CSE-571
Probabilistic Robotics

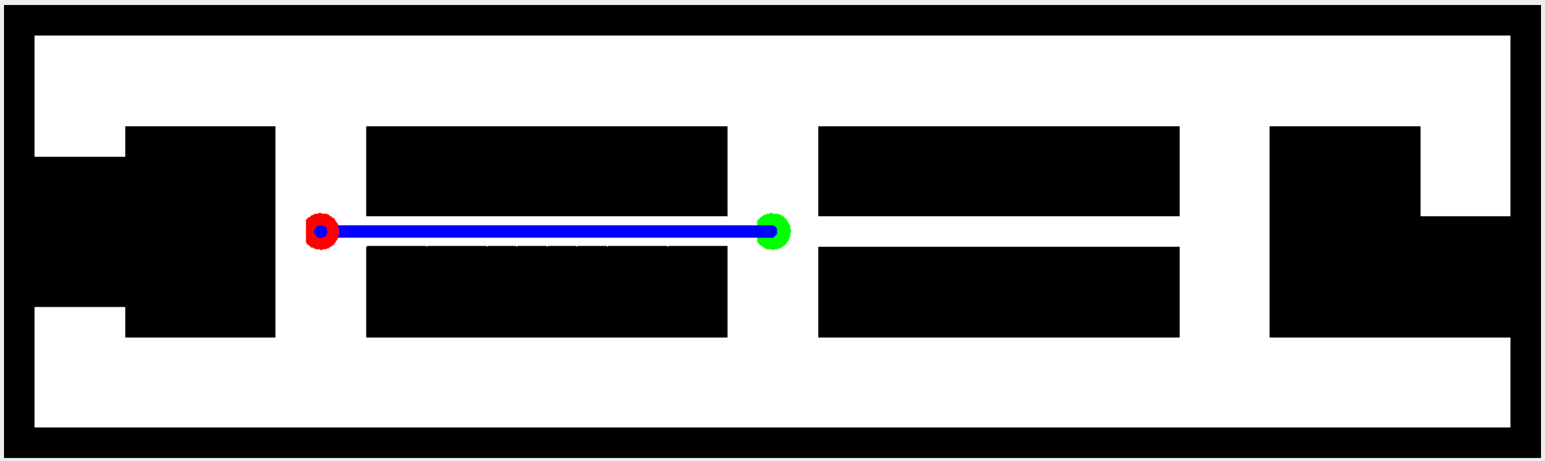
Planning and Control:

Markov Decision Processes

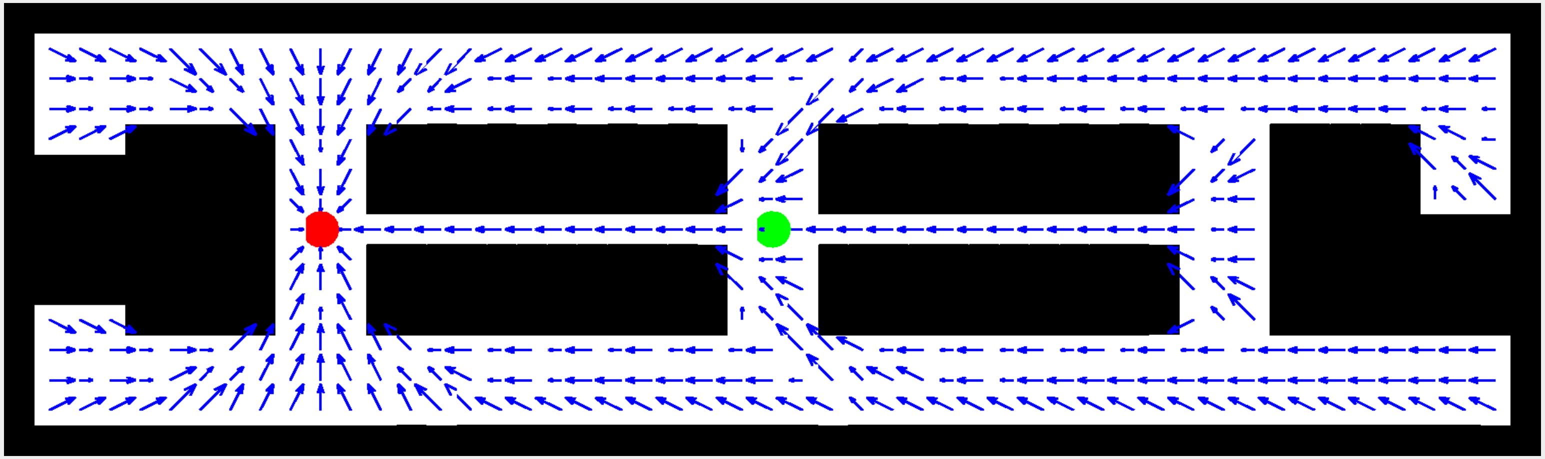
Problem Classes

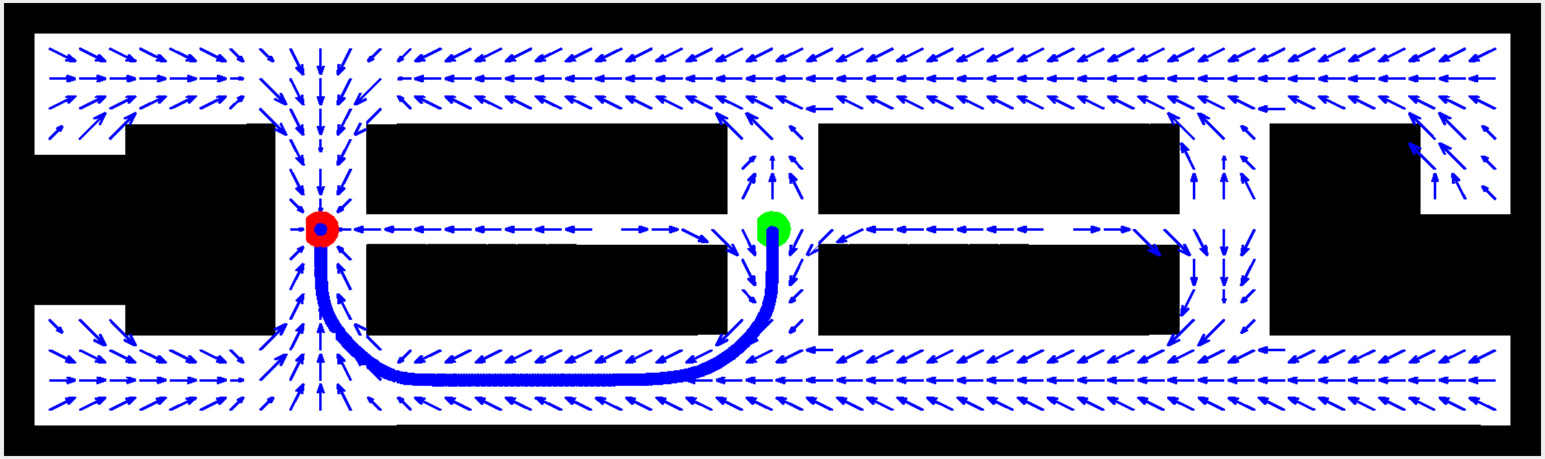
- Deterministic vs. stochastic actions
- Full vs. partial observability

Deterministic, fully observable

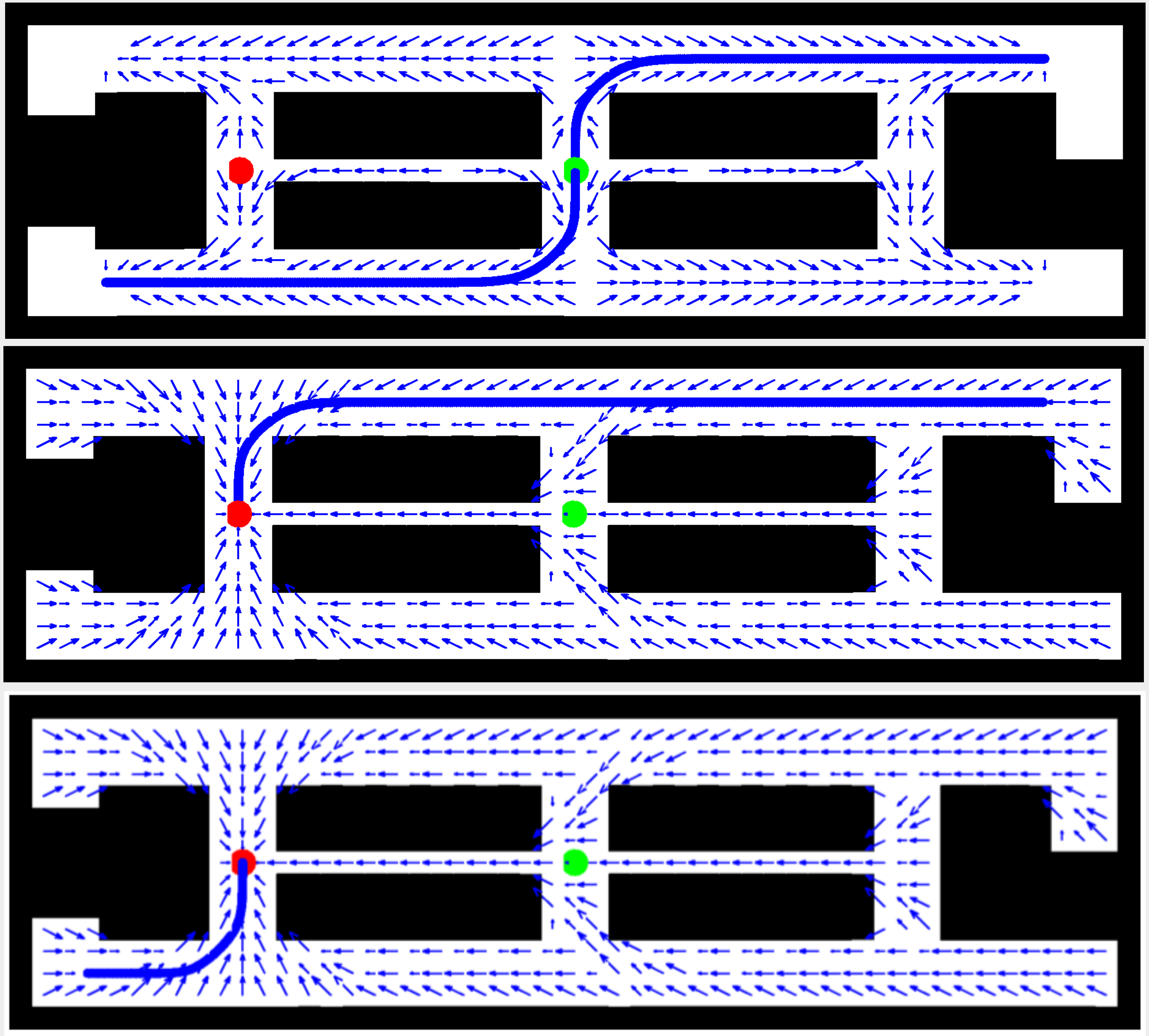


Stochastic, Fully Observable

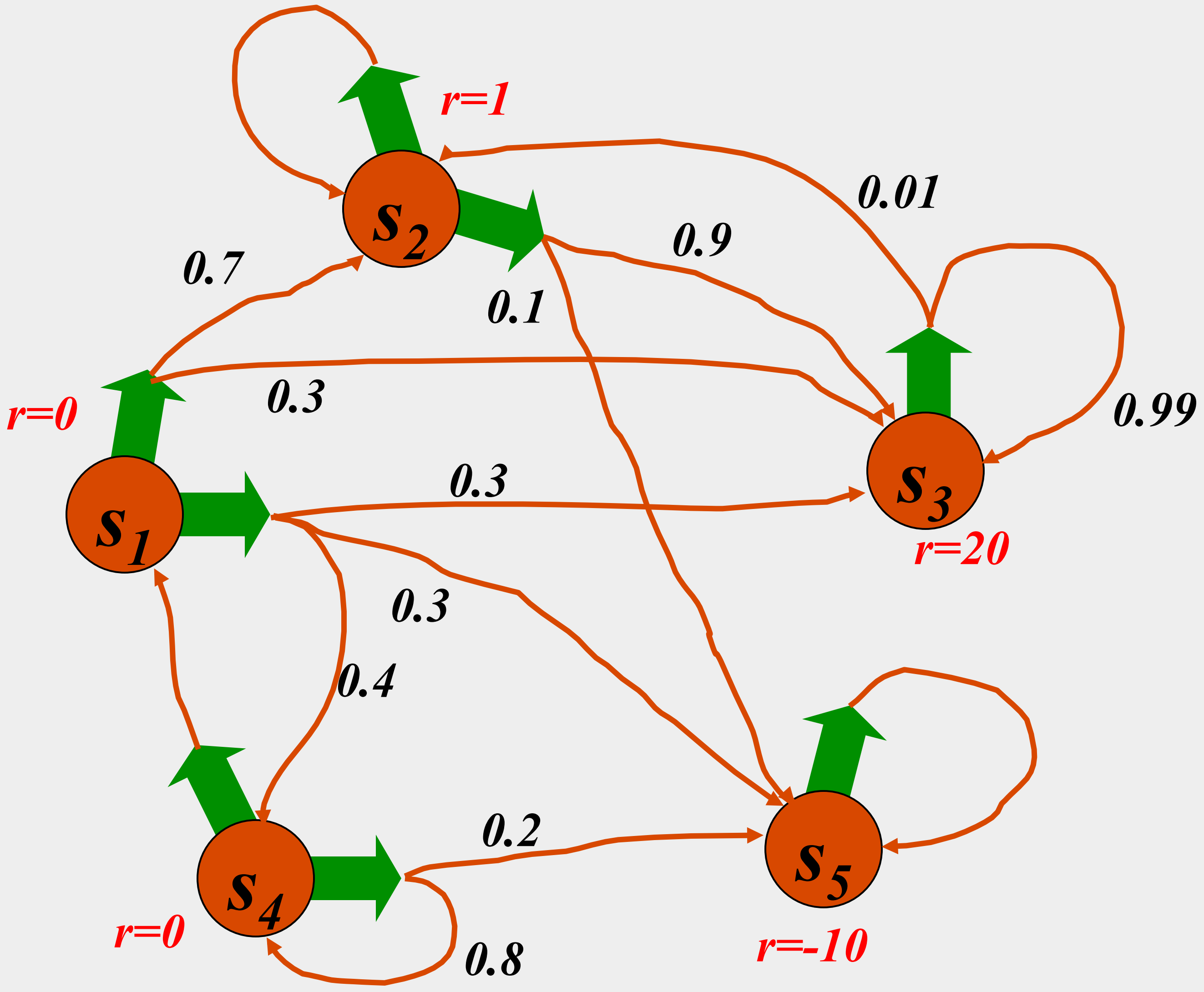




Stochastic, Partially Observable



Markov Decision Process (MDP)



Markov Decision Process (MDP)

- **Given:**
- States x
- Actions u
- Transition probabilities $p(x'|u,x)$
- Reward / payoff function $r(x,u)$

- **Wanted:**
- Policy $p(x)$ that maximizes the future expected reward

Rewards and Policies

- Policy (general case):

$$\pi : \quad z_{1:t-1}, u_{1:t-1} \rightarrow \; u_t$$

- Policy (fully observable case):

$$\pi : \quad x_t \rightarrow \; u_t$$

- Expected cumulative payoff:

$$R_T = E \left[\sum_{\tau=1}^{T} \gamma^{\tau} r_{t+\tau} \right]$$

  - T=1: greedy policy
  - T>1: finite horizon case, typically no discount
  - T=infty: infinite-horizon case, finite reward if discount < 1

Policies contd.

- Expected cumulative payoff of policy:

$$R_T^\pi(x_t) = E\left[\sum_{\tau=1}^{T} \gamma^\tau r_{t+\tau} \mid u_{t+\tau} = \pi\left(z_{1:t+\tau-1} u_{1:t+\tau-1}\right)\right]$$

- Optimal policy:

$$\pi^* = \underset{\pi}{\operatorname{argmax}} \quad R_T^\pi(x_t)$$

- 1-step optimal policy:

$$\pi_1(x) = \underset{u}{\operatorname{argmax}} \quad r(x,u)$$

- Value function of 1-step optimal policy:

$$V_1(x) = \gamma \max_u r(x,u)$$

2-step Policies

- Optimal policy:

$$\pi_2(x) = \underset{u}{\operatorname{argmax}} \left[r(x,u) + \int V_1(x')\, p(x' \mid u,x)\, dx' \right]$$

- Value function:

$$V_2(x) = \gamma \max_u \left[r(x,u) + \int V_1(x')\, p(x' \mid u,x)\, dx' \right]$$

T-step Policies

- Optimal policy:

$$\pi_T(x) = \underset{u}{\operatorname{argmax}} \left[r(x,u) + \int V_{T-1}(x') p(x' \mid u,x) dx' \right]$$

- Value function:

$$V_T(x) = \gamma \max_{u} \left[r(x,u) + \int V_{T-1}(x') p(x' \mid u,x) dx' \right]$$

Infinite Horizon

- Optimal policy:

$$V_\infty(x) = \gamma \max_u \left[r(x,u) + \int V_\infty(x') p(x' \mid u,x) dx' \right]$$

- Bellman equation
- Fix point is optimal policy
- Necessary and sufficient condition

Value Iteration

- for all *x* do

$$\hat{V}(x) \leftarrow r_{\min}$$

- endfor

- repeat until convergence
  - for all *x* do

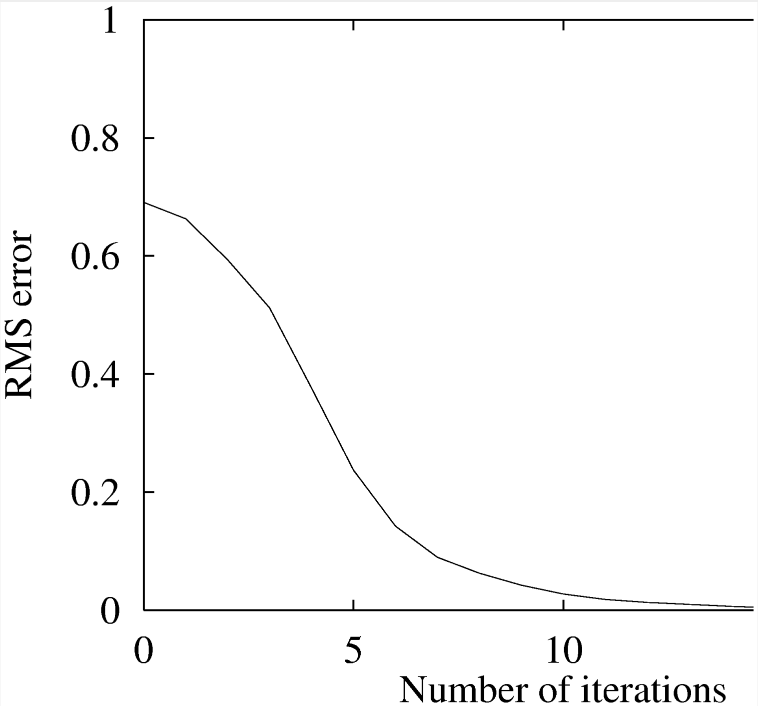
$$\hat{V}(x) \leftarrow \gamma \max_u \left[r(x,u) + \int \hat{V}(x')\, p(x' \mid u,x)\, dx' \right]$$

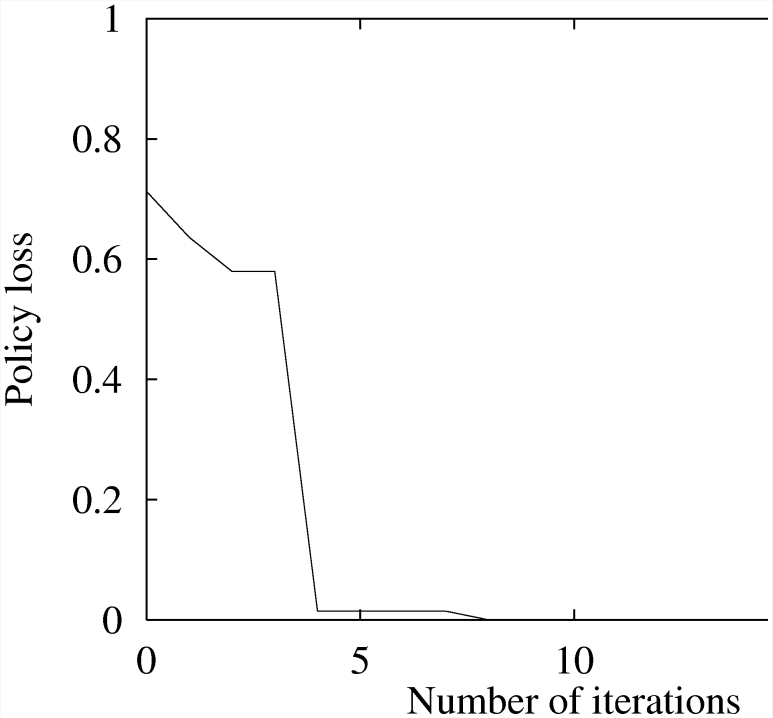
  - endfor
- endrepeat

$$\pi(x) = \operatorname*{argmax}_u \left[r(x,u) + \int \hat{V}(x')\, p(x' \mid u,x)\, dx' \right]$$

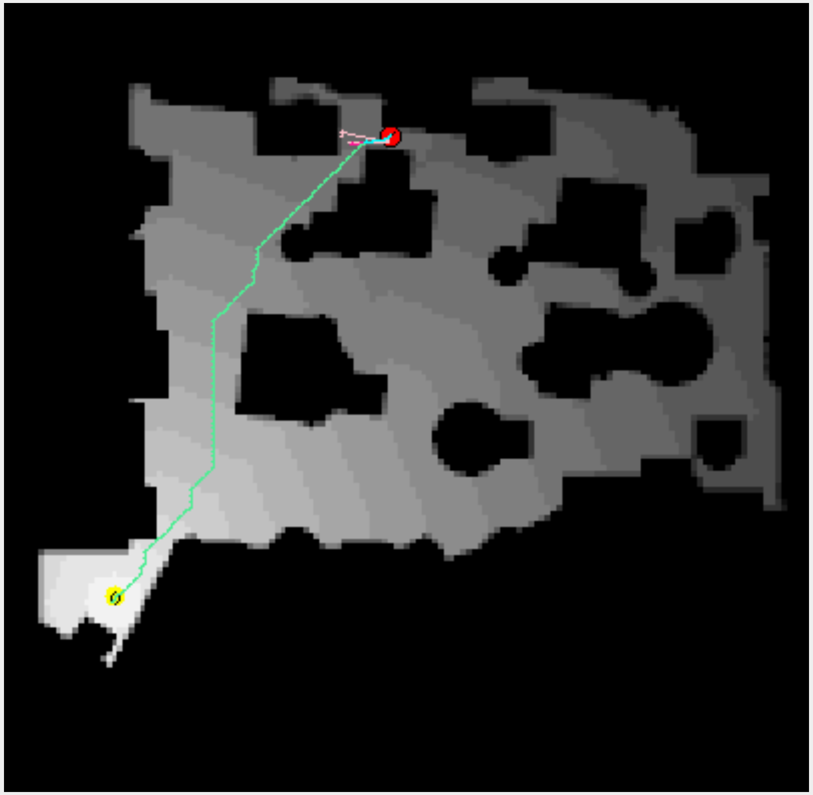
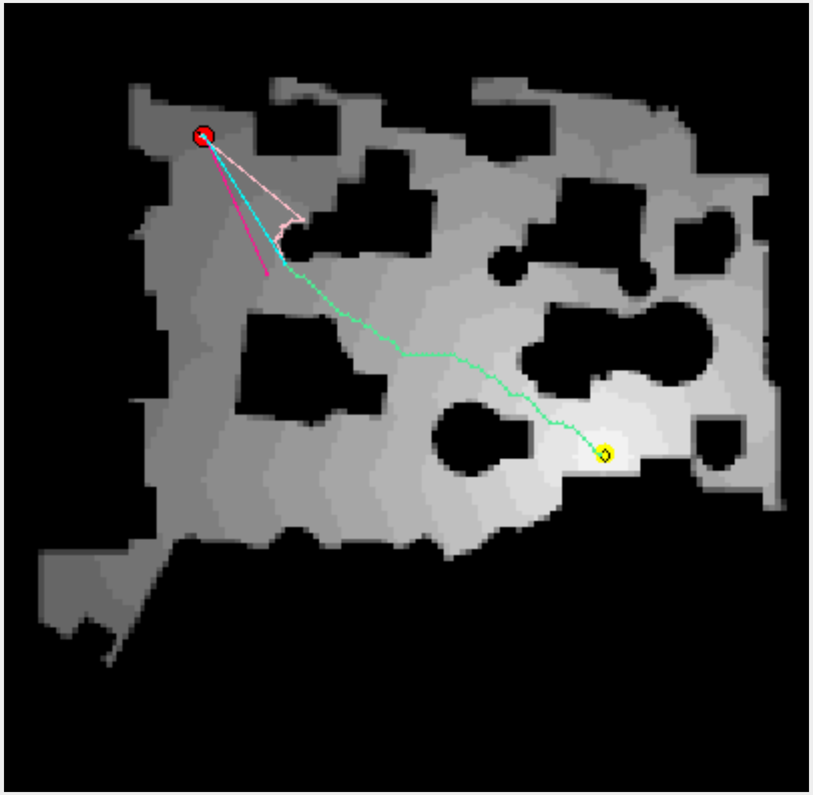
Value Function and Policy

- Each step takes O(|A| |S|) time.
- Number of iterations required is polynomial in |S|, |A|, 1/(1-gamma)



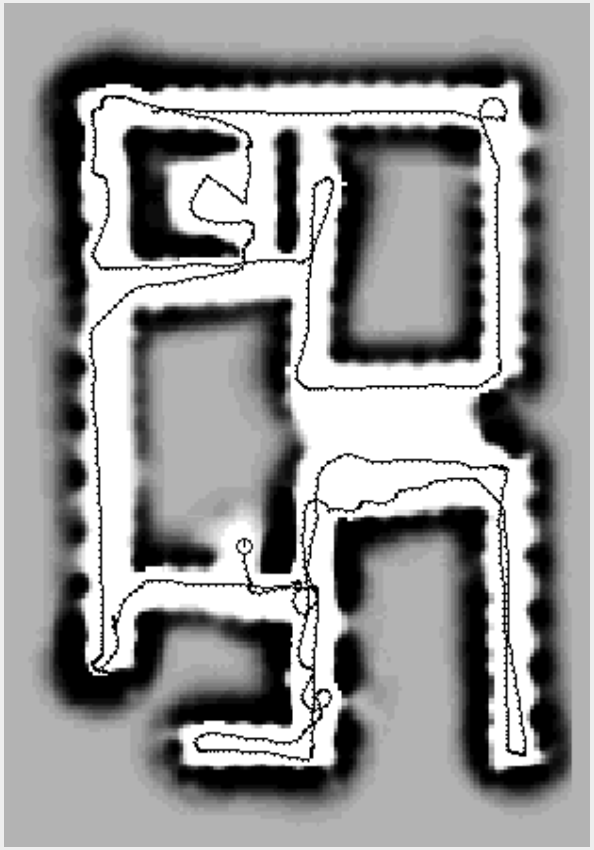


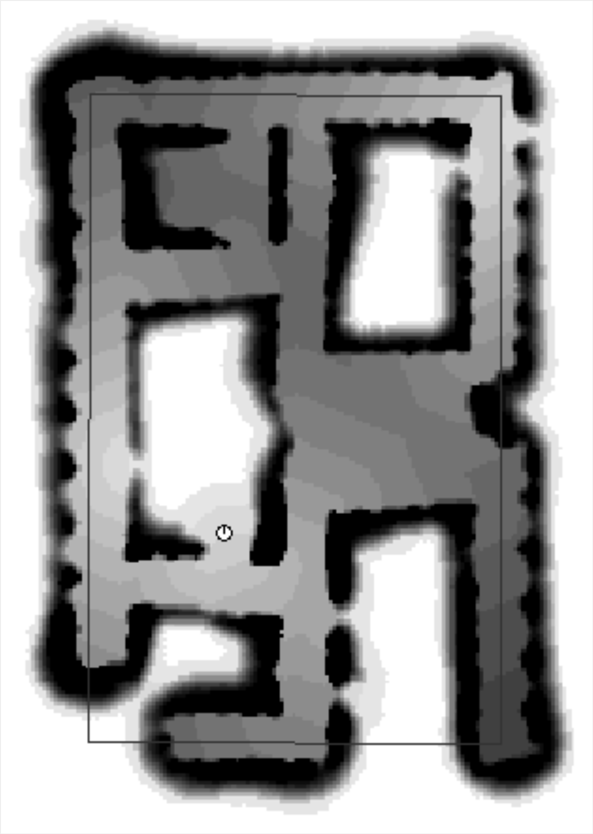
Value Iteration for Motion Planning
(assumes knowledge of robot's location)

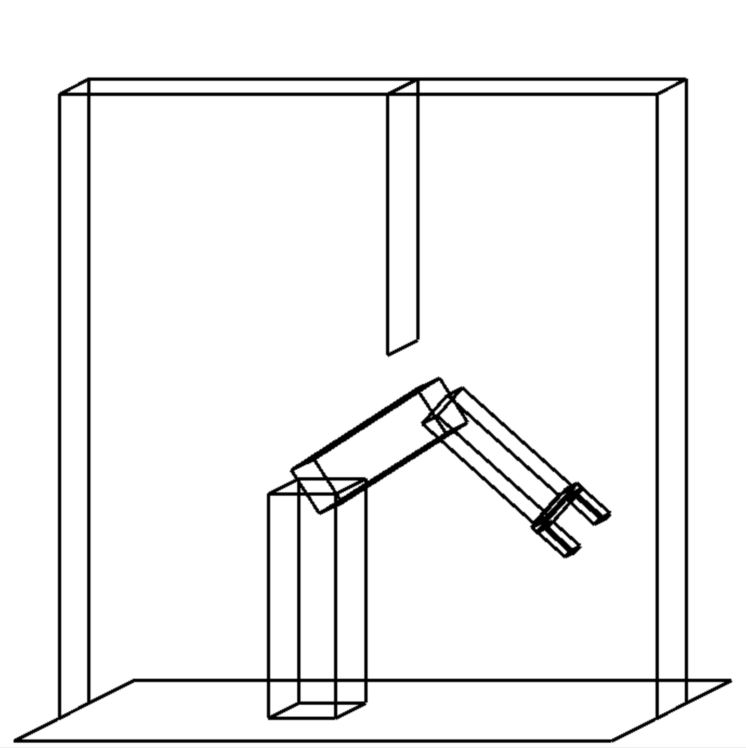
Frontier-based Exploration

- Every unknown location is a target point.

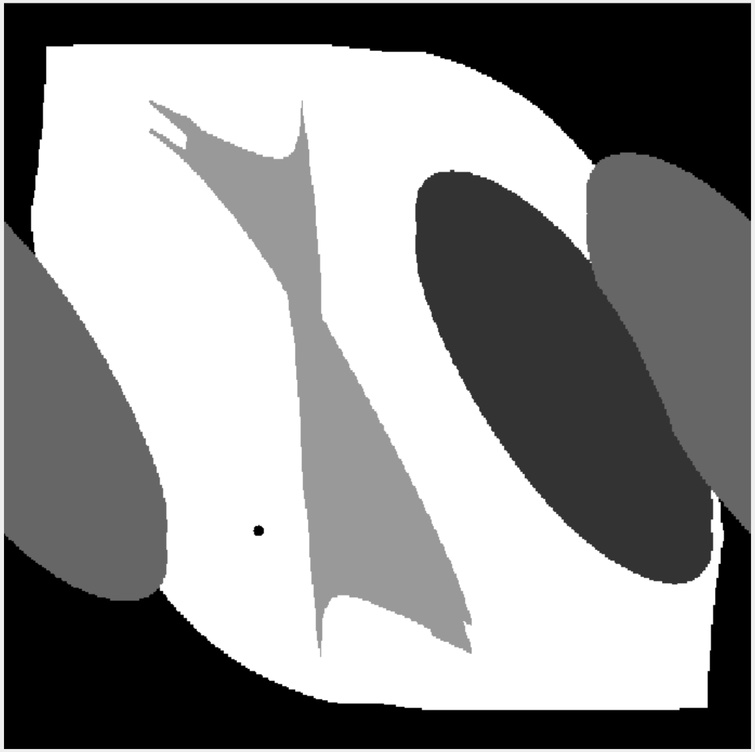




Manipulator Control

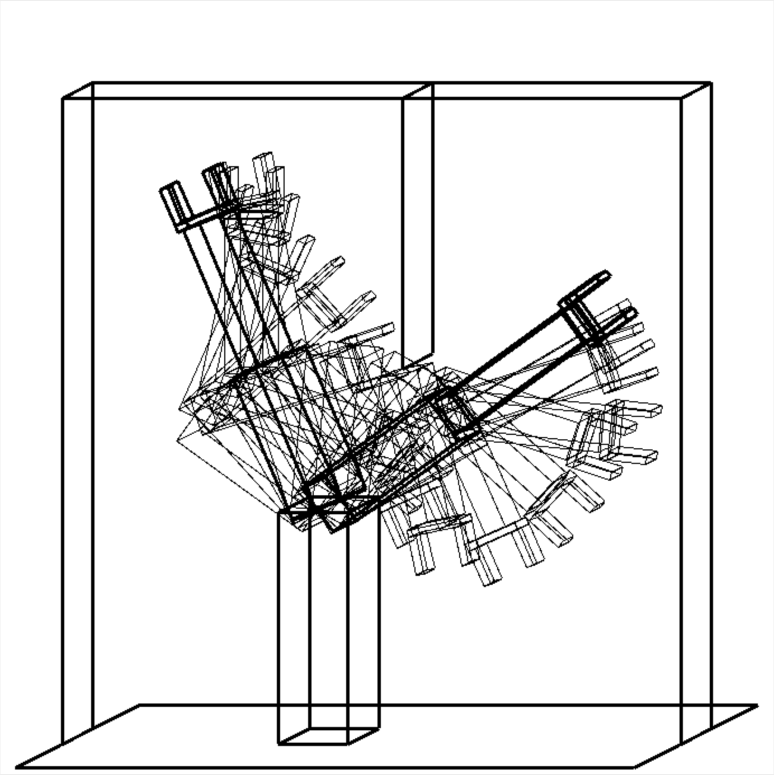


Arm with two joints

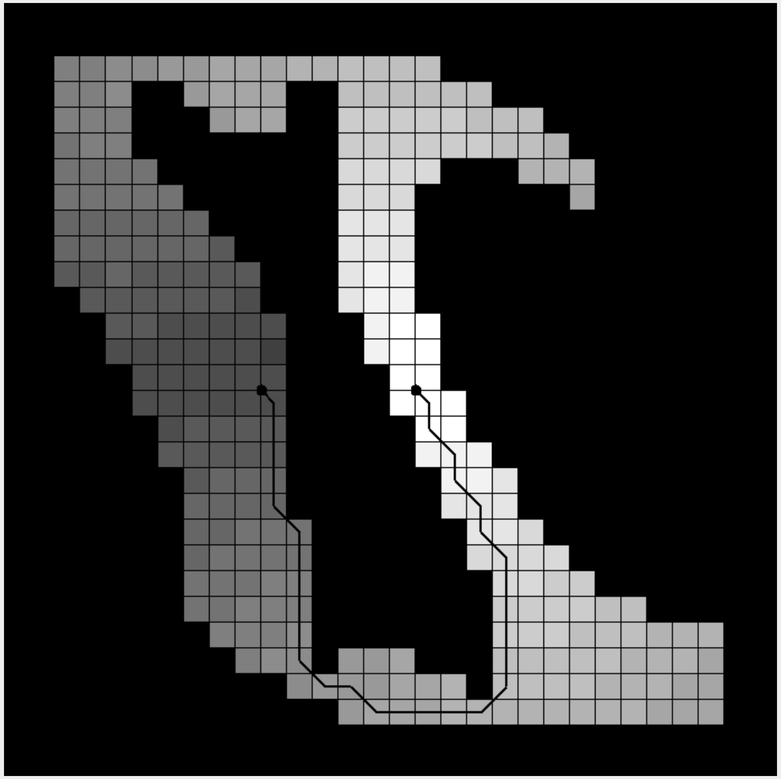


Configuration space

Manipulator Control Path

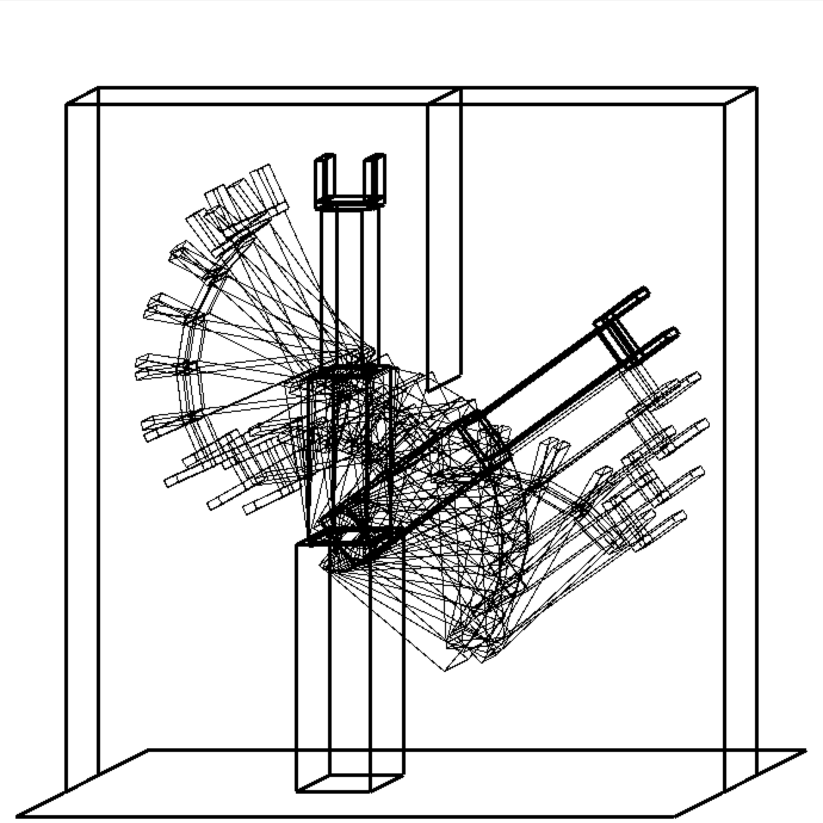


State space

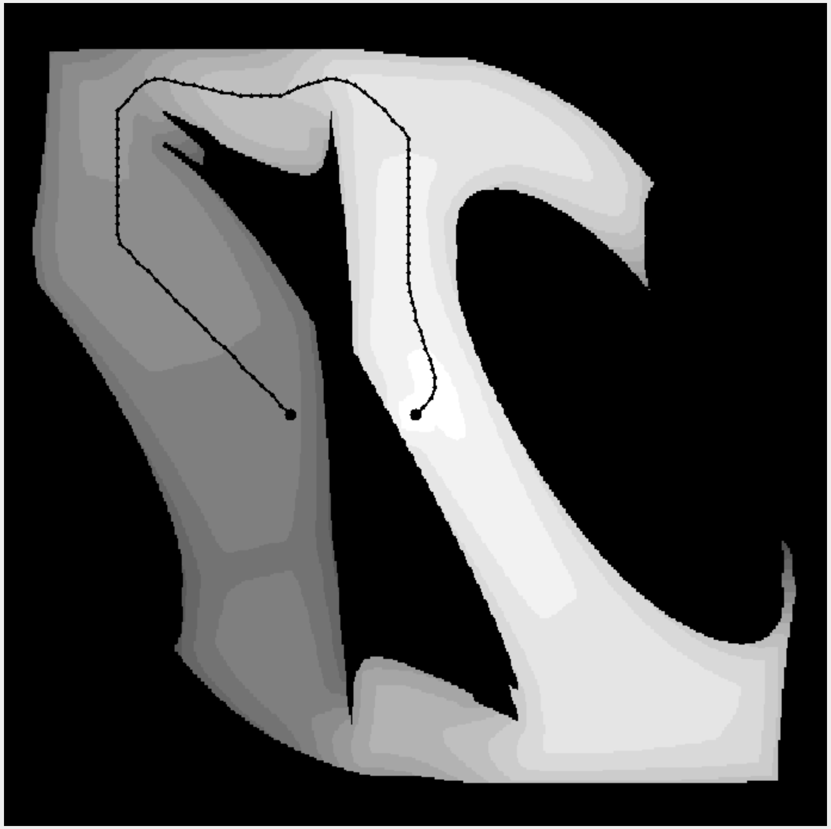


Configuration space

Manipulator Control Path



State space



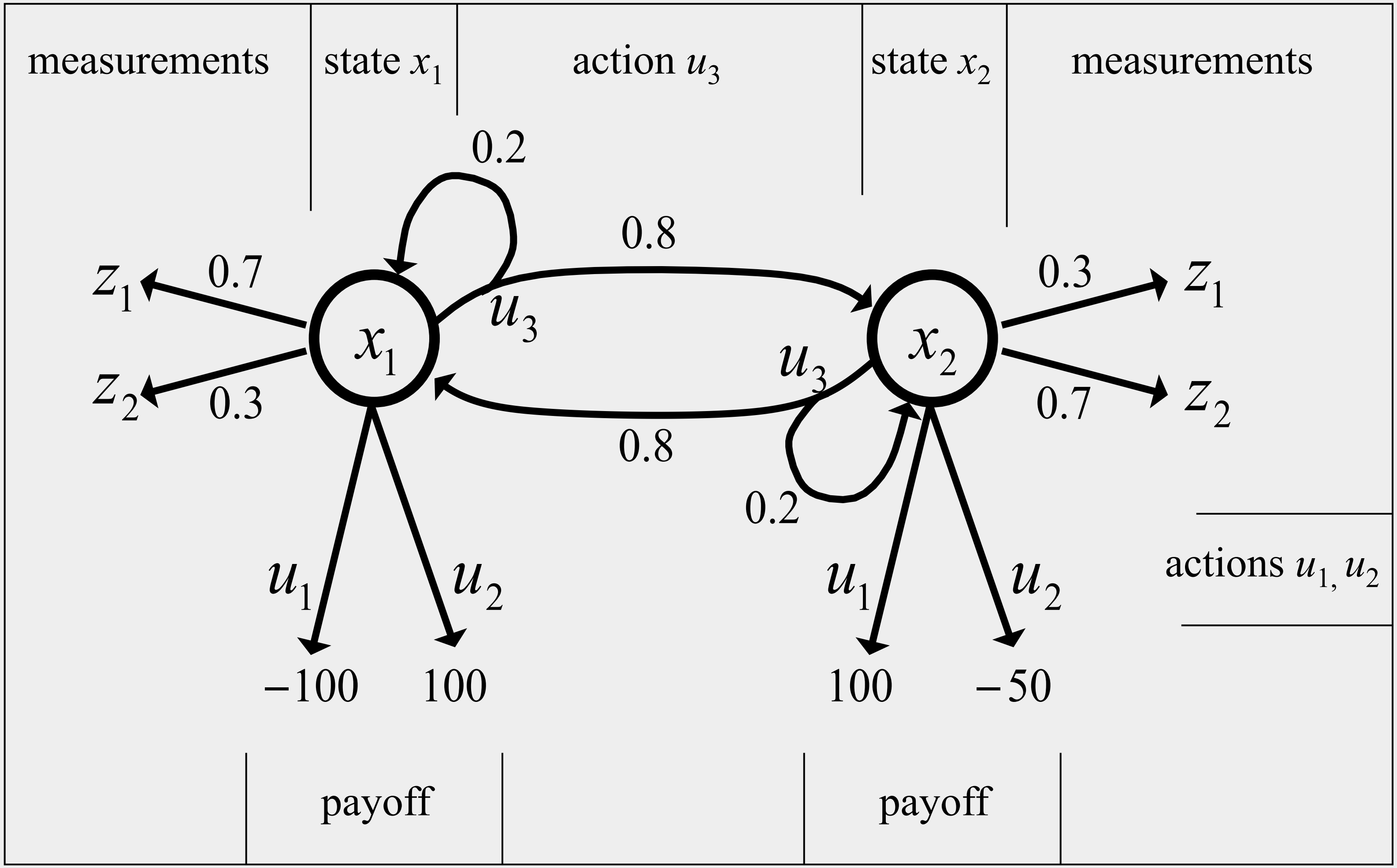
Configuration space

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 state under consideration.
- POMDPs compute a **value function over belief space**:

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

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 no discount.

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

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

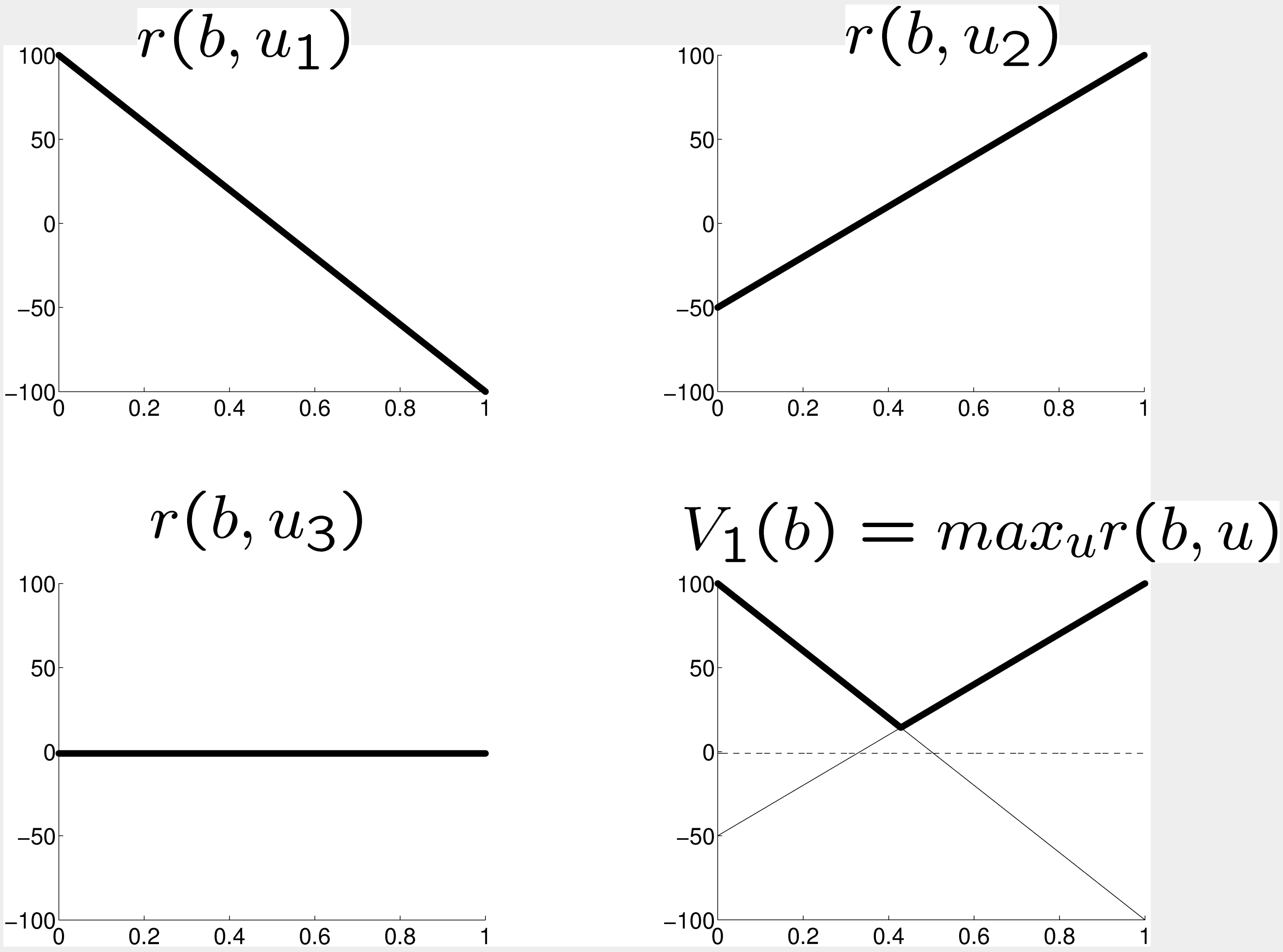
$$
\begin{aligned}
p(z_1 | x_1) &= 0.7 & \quad p(z_2 | x_1) &= 0.3 \\
p(z_1 | x_2) &= 0.3 & \quad p(z_2 | x_2) &= 0.7
\end{aligned}
$$

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**:

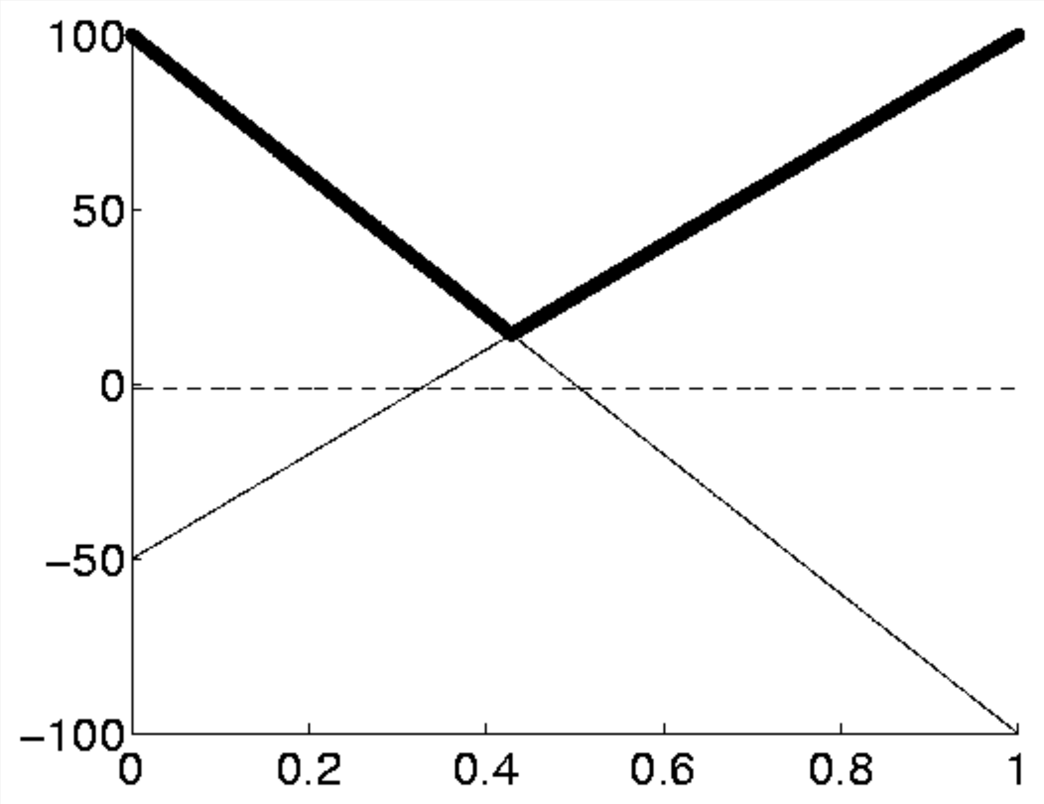
$$
\begin{aligned}
r(b, u_1) &= -100\, p_1 + 100\, p_2 \\
&= -100\, p_1 + 100\, (1 - p_1) \\
r(b, u_2) &= 100\, p_1 - 50\, (1 - p_1) \\
r(b, u_3) &= -1
\end{aligned}
$$

Payoffs in Our Example (2)



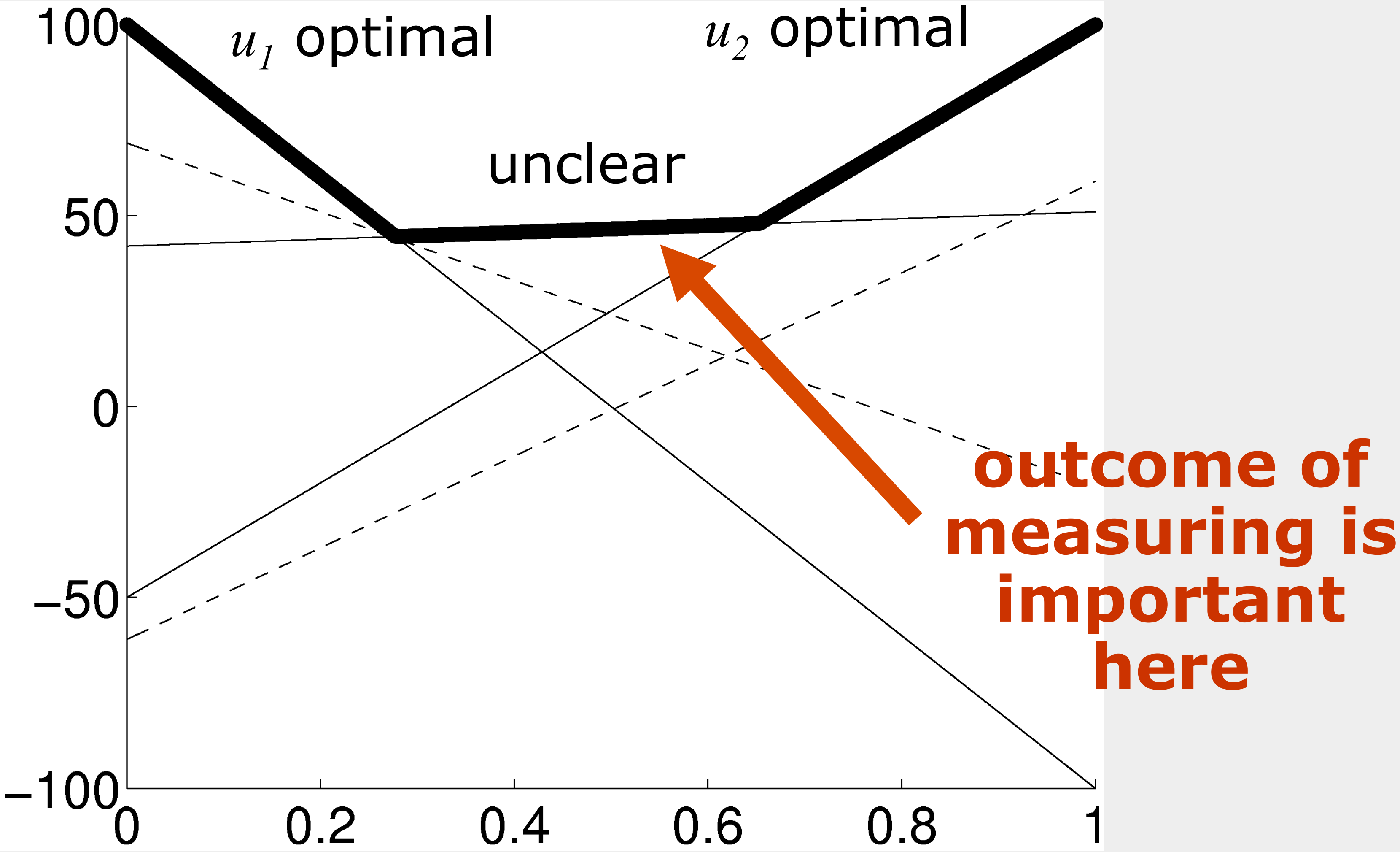
Increasing the Time Horizon

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



$V_1(b)$

Graphical Representation of $V_2(b)$



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.