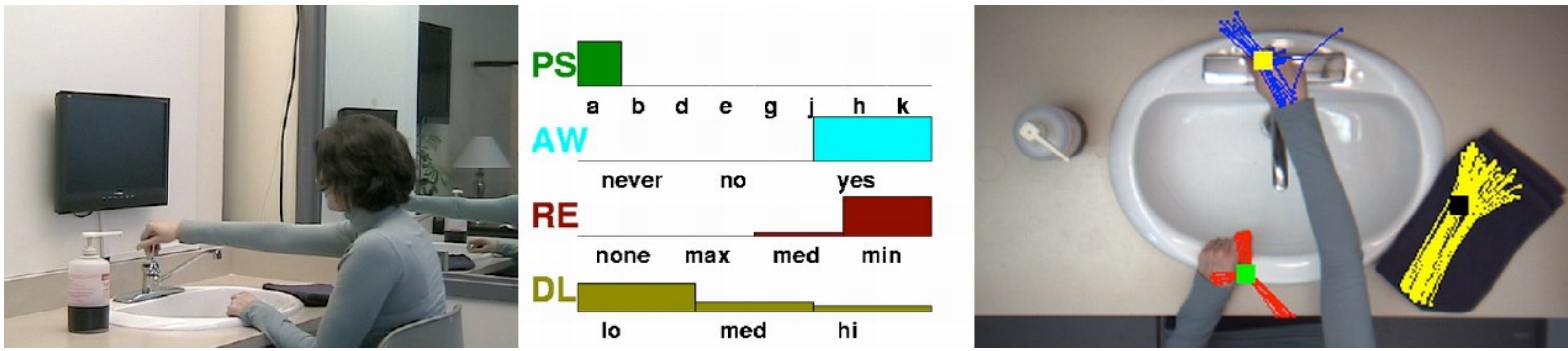
Dialog Systems



- Aim: Find out what the person wants
- Actions: Ask appropriate questions
- Observations: Output of speech recognizer

Williams & Young, 2007

Assistive Technology

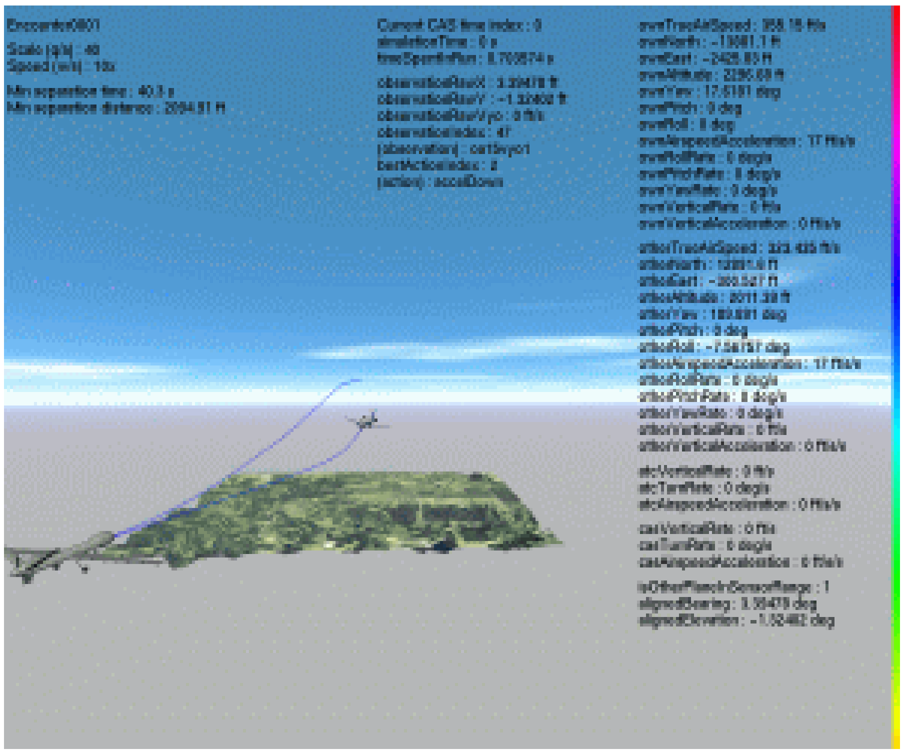


- Aim: Assist person with dementia in handwashing
- Actions: Prompt the person with suggestions when appropriate
- Observations: Video of activity

POMDP

Hoey, Von Bertoldi, Poupart, Mihailidis 2007

Aircraft Collision Avoidance System



- **Aim**: Avoid collision with nearby aircraft
- **Actions**: Maneuver the UAV
- **Observations**: Limited view sensors

[Image from http://web.mit.edu/temizer/www/selim/](http://web.mit.edu/temizer/www/selim/)

Temitzer, Kochenderfer, Kaelbling, Lozano-Perez, Kuchar 2010
Bai, Hsu, Lee, Kochenderfer 2011

- Commonalities in the examples
  - Need to learn, estimate or track the current state of the system from history of actions and observations
  - Based on current state estimate, select an action that leads to good outcomes, not just currently but also in future (planning)

Powerful but Intractable

- Partially Observable Markov Decision Process (POMDP) is a very powerful modeling tool
- But with great power
  … comes great intractability!

No known way to solve it quickly

No small policy

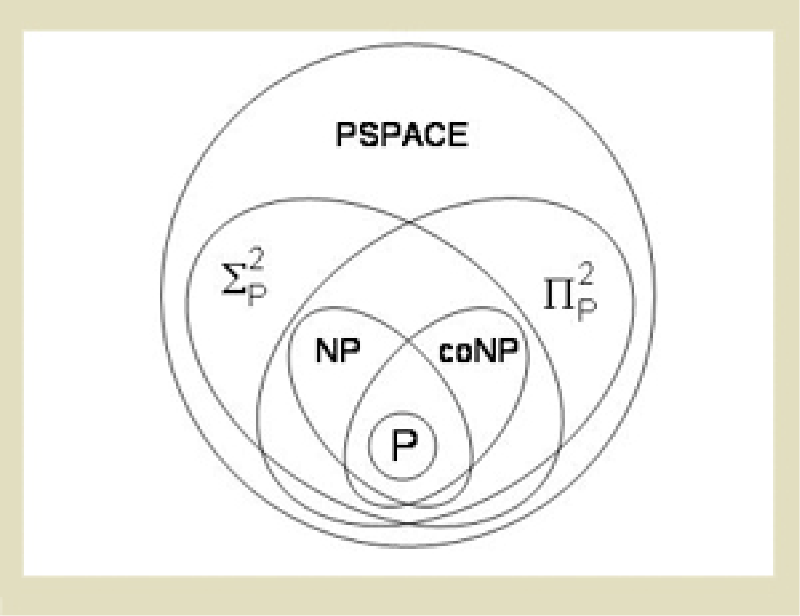
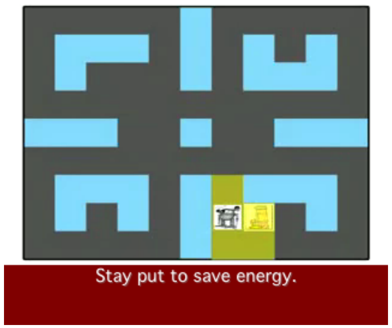


Image from http://ocw.mit.edu/courses/mathematics/18-405j-advanced-complexity-theory-fall-2001/

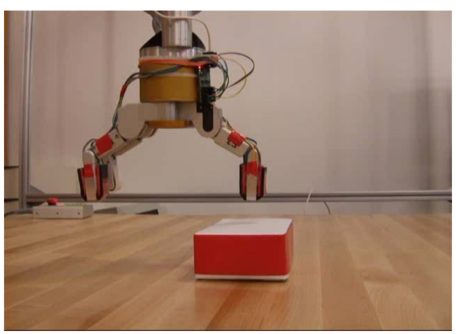
State Space Size

- Moderate state space size



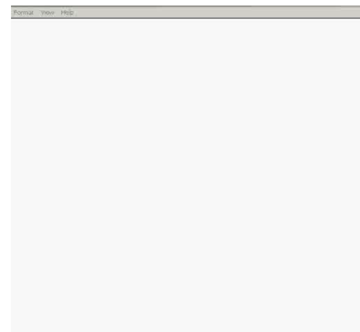
tracking

navigation

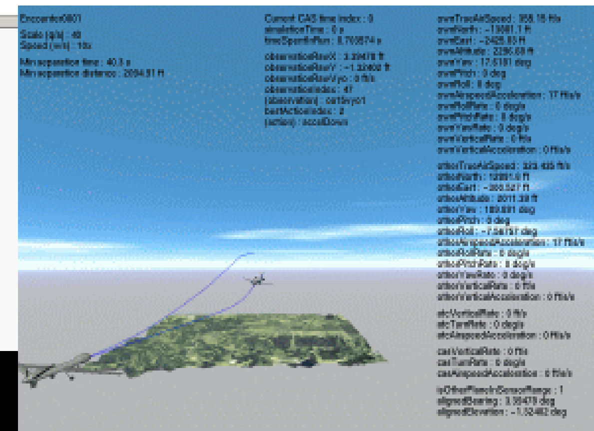


simple grasping

- Very large, continuous state spaces



dialog



collision avoidance

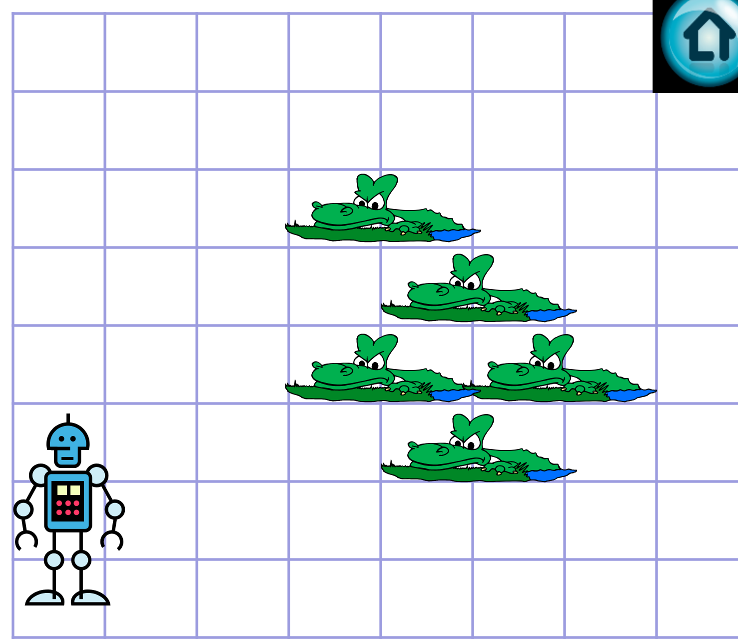
assistance

- Exact belief and policy evaluation
- Approximate belief and policy evaluation

Partially Observable Markov Decision Process

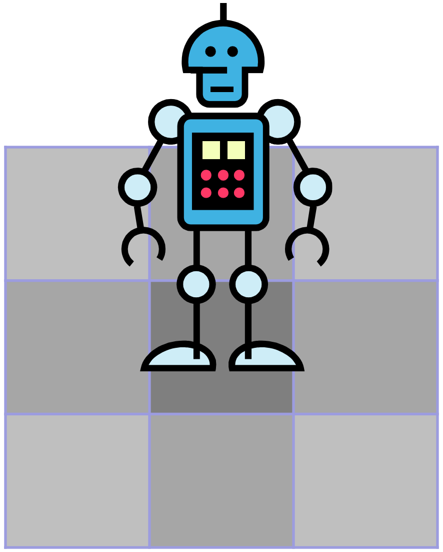
- Partially Observable Markov Decision Process (POMDP) $<S,A,T,R,\Omega,O>$
- Observations Ω: Set of possible observations
    - Navigation example:
        - Set of locations output by GPS sensor

Robot navigation



- POMDP $\langle S,A,T,R,\Omega,O\rangle$
- Observation probability function O: Probability of observing o in state s' when previous action is a
  - $O(o, a, s') = \Pr(o \mid a, s')$
  - Navigation example:
    - Darker shade, higher probability

Robot navigation

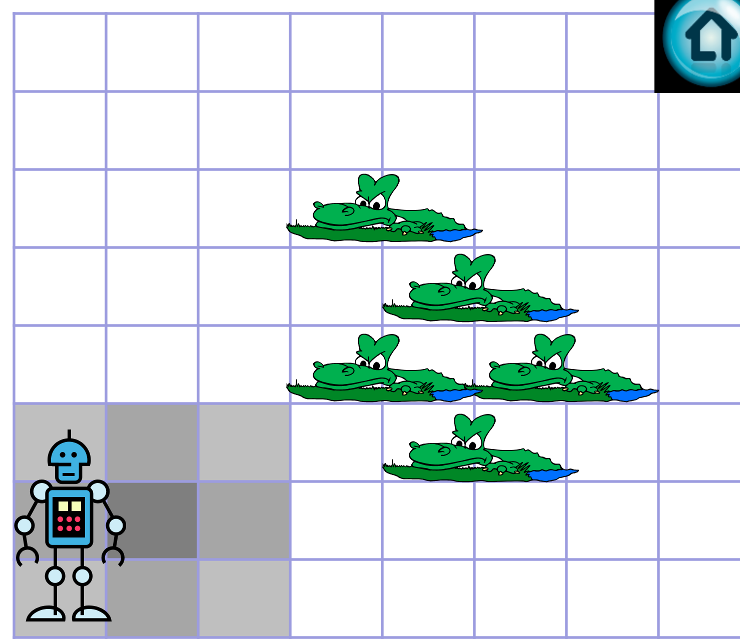


POMDP

Belief

- POMDP $<S,A,T,R,\Omega,O>$
- Belief b : Probability of state s
  - $b(s) = \Pr(s)$
  - Navigation example:
    - Exact position of robot unknown
    - Only have probability distribution of positions, obtained through sensor readings

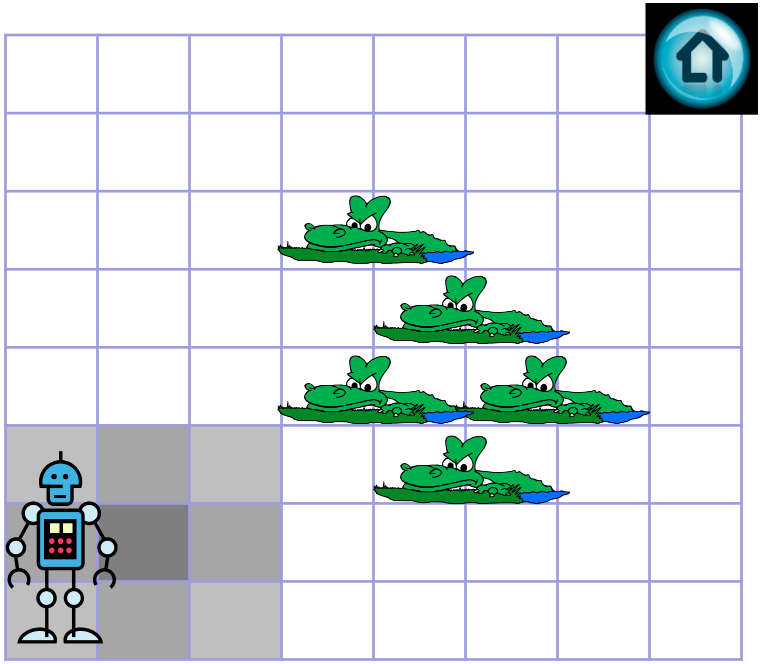
Robot navigation



POMDP Policy

- POMDP $<S,A,T,R,\Omega,O>$
- Policy π : Function from belief to action
  - $a = \pi(b)$
  - Navigation example:
    - Which way to move, based on current belief

Robot navigation



- POMDP *<S,A,T,R,Ω,O>*
- *R(a,b)* : Expected reward for taking action *a* when belief is *b*

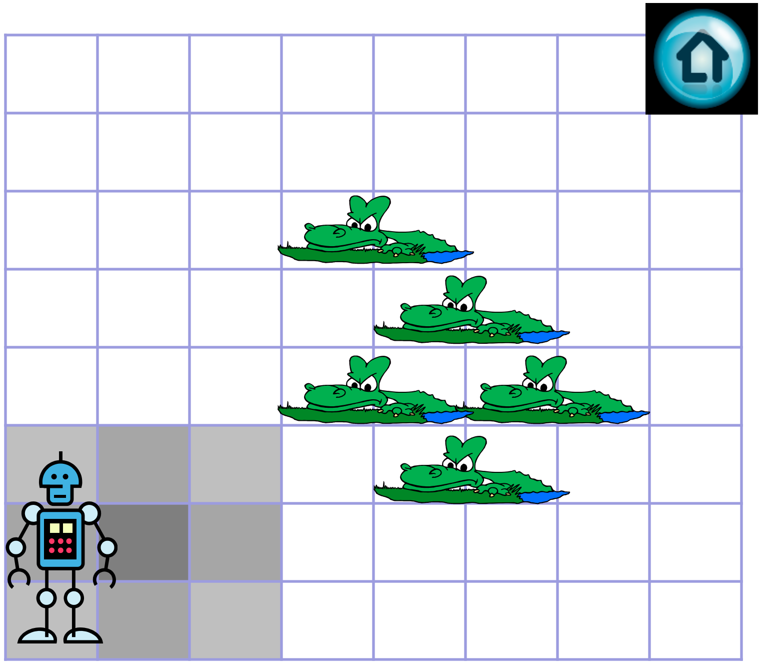
$$R(a,b) = E(R(s,a,s))$$

$$= \sum_i \sum_j T(s_i,a,s_j) b(s_i) R(s_i,a,s_j)$$

- Optimal Policy π^* : Function π that maximizes

$$\sum_{t=0}^{\infty} \gamma^t R(\pi(b_t), b_t)$$

Robot navigation



POMDP

Value Function

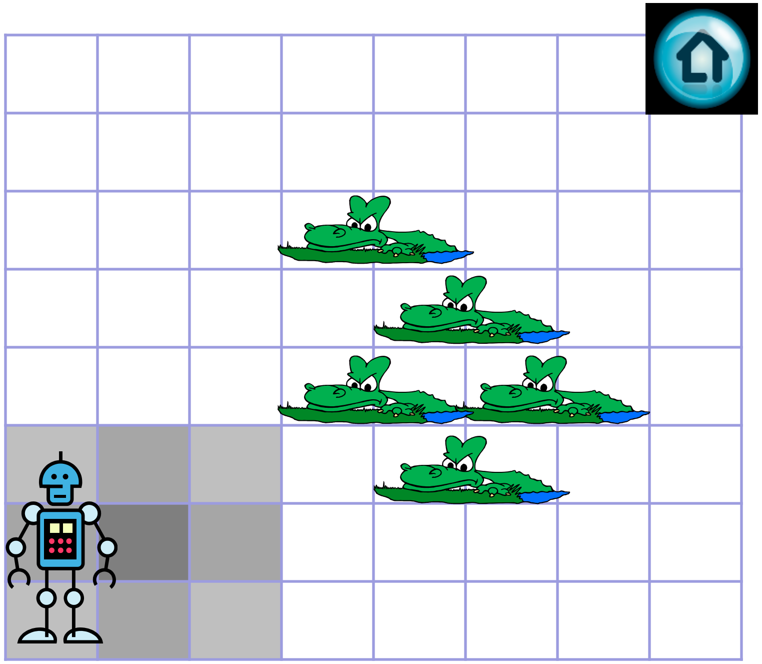
- POMDP *<S,A,T,R,Ω,O>*
- Value function for π : Expected return for starting from belief *b*

$$V^{\pi}(b) = \sum_{t=0}^{\infty} \gamma^t R(\pi(b_t), b_t),$$

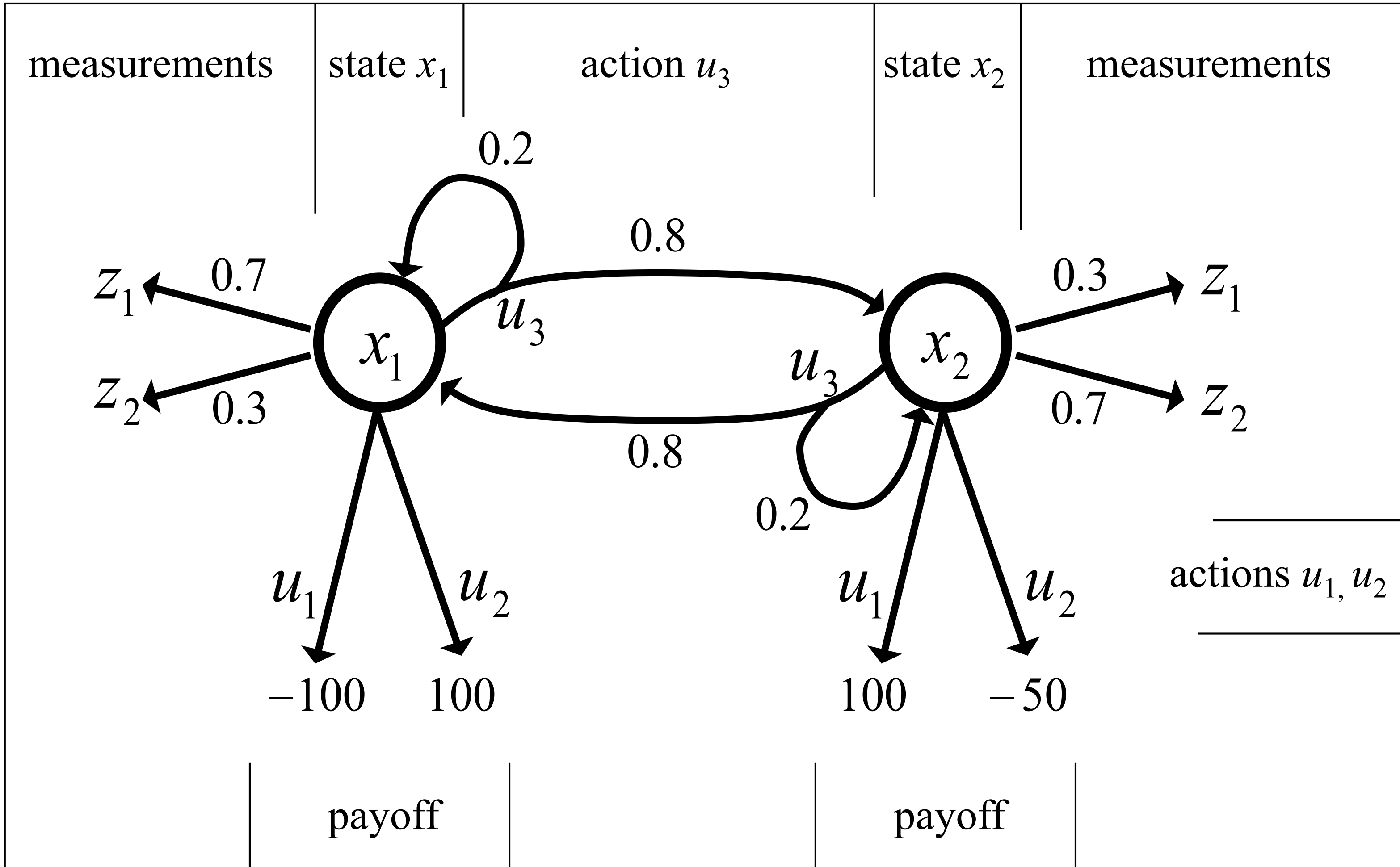
$$\textbf{with } b_0 = b$$

- Optimal value function *V* :* Value function associated with an optimal policy π*

Robot navigation



An Illustrative Example



The Parameters of the Example

- The actions u_1 and u_2 are terminal actions.
- The action u_3 is a sensing action that potentially leads to a state transition.
- The horizon is finite and γ=1.

$$
\begin{aligned}
r(x_1, u_1) &= -100 & \quad r(x_2, u_1) &= +100 \\
r(x_1, u_2) &= +100 & \quad r(x_2, u_2) &= -50 \\
r(x_1, u_3) &= -1 & \quad r(x_2, u_3) &= -1
\end{aligned}
$$

$$
\begin{aligned}
p(x'_1 | x_1, u_3) &= 0.2 & \quad p(x'_2 | x_1, u_3) &= 0.8 \\
p(x'_1 | x_2, u_3) &= 0.8 & \quad p(z'_2 | x_2, u_3) &= 0.2
\end{aligned}
$$

$$
\begin{aligned}
p(z_1 | x_1) &= 0.7 & \quad p(z_2 | x_1) &= 0.3 \\
p(z_1 | x_2) &= 0.3 & \quad p(z_2 | x_2) &= 0.7
\end{aligned}
$$

Payoff in POMDPs

- In MDPs, the payoff (or return) depended on the state of the system.
- In POMDPs, however, the true state is not exactly known.
- Therefore, we compute the **expected payoff** by **integrating over all states**:

$$\begin{aligned} r(b,u) &= E_x[r(x,u)] \\ &= \int r(x,u)p(x)\ dx \\ &= p_1\, r(x_1,u) + p_2\, r(x_2,u) \end{aligned}$$

Payoffs in Our Example (1)

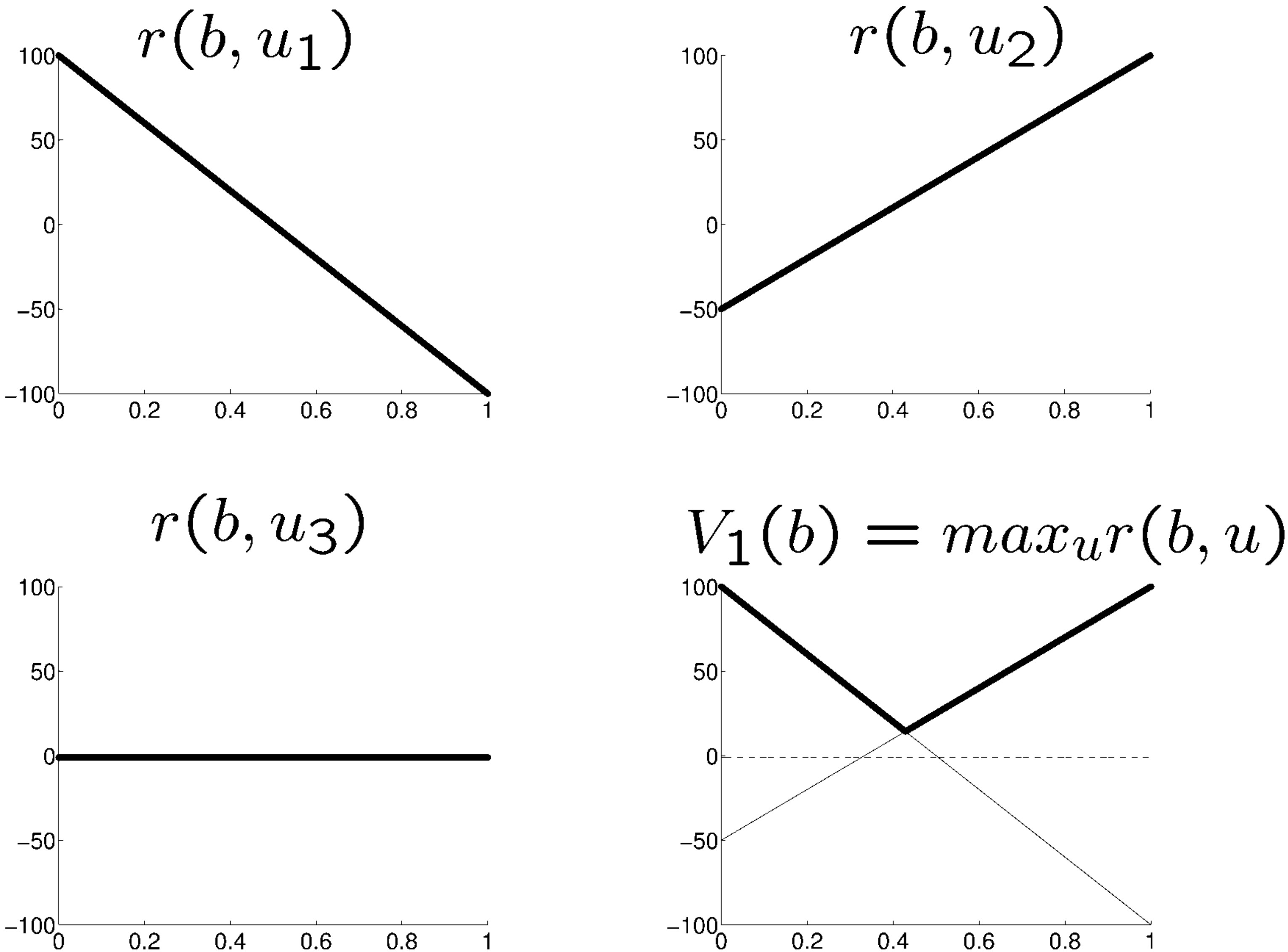
- If we are totally certain that we are in state x_1 and execute action u_1, we receive a reward of -100
- If, on the other hand, we definitely know that we are in x_2 and execute u_1, the reward is +100.
- In between it is the linear combination of the extreme values weighted by the probabilities

$$
\begin{aligned}
r(b, u_1) &= -100\, p_1 + 100\, p_2 \\
&= -100\, p_1 + 100\, (1 - p_1)
\end{aligned}
$$

$$
r(b, u_2) = 100\, p_1 - 50\, (1 - p_1)
$$

$$
r(b, u_3) = -1
$$

Payoffs in Our Example (2)



The Resulting Policy for T=1

- Given we have a finite POMDP with T=1, we would use $V_1(b)$ to determine the optimal policy.
- In our example, the optimal policy for T=1 is

$$\pi_1(b) = \begin{cases} u_1 & \text{if } p_1 \le \frac{3}{7} \\ \\ u_2 & \text{if } p_1 > \frac{3}{7} \end{cases}$$

- This is the upper thick graph in the diagram.

Piecewise Linearity, Convexity

- The resulting value function $V_1(b)$ is the maximum of the three functions at each point

$$\begin{aligned} V_1(b) &= \max_u r(b,u) \\ &= \max \left\{ \begin{array}{rr} -100\, p_1 & +100\,(1-p_1) \\ 100\, p_1 & -50\,(1-p_1) \\ -1 & \end{array} \right\} \end{aligned}$$

- It is piecewise linear and convex.

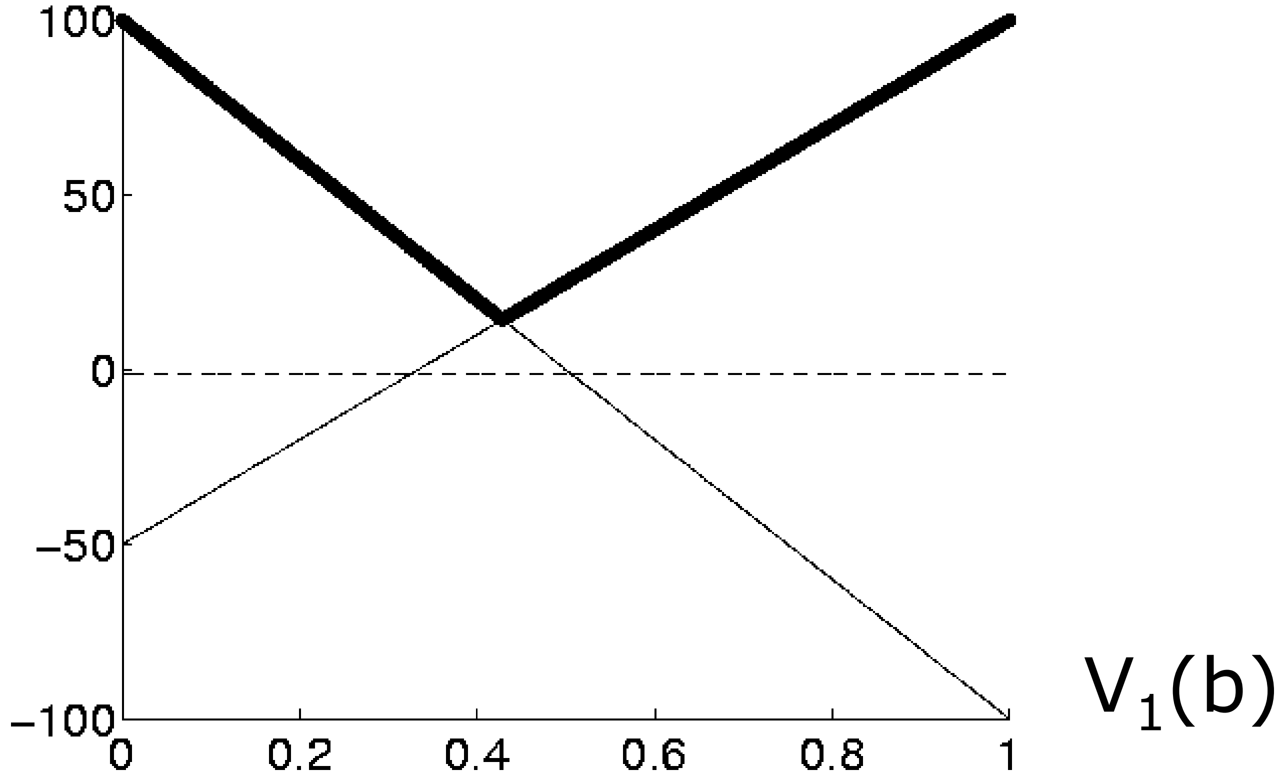
Pruning

- If we carefully consider $V_1(b)$, we see that only the first two components contribute.
- The third component can therefore safely be pruned away from $V_1(b)$.

$$V_1(b) = \max \left\{ \begin{array}{ll} -100\, p_1 & +100\,(1-p_1) \\ 100\, p_1 & -50\,(1-p_1) \end{array} \right\}$$

Increasing the Time Horizon

- Assume the robot can make an observation before deciding on an action.



Increasing the Time Horizon

- Assume the robot can make an observation before deciding on an action.
- Suppose the robot perceives z_1 for which $p(z_1 \mid x_1)=0.7$ and $p(z_1 \mid x_2)=0.3$.
- Given the observation z_1 we update the belief using Bayes rule.

$$p'_1 = \frac{0.7\, p_1}{p(z_1)}$$

$$p'_2 = \frac{0.3(1 - p_1)}{p(z_1)}$$

$$p(z_1) = 0.7\, p_1 + 0.3(1 - p_1) = 0.4\, p_1 + 0.3$$

Increasing the Time Horizon

- Assume the robot can make an observation before deciding on an action.
- Suppose the robot perceives z_1 for which $p(z_1 \mid x_1)=0.7$ and $p(z_1 \mid x_2)=0.3$.
- Given the observation z_1 we update the belief using Bayes rule.
- Thus $V_1(b \mid z_1)$ is given by

$$
\begin{aligned}
V_1(b \mid z_1) &= \max \left\{ \begin{array}{llll} -100 \cdot \frac{0.7\, p_1}{p(z_1)} & +100 \cdot \frac{0.3\,(1-p_1)}{p(z_1)} \\ \\ 100 \cdot \frac{0.7\, p_1}{p(z_1)} & -50 \cdot \frac{0.3\,(1-p_1)}{p(z_1)} \end{array} \right\} \\
&= \frac{1}{p(z_1)} \max \left\{ \begin{array}{ll} -70\, p_1 & +30\,(1-p_1) \\ 70\, p_1 & -15\,(1-p_1) \end{array} \right\}
\end{aligned}
$$

Expected Value after Measuring

- Since we do not know in advance what the next measurement will be, we have to compute the expected belief

$$\bar{V}_1(b) = E_z[V_1(b \mid z)] = \sum_{i=1}^{2} p(z_i) V_1(b \mid z_i)$$

$$= \sum_{i=1}^{2} p(z_i) V_1\left(\frac{p(z_i \mid x_1) p_1}{p(z_i)} \right)$$

$$= \sum_{i=1}^{2} V_1\left(p(z_i \mid x_1) p_1 \right)$$

Expected Value after Measuring

- Since we do not know in advance what the next measurement will be, we have to compute the expected belief

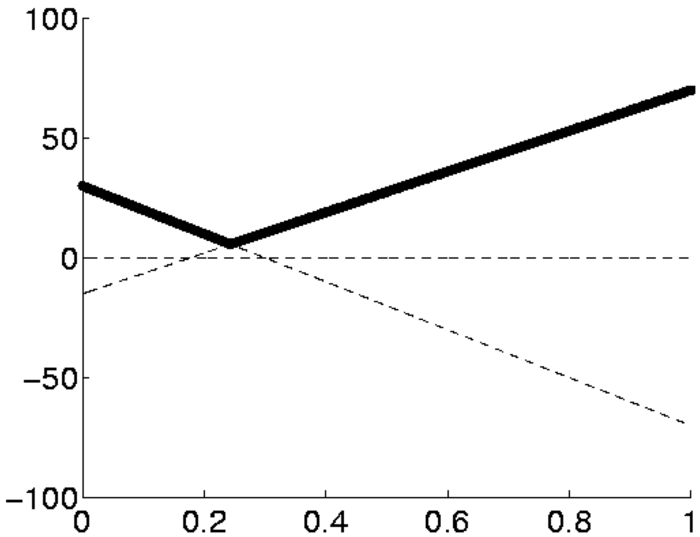
$$
\begin{aligned}
\bar{V}_1(b) &= E_z[V_1(b \mid z)] \\
&= \sum_{i=1}^{2} p(z_i)\, V_1(b \mid z_i) \\
&= \max\left\{ \begin{array}{ll} -70\, p_1 & +30\,(1-p_1) \\ 70\, p_1 & -15\,(1-p_1) \end{array} \right\} \\
&\quad + \max\left\{ \begin{array}{ll} -30\, p_1 & +70\,(1-p_1) \\ 30\, p_1 & -35\,(1-p_1) \end{array} \right\}
\end{aligned}
$$

Resulting Value Function

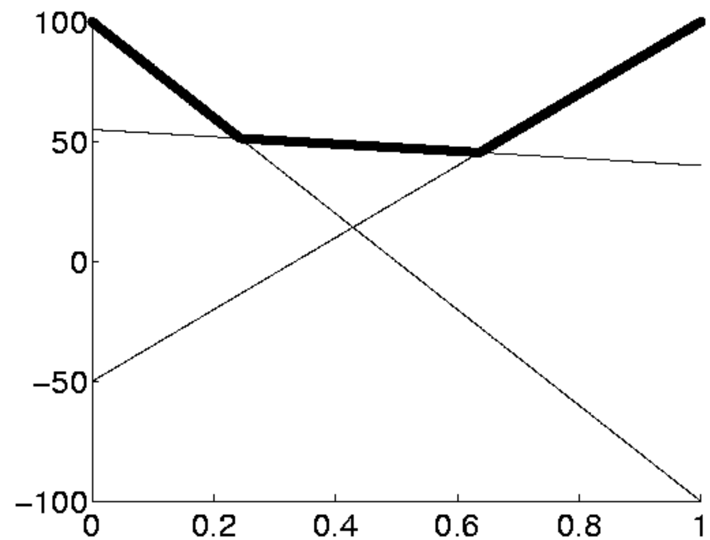
- The four possible combinations yield the following function which then can be simplified and pruned.

$$
\begin{aligned}
\bar{V}_1(b) &= \max \left\{ \begin{array}{llll} -70\,p_1 & +30\,(1-p_1) & -30\,p_1 & +70\,(1-p_1) \\ -70\,p_1 & +30\,(1-p_1) & +30\,p_1 & -35\,(1-p_1) \\ +70\,p_1 & -15\,(1-p_1) & -30\,p_1 & +70\,(1-p_1) \\ +70\,p_1 & -15\,(1-p_1) & +30\,p_1 & -35\,(1-p_1) \end{array} \right\} \\
&= \max \left\{ \begin{array}{ll} -100\,p_1 & +100\,(1-p_1) \\ +40\,p_1 & +55\,(1-p_1) \\ +100\,p_1 & -50\,(1-p_1) \end{array} \right\}
\end{aligned}
$$

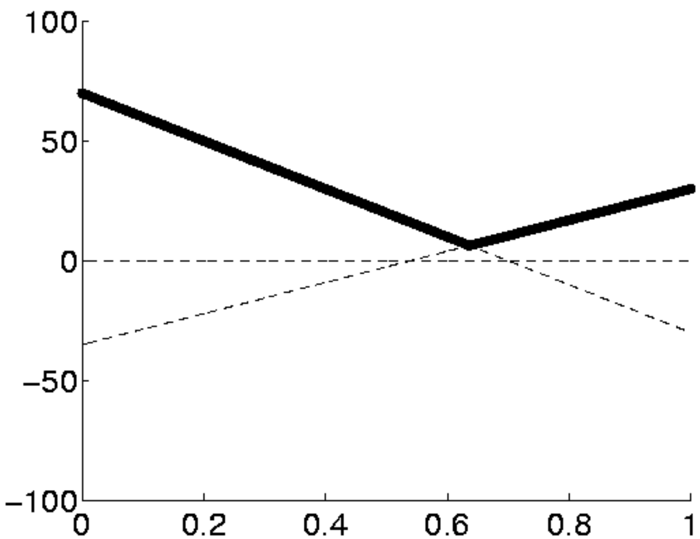
Value Function



$p(z_1)\ V_1(b|z_1)$



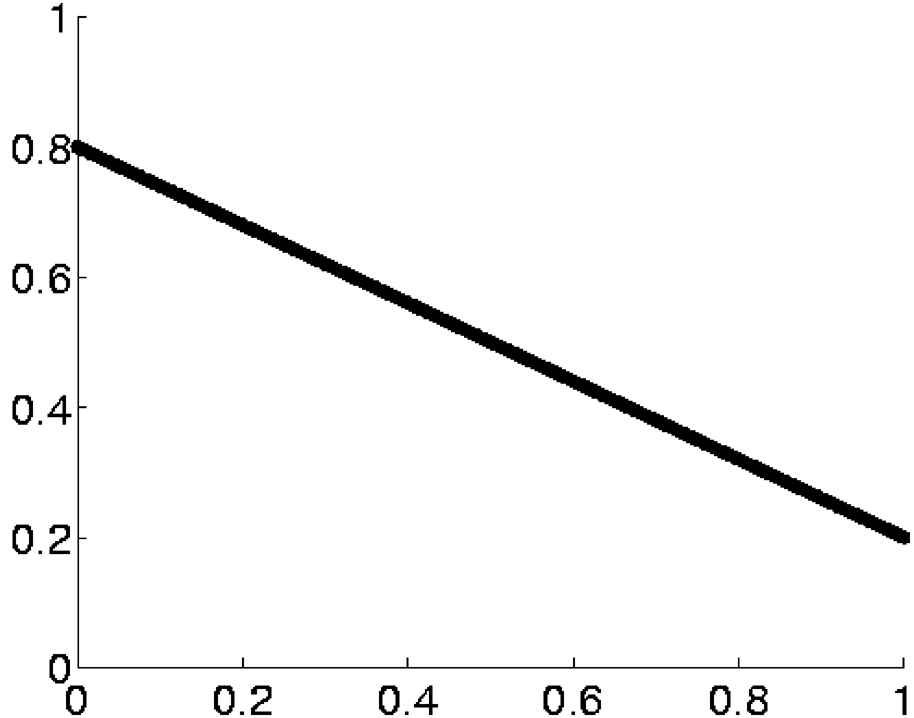
\bar{V}$_1$(b)



$p(z_2)\ V_2(b|z_2)$

State Transitions (Prediction)

- When the agent selects u_3 its state potentially changes.
- When comp… …have to take these… …account.



$$p_1' = \quad = \quad = \quad = 0.8 - 0.6 p_1$$

Resulting Value Function after executing u_3

- Taking the state transitions into account, we finally obtain.

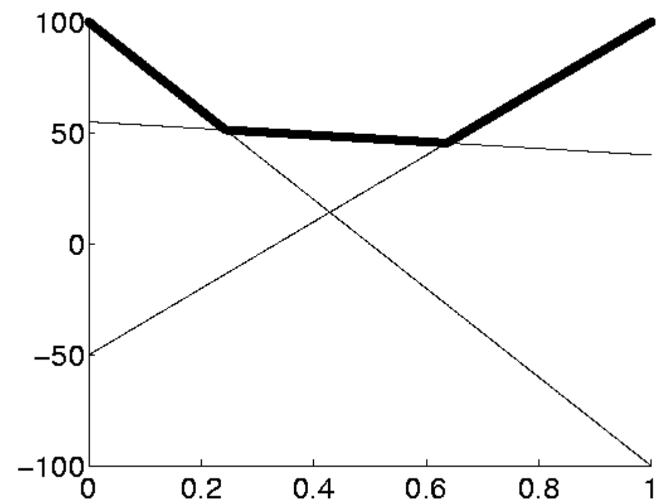
$$\bar{V}_1(b) = \max \left\{ \begin{array}{llll} -70\,p_1 & +30\,(1-p_1) & -30\,p_1 & +70\,(1-p_1) \\ -70\,p_1 & +30\,(1-p_1) & +30\,p_1 & -35\,(1-p_1) \\ +70\,p_1 & -15\,(1-p_1) & -30\,p_1 & +70\,(1-p_1) \\ +70\,p_1 & -15\,(1-p_1) & +30\,p_1 & -35\,(1-p_1) \end{array} \right\}$$

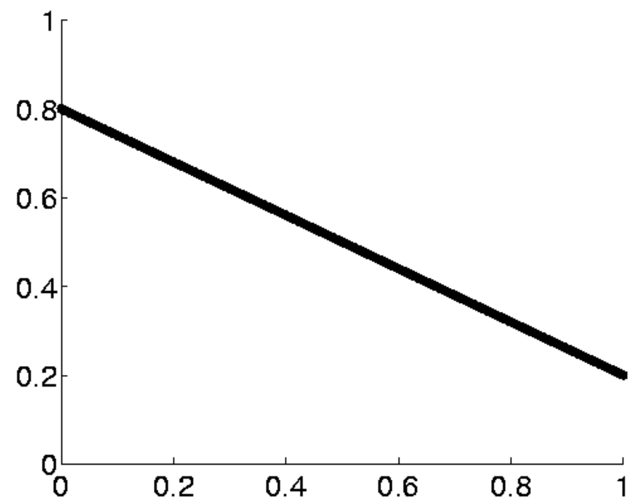
$$= \max \left\{ \begin{array}{ll} -100\,p_1 & +100\,(1-p_1) \\ +40\,p_1 & +55\,(1-p_1) \\ +100\,p_1 & -50\,(1-p_1) \end{array} \right\}$$

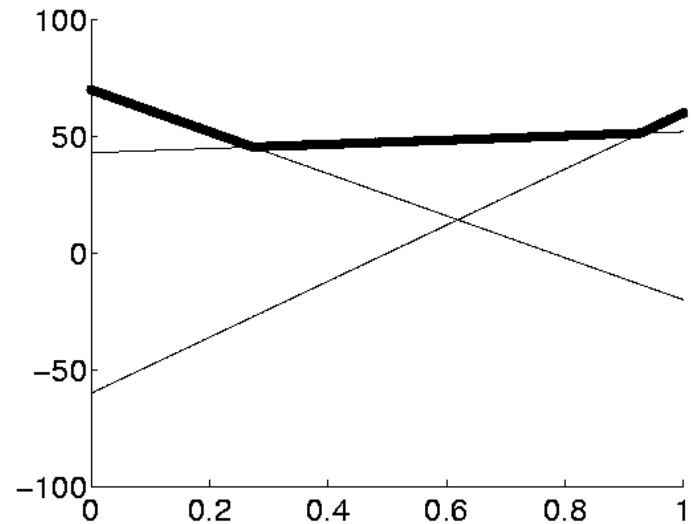
$$\bar{V}_1(b \mid u_3) = \max \left\{ \begin{array}{ll} 60\,p_1 & -60\,(1-p_1) \\ 52\,p_1 & +43\,(1-p_1) \\ -20\,p_1 & +70\,(1-p_1) \end{array} \right\}$$

Value Function after executing *u_3*

\bar{V}$_1$(b)







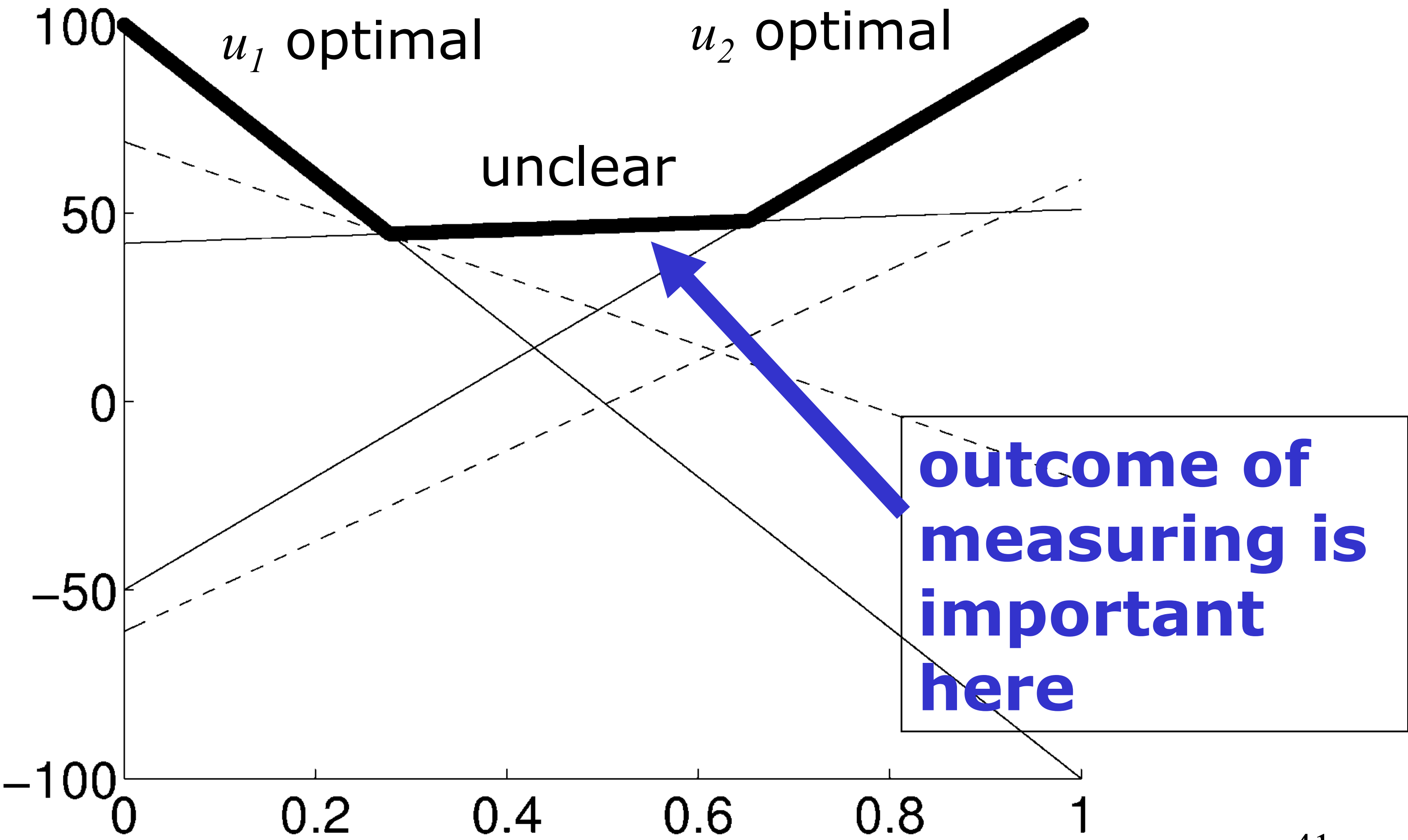
\bar{V}$_1$(b|u$_3$)

Value Function for T=2

- Taking into account that the agent can either directly perform u_1 or u_2 or first u_3 and then u_1 or u_2, we obtain (after pruning)

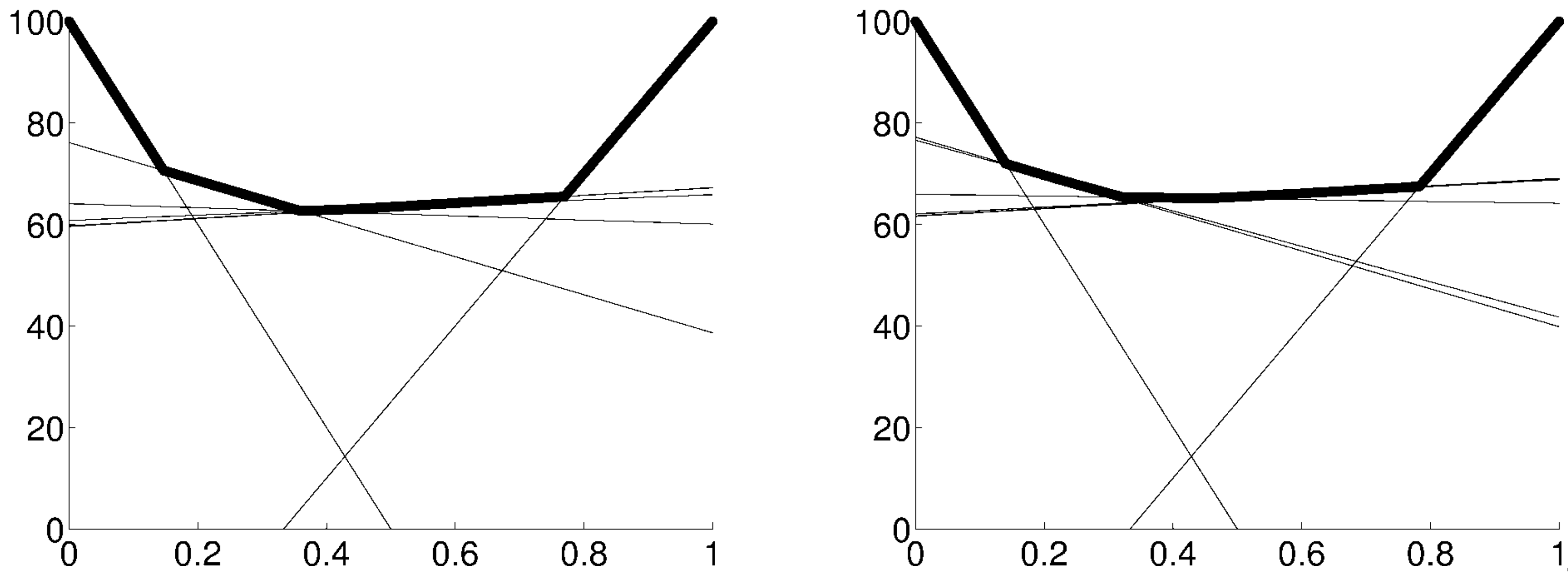
$$\bar{V}_2(b) = \max \left\{ \begin{array}{rr} -100\, p_1 & +100\, (1 - p_1) \\ 100\, p_1 & -50\, (1 - p_1) \\ 51\, p_1 & +42\, (1 - p_1) \end{array} \right\}$$

Graphical Representation of $V_2(b)$

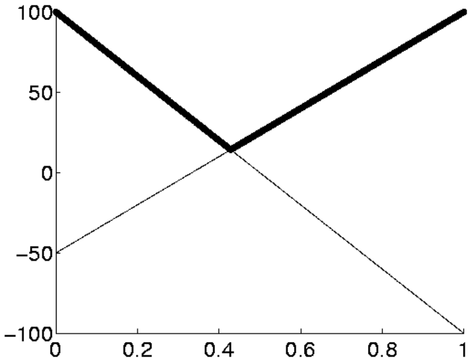


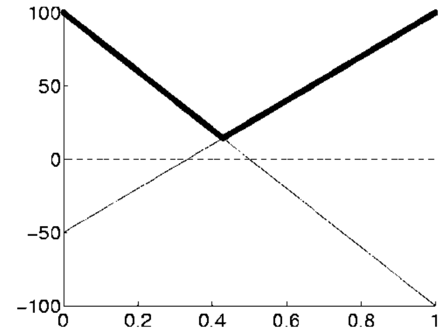
Deep Horizons and Pruning

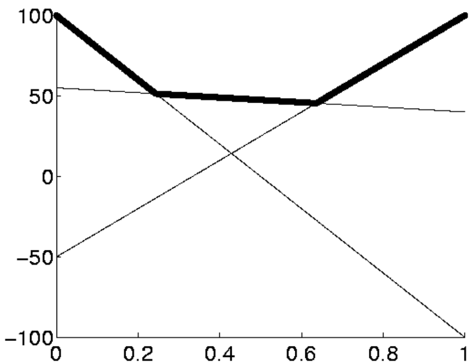
- We have now completed a full backup in belief space.
- This process can be applied recursively.
- The value functions for T=10 and T=20 are

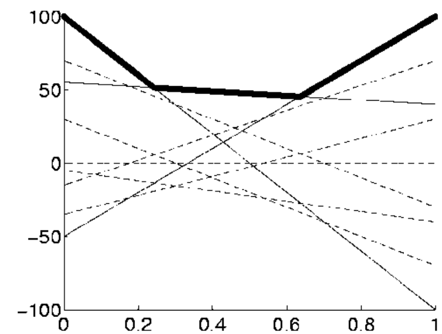


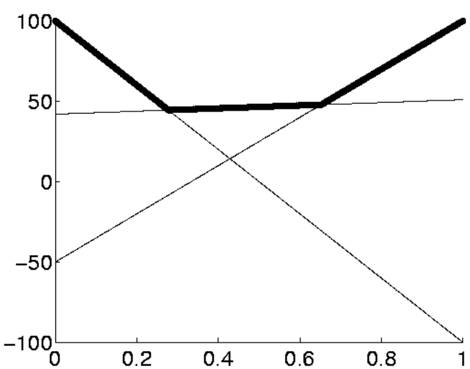
Deep Horizons and Pruning

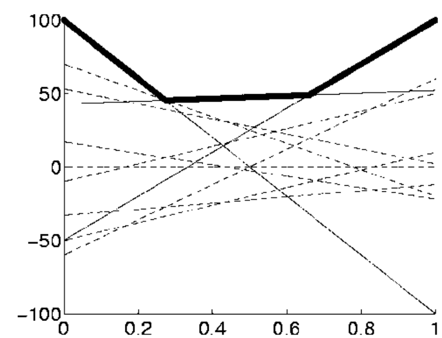












Algorithm POMDP(T):

$\Upsilon = (0, \ldots, 0)$
for $\tau = 1$ *to* T *do*
  $\Upsilon' = \emptyset$
  for all $(u'; v_1^k, \ldots, v_N^k)$ *in* Υ *do*
    for all control actions u *do*
      for all measurements z *do*
        for $j = 1$ *to* N *do*

$$v_{j,u,z}^k = \sum_{i=1}^{N} v_i^k \; p(z \mid x_i) \; p(x_i \mid u, x_j)$$

        endfor
      endfor
    endfor
  endfor
  for all control actions u *do*
    for all $k(1), \ldots, k(M) = (1, \ldots, 1)$ *to* $(|\Upsilon|, \ldots, |\Upsilon|)$ *do*
      for $i = 1$ *to* N *do*

$$v_i' = \gamma \left[r(x_i, u) + \sum_z v_{u,z,i}^{k(z)} \right]$$

      endfor
      add $(u; v_1', \ldots, v_N')$ *to* Υ'
    endfor
  endfor
  optional: prune Υ'
  $\Upsilon = \Upsilon'$
endfor
return Υ

Why Pruning is Essential

- Each update introduces additional linear components to V.
- Each measurement squares the number of linear components.
- Thus, an unpruned value function for T=20 includes more than $10^{547,864}$ linear functions.
- At T=30 we have $10^{561,012,337}$ linear functions.
- The pruned value functions at T=20, in comparison, contains only 12 linear components.
- The combinatorial explosion of linear components in the value function are the major reason why POMDPs are impractical for most applications.

POMDP Summary

- POMDPs compute the optimal action in partially observable, stochastic domains.
- For finite horizon problems, the resulting value functions are piecewise linear and convex.
- In each iteration the number of linear constraints grows exponentially.
- POMDPs so far have only been applied successfully to very small state spaces with small numbers of possible observations and actions.