Dialogue as a
Decision Making Process

Nicholas Roy

AERO | ASTRO

Challenges of Autonomy in the Real World

Wide range of sensors
Noisy sensors
World dynamics
Adaptability
Incomplete information

Robustness under uncertainty

Minerva

Pearl

Predicted Health Care Needs

- By 2008, need 450,000 additional nurses:
  - Monitoring and walking assistance
    30 % of adults 65 years and older have fallen this year

Cost of preventable falls: *Alexander 2001*
$32 Billion US/year

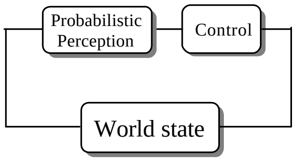
  - Intelligent reminding

Cost of medication non-compliance:
$1 Billion US/year *Dunbar-Jacobs 2000*

Spoken Dialogue Management

- We want...
  - Natural dialogue...
  - With untrained (and untrainable) users...
  - In an uncontrolled environment...
  - Across many unrelated domains
- Cost of errors...
  - Medication is not taken, or taken incorrectly
  - Robot behaves inappropriately
  - User becomes frustrated, robot is ignored, and becomes useless
- How to generate such a policy?

Perception and Control



Probabilistic Methods for Dialogue Management

- Markov Decision Processes model action uncertainty
  - (Levin et. al, 1998, Goddeau & Pineau, 2000)
- Many techniques for learning optimal policies, especially reinforcement learning
  - (Singh et al. 1999, Litman et al. 2000, Walker 2000)

Markov Decision Processes

- A Markov Decision Process is given formally by the following:
  - a set of states S={s_1, s_2, ..., s_n}
  - a set of actions A={a_1, a_2, ..., a_m}
  - a set of transition probabilities T(s_i, a, s_j) = p(s_j |a, s_i)
  - a set of rewards R: S ×A? ℜ
  - a discount factor γ⊂[0, 1]
  - an initial state s_0∈ S
- Bellman's equation (Bellman, 1957) computes the expected reward for each state recursively,

$$J(\mathbf{s}_i) = \max_a \left(R(\mathbf{s}_i, a) + \gamma \sum_{j=1}^{N} p(\mathbf{s}_j | \mathbf{s}_i, a) \cdot J(\mathbf{s}_j) \right)$$

- and determines the policy that maximises the expected, discounted reward

The POMDP in Dialogue Management

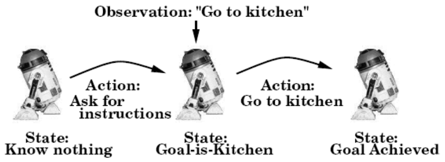
- State: Represents desire of user
  e.g. `want_tv, want_meds`
- This state is unobservable to the dialogue system
- Observations: Utterances from speech recogniser
  e.g. .I want to take my pills now.
- The system must infer the user's state from the possibly noisy or ambiguous observations
- Where do the emission probabilities come from?
  - At planning time, from a prior model
  - At run time, from the speech recognition engine

The MDP in Dialogue Management

- State: Represents desire of user
  e.g. `want_tv, want_meds`
- Assume utterances from speech recogniser give state
  e.g. I want to take my pills now.
- Actions are: robot motion, speech acts
- Reward: maximised for satisfying user task

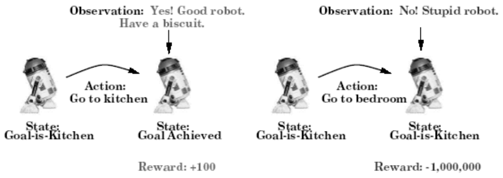
Markov Decision Processes

- Model the world as different states the system can be in
  e.g. current state of completion of a form
- Each action moves to some new state with probability p(i; j)
- Observation from user determines posterior state



Markov Decision Processes

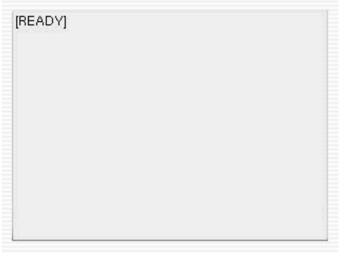
- Optimal policy maximizes expected future (discounted) reward
- Policy found using value iteration



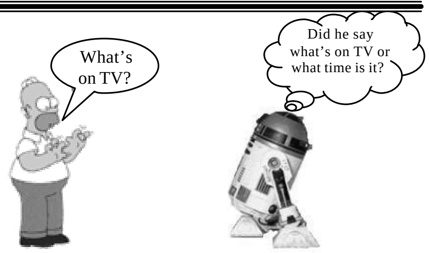
Markov Decision Processes

- Since we can compute a policy that maximises the expected reward...
- then if we have ...
  - a reasonable reward function
  - a reasonable transition model
- Do we get behaviour that satisfies the user?

Fully Observable State Representation

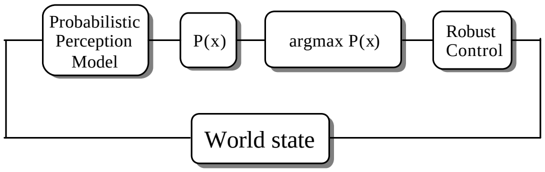

[READY]

Fully Observable State Representation



- Advantage: No state identification/tracking problems
- Disadvantage: What if the observation is noisy or false?

Perception and Control


Probabilistic Perception Model

P(x)

argmax P(x)

Robust Control

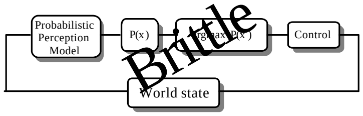
World state

Talk Outline

- Robots in the real world
- **Partially Observable Markov Decision Processes**
- Solving large POMDPs
- Deployed POMDPs

Control Models

- Markov Decision Processes



Probabilistic Perception Model

P(x)

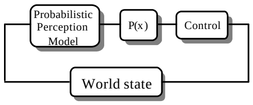
argmax P(x)

Control

World state

Brittle

- Partially Observable Markov Decision Processes



Probabilistic Perception Model

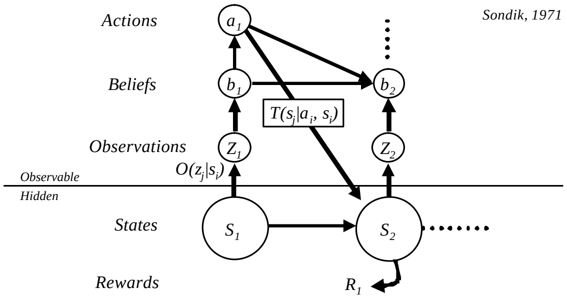
P(x)

Control

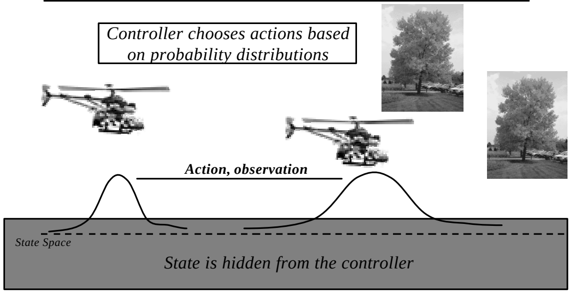
World state

POMDPs

Sondik, 1971



Navigation as a POMDP



Controller chooses actions based on probability distributions

Action, observation

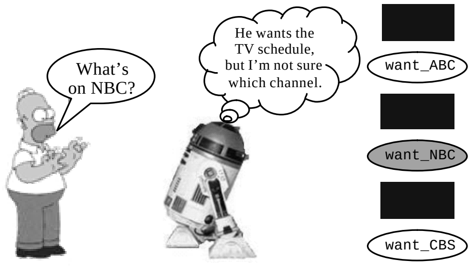
State Space

State is hidden from the controller

The POMDP in Dialogue Management

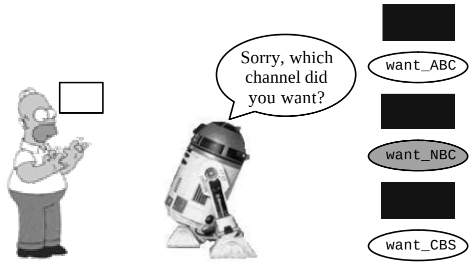
- State: Represents desire of user *e.g.* `want_tv, want_meds`
- This state is unobservable to the dialogue system
- Observations: Utterances from speech recogniser *e.g. .I want to take my pills now.*
- The system must infer the user's state from the possibly noisy or ambiguous observations
- Where do the emission probabilities come from?
  - At planning time, from a prior model
  - At run time, from the speech recognition engine
- Actions are still robot motion, speech acts
- Reward: maximised for satisfying user task

The POMDP in Dialogue Management

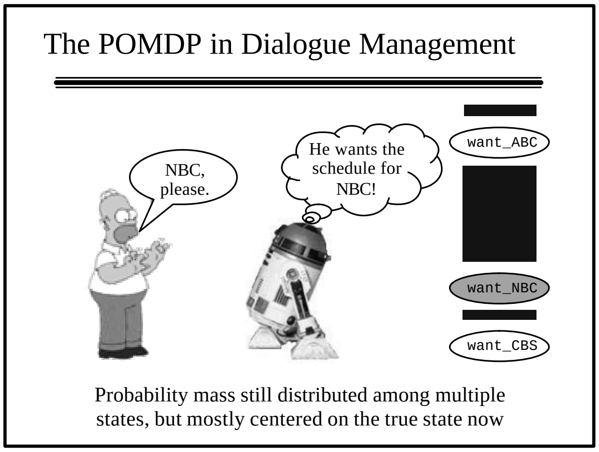


Probability distributed among multiple states

The POMDP in Dialogue Management

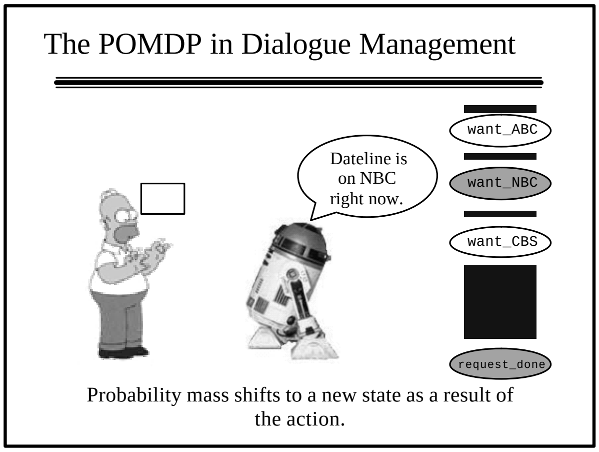


Probability still distributed among multiple states



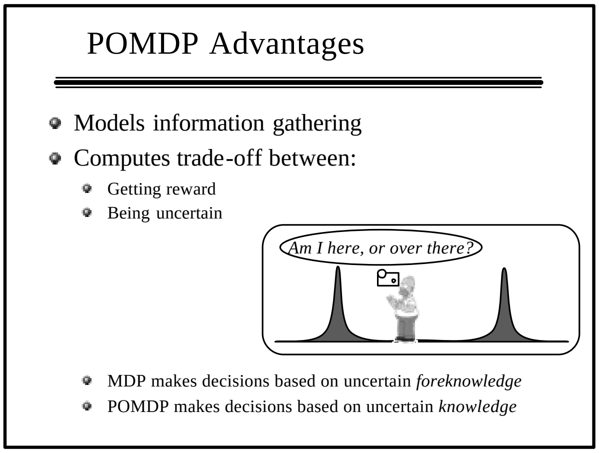
The POMDP in Dialogue Management

Probability mass still distributed among multiple states, but mostly centered on the true state now



The POMDP in Dialogue Management

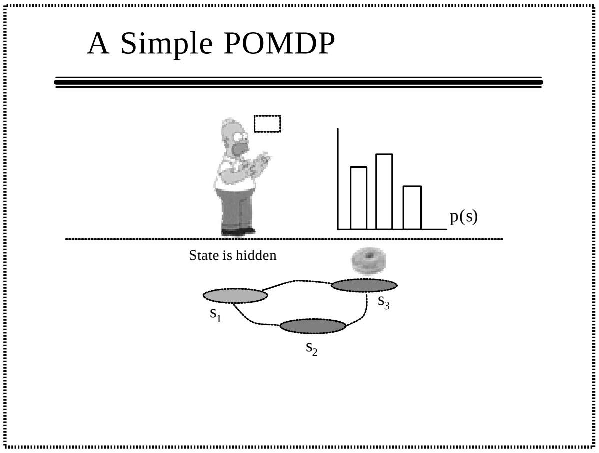
Probability mass shifts to a new state as a result of the action.



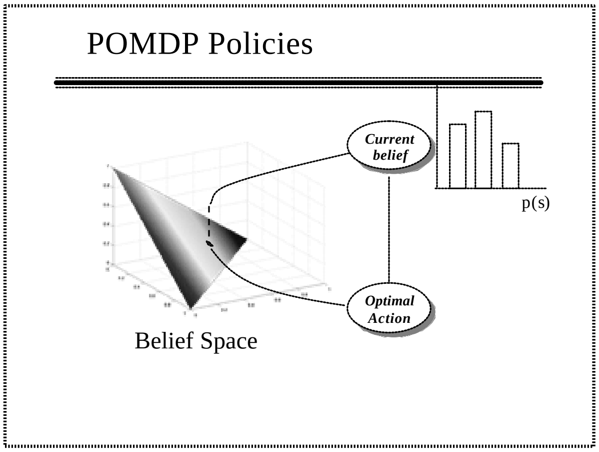
POMDP Advantages

- Models information gathering
- Computes trade-off between:
  - Getting reward
  - Being uncertain

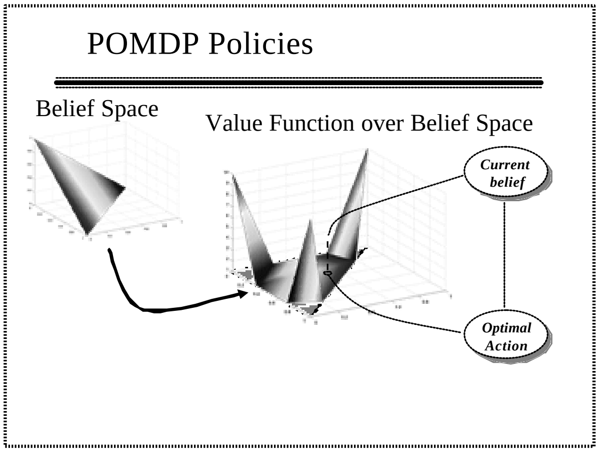
  - MDP makes decisions based on uncertain *foreknowledge*
  - POMDP makes decisions based on uncertain *knowledge*



A Simple POMDP



POMDP Policies

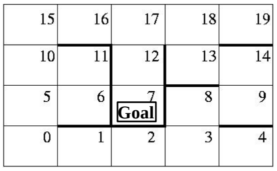


POMDP Policies

Scaling POMDPs

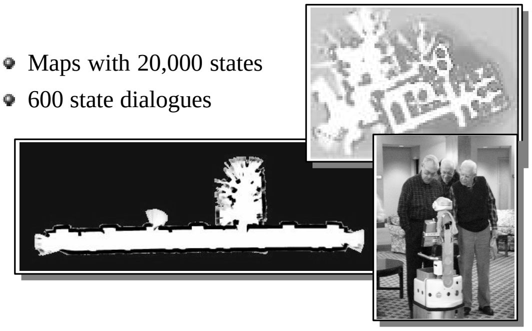
~~This simple 20 state maze problem takes 24 hours for 7 steps of value iteration.~~

1 hour, Zhang & Zhang 2001



The Real World

- Maps with 20,000 states
- 600 state dialogues



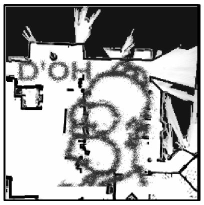
Structure in POMDPs

- Factored models
  - Boutilier & Poole, 1996
  - Guestrin, Koller & Parr, 2001
- Information Bottleneck models
  - Poupart & Boutilier, 2002
- Hierarchical POMDPs
  - Pineau & Thrun, 2000
  - Mahadevan & Theocharous 2002
- Many others

Belief Space Structure

The controller may be globally uncertain...

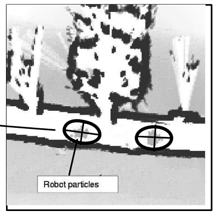
but not usually.



Belief Compression

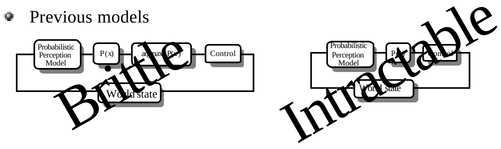
- If uncertainty has few degrees of freedom, belief space should have few dimensions

Each mode has few degrees of freedom

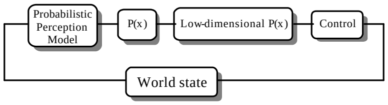


Control Models

- Previous models



- Compressed POMDPs



The Augmented MDP

- Represent beliefs using

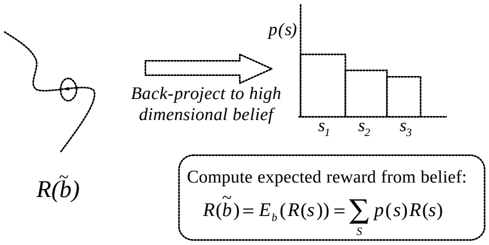
$$\tilde{b} = \left\langle \underset{s}{\operatorname{argmax}}\, b(s); H(b) \right\rangle$$

$$H(b) = -\sum_{i=1}^{N} p(s_i) \log_2 p(s_i)$$

- Discretise into 2-dimensional belief space MDP

Model Parameters

- Reward function

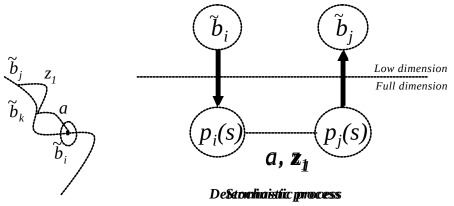


Compute expected reward from belief:

$$R(\tilde{b}) = E_b(R(s)) = \sum_S p(s)R(s)$$

Model Parameters

- Use forward model



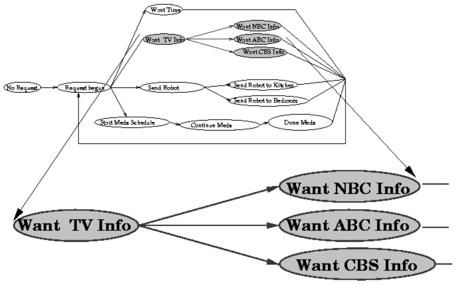
Augmented MDP

1. Discretize state-entropy space
2. Compute reward function and transition function
3. Solve belief state MDP

Nursebot Domain

- Medication scheduling
- Time and place tracking
- Appointment scheduling
- Simple outside knowledge e.g. weather
- Simple entertainment e.g. TV schedules
- *Sphinx speech recognition, Festival speech synthesis*

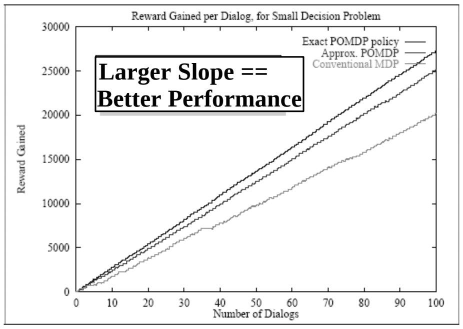
MDP Graph



An Example Dialogue

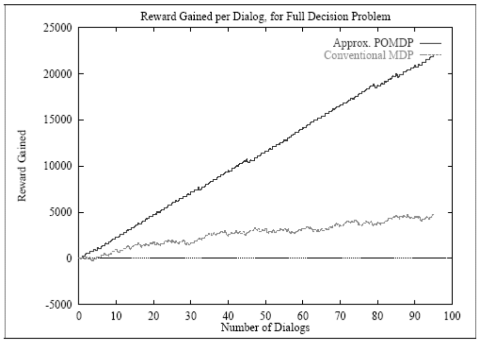
Observation	True State	Belief Entropy	Action	Reward
hello	request begun	0.406	say hello	100
what is like	start meds	2.735	**ask repeat**	-100
what time is it for will the	want time	0.490	say time	100
was on abc	want tv	1.176	ask which station	-1
was on abc	want abc	0.886	say abc	100
what is on nbc	want nbc	1.375	**confirm channel nbc**	-1
yes	want nbc	0.062	say nbc	100
go to the that pretty good what	send robot	0.864	ask robot where	-1
that that hello be	send robot bedroom	1.839	**confirm robot place**	-1
the bedroom any i	send robot bedroom	0.194	go to bedroom	100
go it eight a hello	send robot	1.110	ask robot where	-1
the kitchen hello	send robot kitchen	1.184	go to kitchen	100

Accumulation of Reward – Simulated 7 State Domain

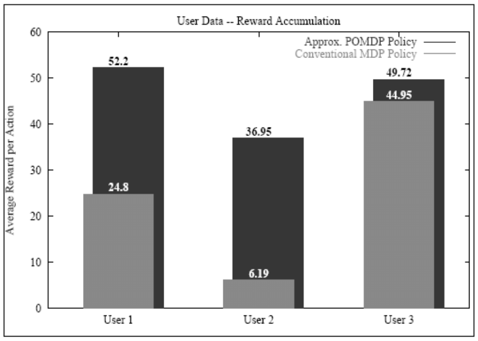


Larger Slope == Better Performance

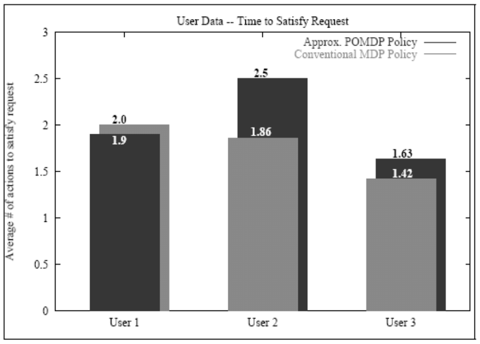
Accumulation of Reward – Simulated 17 State Domain



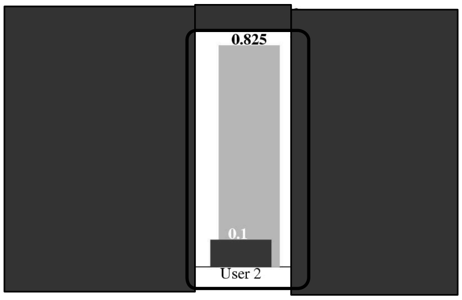
POMDP Dialogue Manager Performance



POMDP Dialogue Manager Performance

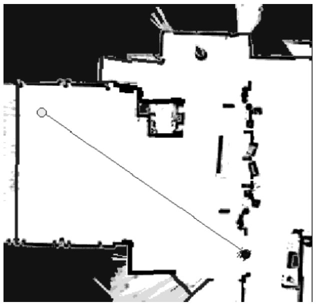


POMDP Dialogue Manager Performance



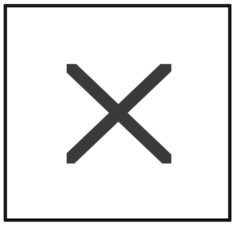
POMDPs for Navigation

- Conventional trajectories may not be robust to localization error



Estimated robot position ●
True robot position ●
Goal position ○

POMDPs for Navigation



Nursebot Pearl

Assisting Nursing
Home Residents

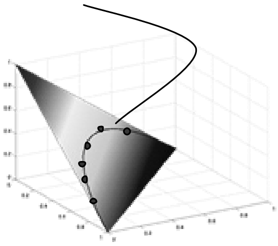
Longwood, Oakdale, May 2001
CMU/Pitt/Mich Nursebot Project

Talk Outline

- Robots in the real world
- Partially Observable Markov Decision Processes
- **Solving large POMDPs**
- Deployed POMDPs

Belief Compression

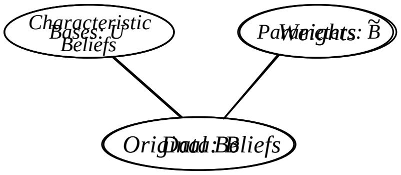
- Belief space is a low-dimensional sub-manifold



Full Belief Space

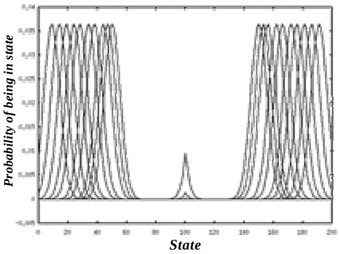
Dimensionality Reduction

- Principal Components Analysis



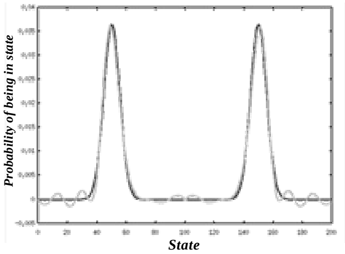
Principal Components Analysis

- Given belief B Î $\Re^n$, we want $\tilde{B}$ Î Â^m, m«n.
- Collection of beliefs drawn from 200 state problem

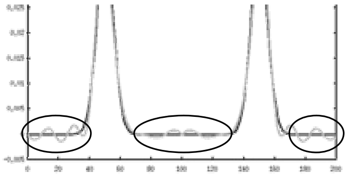


Principal Components Analysis

- Given belief B Î $\Re^n$, we want $\tilde{B}$ Î Â^m, m«n.
- m=9 gives this representation for one sample distribution

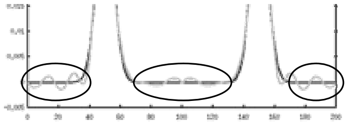


Principal Components Analysis



Many real world POMDP distributions are characterized by large regions of low probability.

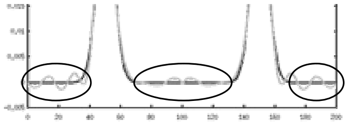
Principal Components Analysis



- PCA loss function:

$$L(b,U,\tilde{b}) = \left\| b - U\tilde{b} \right\|^2$$

Principal Components Analysis



- PCA data likelihood:

$$-\log P(b;U\tilde{b}) = -\log N(b;U\tilde{b})$$

Data are not normally distributed

Principal Components Analysis

- Minimizing PCA loss function:

$$L(b,U,\tilde{b}) = \left\| b - U\tilde{b} \right\|^2$$

- Equivalent to minimizing:

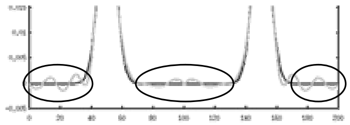
$$-\log P(b;\Theta) = -\log N(b;\Theta)$$

- Equivalent to minimizing:

$$\cancel{\log P_0(b) - F(b)} + B_F(b \,\|\, g(q))$$

Collins, Dasgupta & Schapire, 2000

Principal Components Analysis



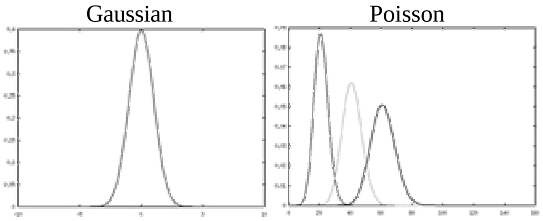
- PCA data likelihood:

$$-\log P(b;U\tilde{b}) = -\log Poisson(b;U\tilde{b})$$

Use a Poisson likelihood model

Collins, Dasgupta & Schapire, 2000

Different Error Functions



$$p(x) \propto e^{\frac{-(x-m)^2}{s^2}} \qquad p(x) \propto e^{-l}\, l^x$$

Solving for Bases and Parameters

- Bregman Divergence for Poisson error model:

$$B_F(b \,\|\, U\tilde{b}) = e^{(U\tilde{b})} - b \circ U\tilde{b}$$

Solving for Bases and Parameters

- Bregman Divergence for Poisson error model:

$$B_F(b \,\|\, U\tilde{b}) = e^{(U\tilde{b})} - b \circ U\tilde{b}$$

$$\frac{\partial B_F(b \,\|\, U\tilde{b})}{\partial U} = \frac{\partial}{\partial U} F(U\tilde{b}) - b \circ U\tilde{b}$$

$$= e^{(U\tilde{b})} b^T - b\tilde{b}^T$$

$$\frac{\partial B_F(b \,\|\, U\tilde{b})}{\partial \tilde{b}} = \frac{\partial}{\partial \tilde{b}} F(U\tilde{b}) - b \circ U\tilde{b}$$

$$= U^T e^{(U\tilde{b})} - U^T b$$

Solving for Bases and Parameters

- Loss function for Poisson error model:

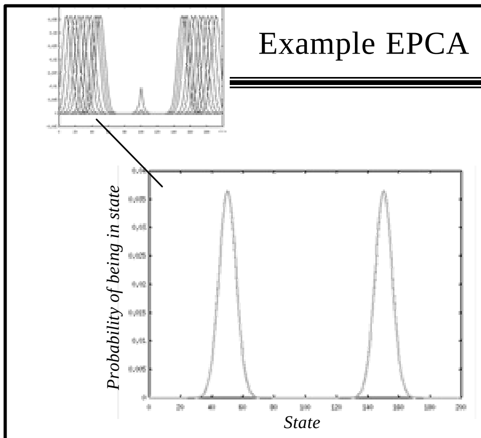
$$-\log(x; e^l) \propto e^l - x\,l$$

$$\arg\min -\log(b; U\tilde{b}) = \arg\min\, e^{(U\tilde{b})} - b \circ U\tilde{b}$$

- Equivalent to minimising:

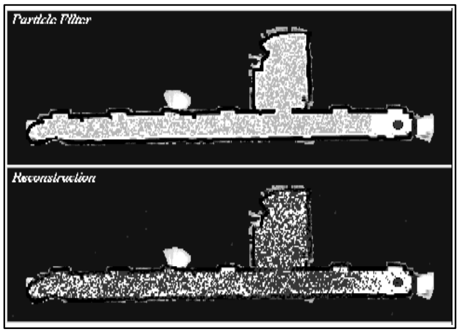
$$\arg\min \| D^{-1/2}(b - \exp(U\tilde{b})) \|$$

Example EPCA



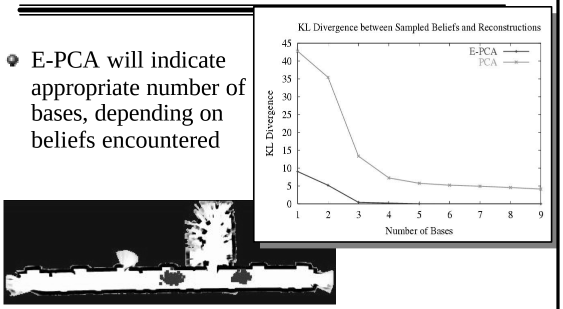
4 bases

Example Reduction

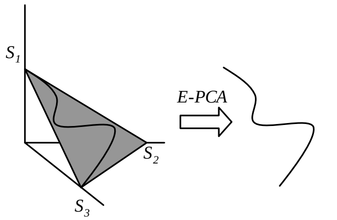


Finding Dimensionality

- E-PCA will indicate appropriate number of bases, depending on beliefs encountered

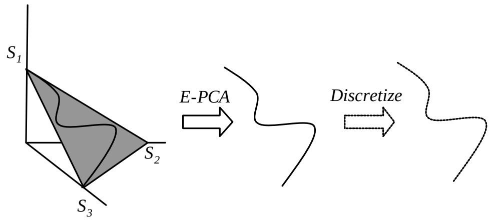


Planning



Original POMDP *Low-dimensional belief space*

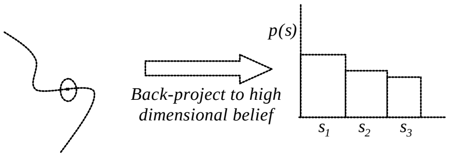
Planning



Original POMDP *Low-dimensional belief space* *Discrete belief space MDP*

Model Parameters

- Reward function



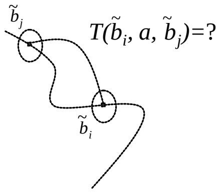
$R(\tilde{b})$

Compute expected reward from belief:

$$R(\tilde{b}) = E_b(R(s)) = \sum_s p(s)R(s)$$

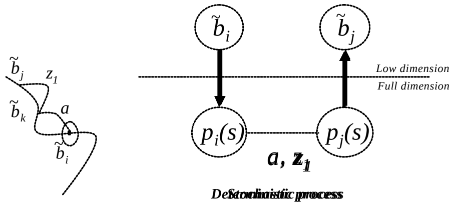
Model Parameters

- Transition function



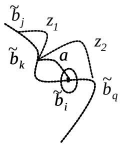
Model Parameters

- Use forward model



Model Parameters

- Use forward model

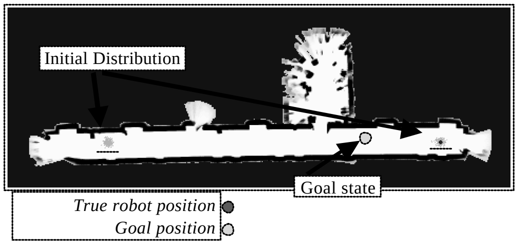


$$T(\tilde{b}_i, a, \tilde{b}_j) \propto p(z|s)b_i(s|a) \quad \text{if } b_j(s) = b_i(s|a,z)$$
$$= 0 \quad \text{otherwise}$$

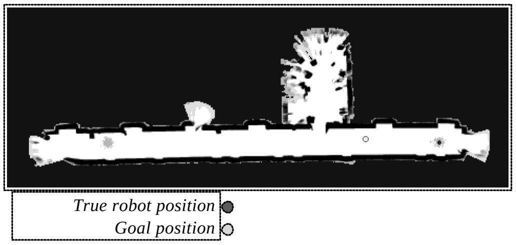
E-PCA POMDPs

1. Collect sample beliefs
2. Find low-dimensional belief representation
3. Discretize
4. Compute reward function and transition function
5. Solve belief state MDP

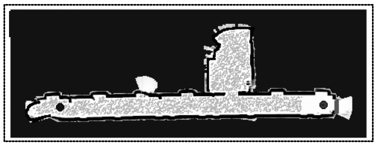
Robot Navigation Example



Robot Navigation Example



People Finding as a POMDP



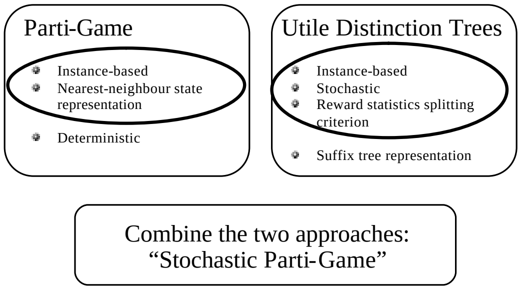
- Factored state space
  - 2 dimensions: fully-observable robot position
  - 6 dimensions: distribution over person positions

Regular grid gives ~ 10^{16} states

Variable Resolution Discretization

- Variable Resolution Dynamic Programming (1991)
- Parti-game (Moore, 1993)
- Variable Resolution Discretization (Munos & Moore, 2000)
- POMDP Grid-based Approximations (Hauskrecht, 2001)
- Improved POMDP Grid-based Approximations (Zhou & Hansen, 2001)

Variable Resolution



Parti-Game

- Instance-based
- Nearest-neighbour state representation
- Deterministic

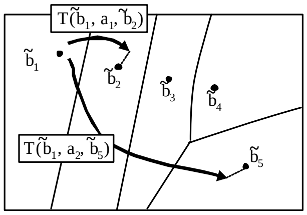
Utile Distinction Trees

- Instance-based
- Stochastic
- Reward statistics splitting criterion
- Suffix tree representation

Combine the two approaches:
“Stochastic Parti-Game”

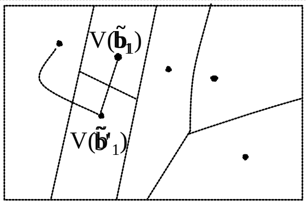
Variable Resolution

- Non-regular grid using samples


$T(\tilde{b}_1, a_1, \tilde{b}_2)$
$\tilde{b}_1$
$\tilde{b}_2$
$\tilde{b}_3$
$\tilde{b}_4$
$T(\tilde{b}_1, a_2, \tilde{b}_5)$
$\tilde{b}_5$

- Compute model parameters using nearest-neighbour

Refining the Grid

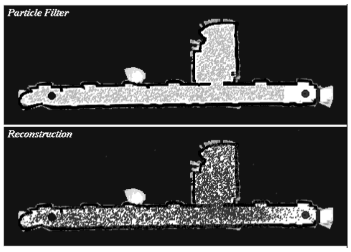


- Sample beliefs according to policy
- Construct new model
- Keep new belief if $V(\tilde{b}'_1) > V(\tilde{b}_1)$

The Optimal Policy

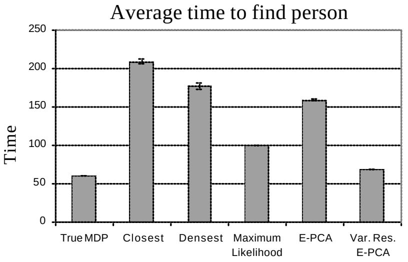
Original distribution

Reconstruction using EPCA and 6 bases

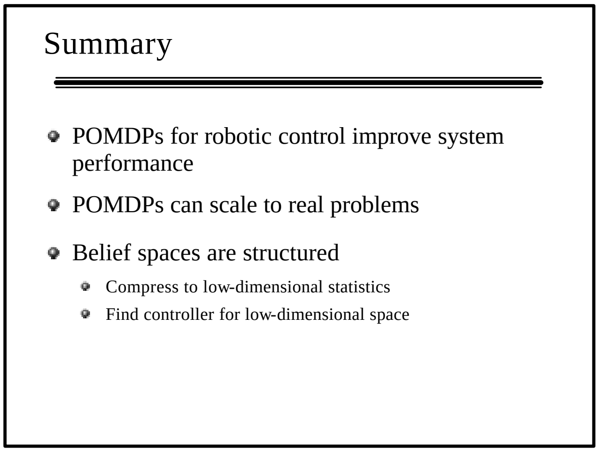


Robot position ●
True person position ●

Policy Comparison

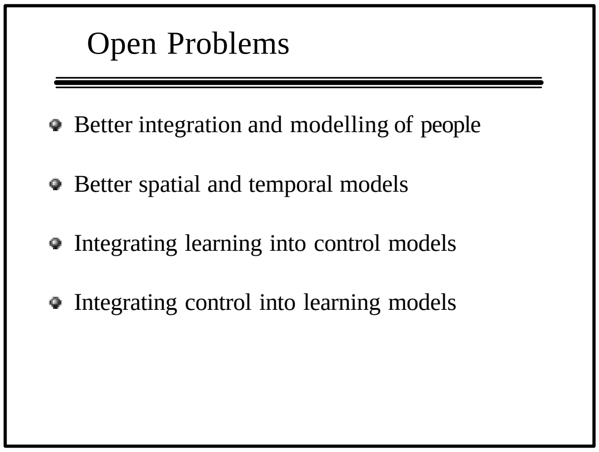


E-PCA: 72 states
Var. Res. E-PCA: 260 states



Summary

- POMDPs for robotic control improve system performance
- POMDPs can scale to real problems
- Belief spaces are structured
  - Compress to low-dimensional statistics
  - Find controller for low-dimensional space



Open Problems

- Better integration and modelling of people
- Better spatial and temporal models
- Integrating learning into control models
- Integrating control into learning models