Robotic Hide and Seek

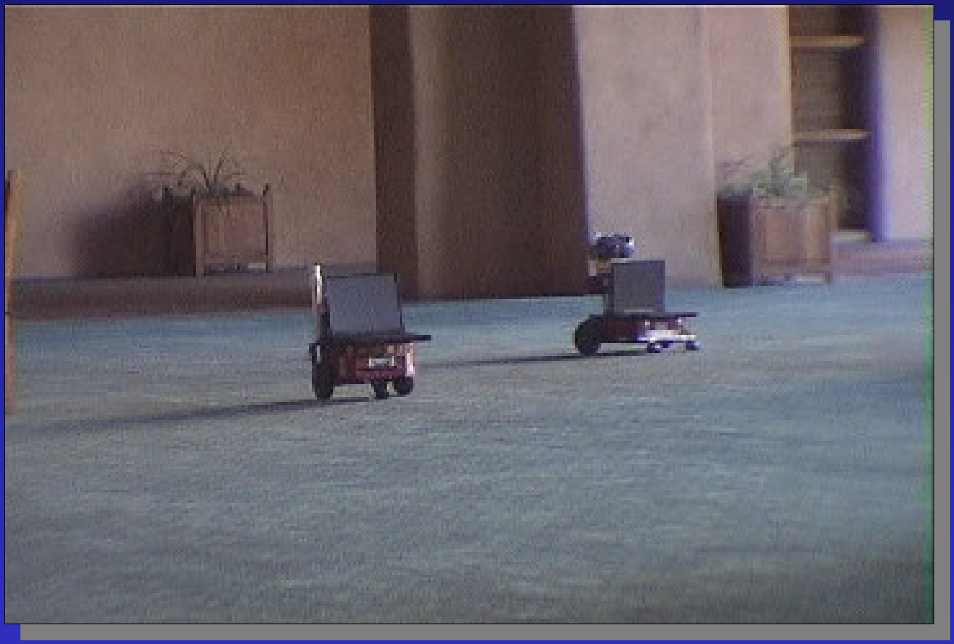
Geoff Gordon

Multi-robot reconnaissance and surveillance

Geoff Gordon

Multi-robot planning

- Huge cross-product state spaces
- Partially observable
- Nonstationary



- Coordination w/ teammates

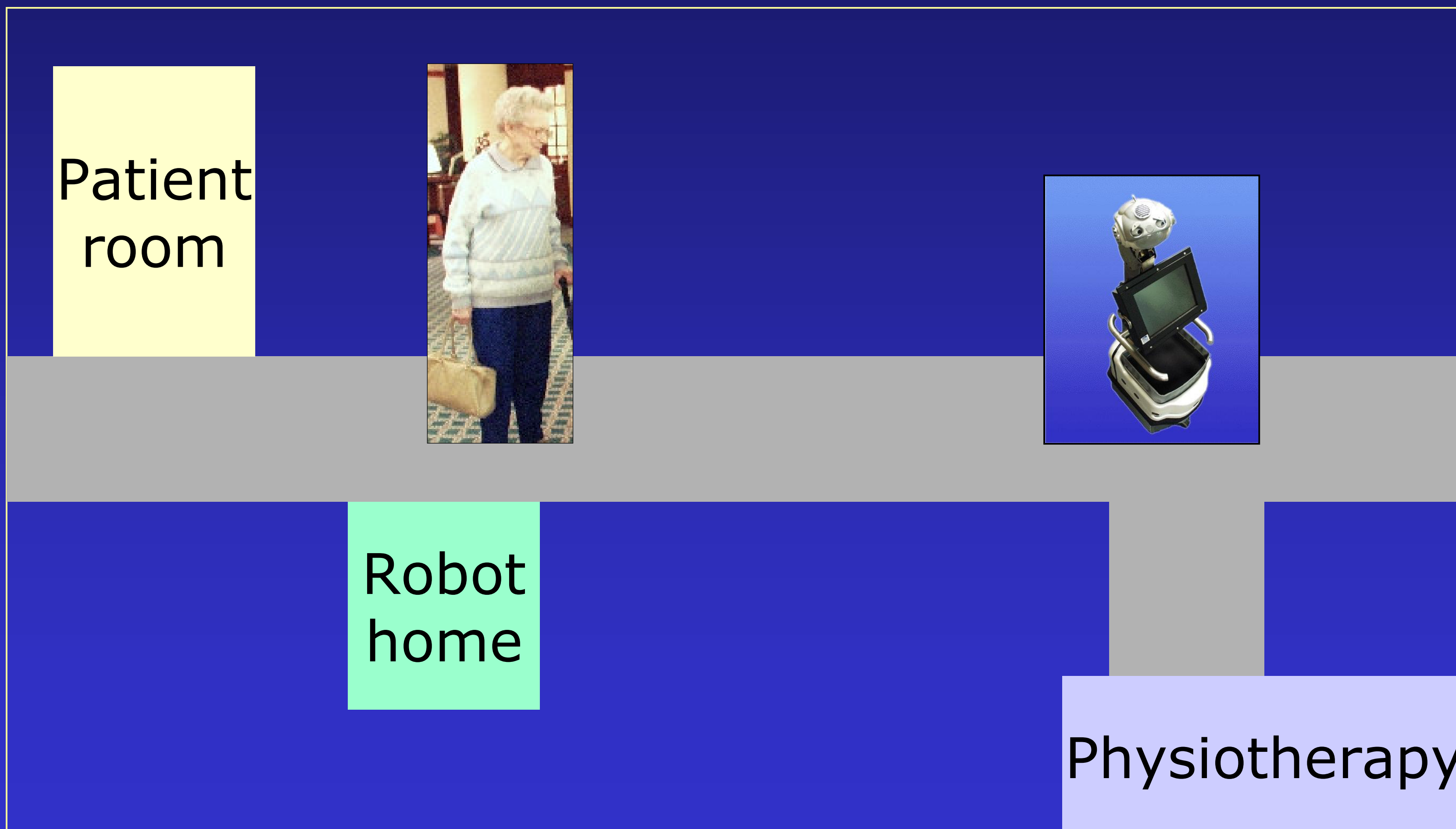
Outline

Factorize and approximate

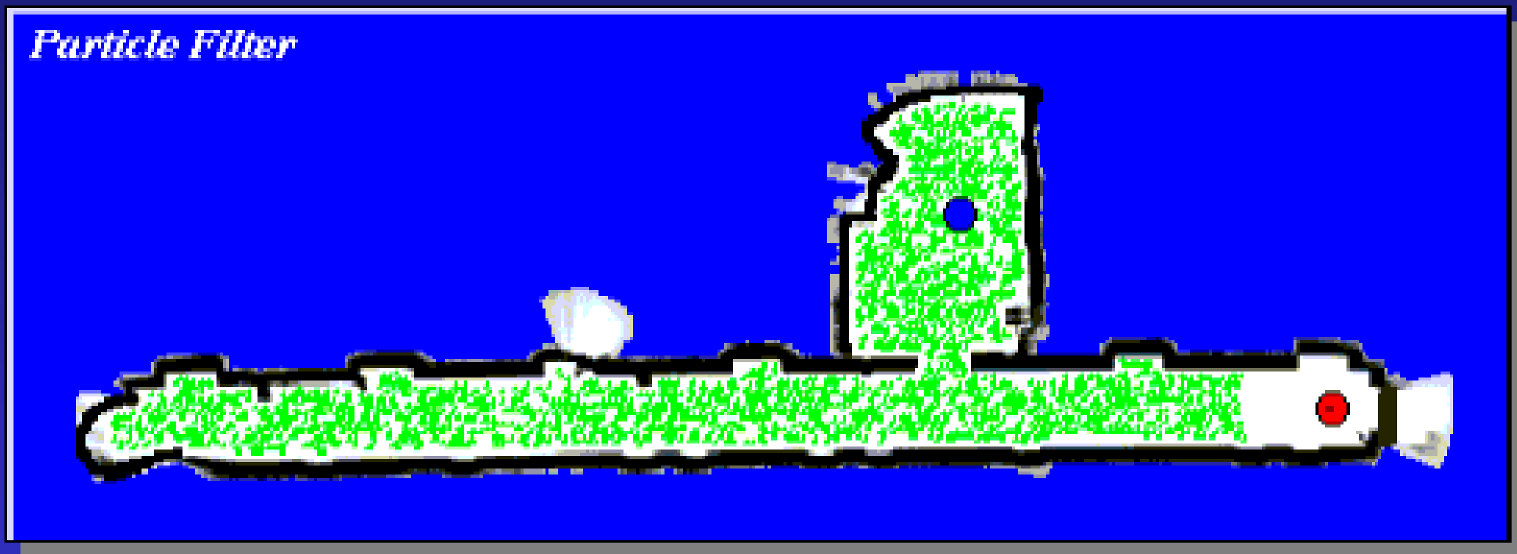
- Belief compression
  - 1 on 1 hide and seek
  - model opponent as hidden state (POMDP)
- Auctions for coordination
  - mixed initiative team reconnaissance problem
  - model teammates as a market

Searching for robots

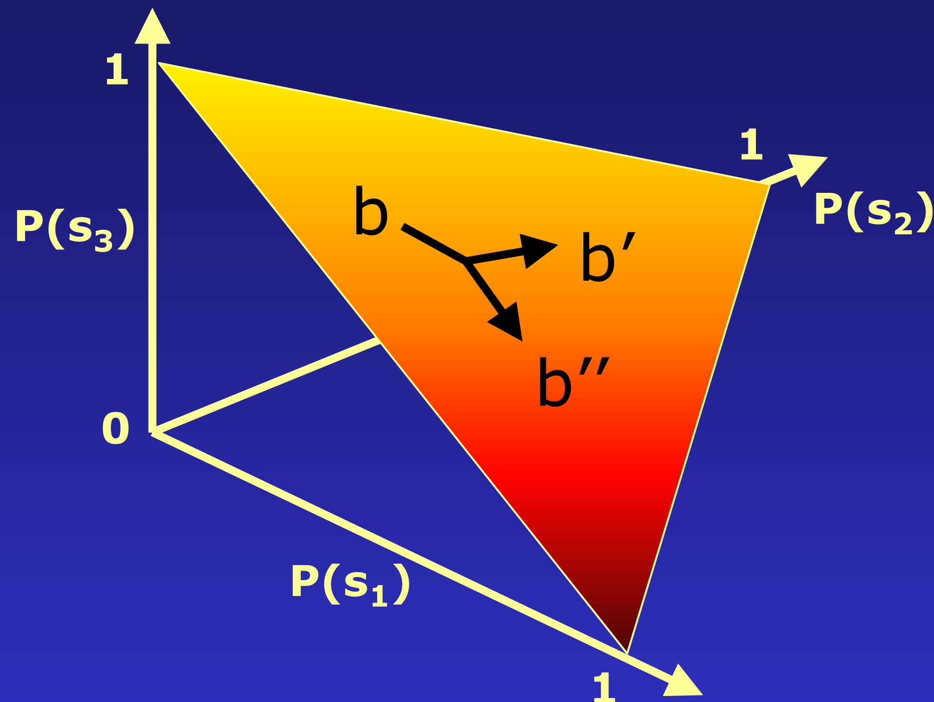
Searching for patients



Example run



Belief dynamics



Transitions: $b \rightarrow bT_a$

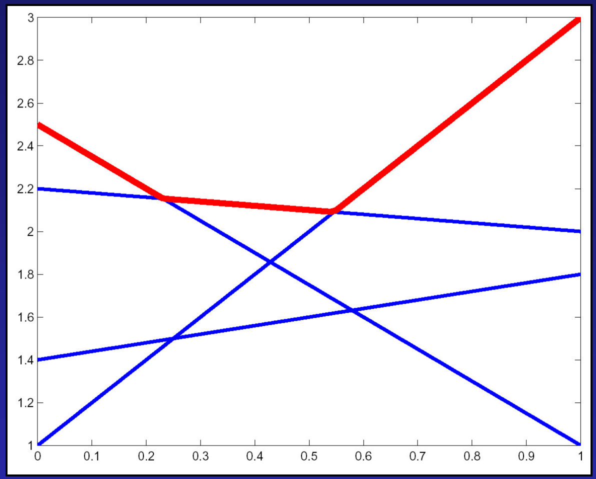
Observations: $b \rightarrow w_z \times b$

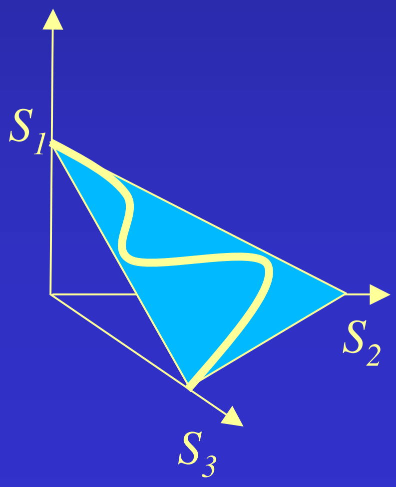
Value iteration

- Planning = finding value function
- $V(b)$ = expected future cost starting at b
- V is best possible heuristic fn when searching for actions

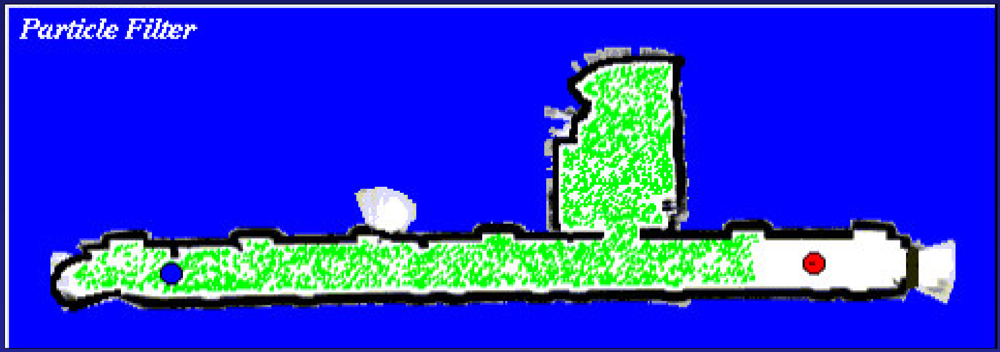
Planning in POMDPs

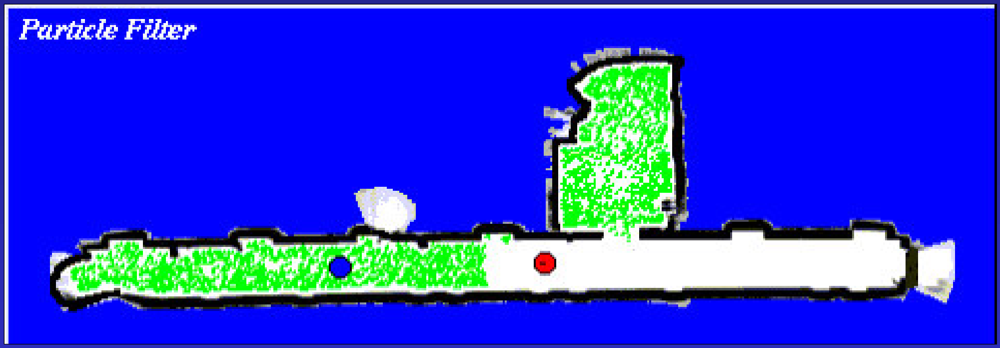
- POMDP VI
  - convexity
  - Littman, Pineau, Poupart, ...
- Transform to MDP and approximate
  - value approximation
  - belief compression

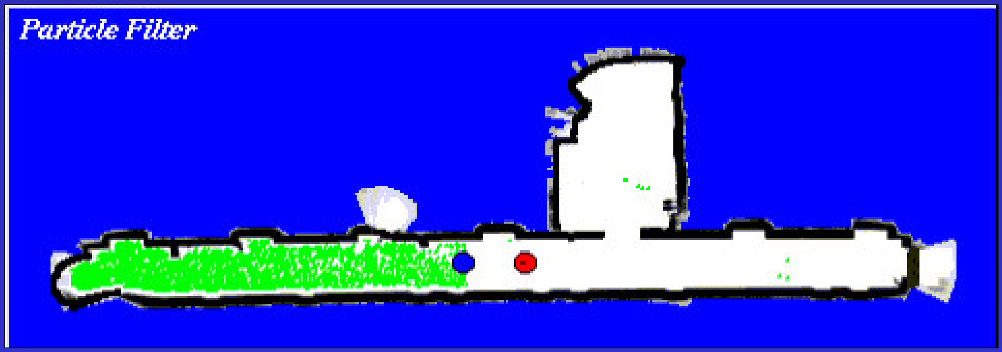




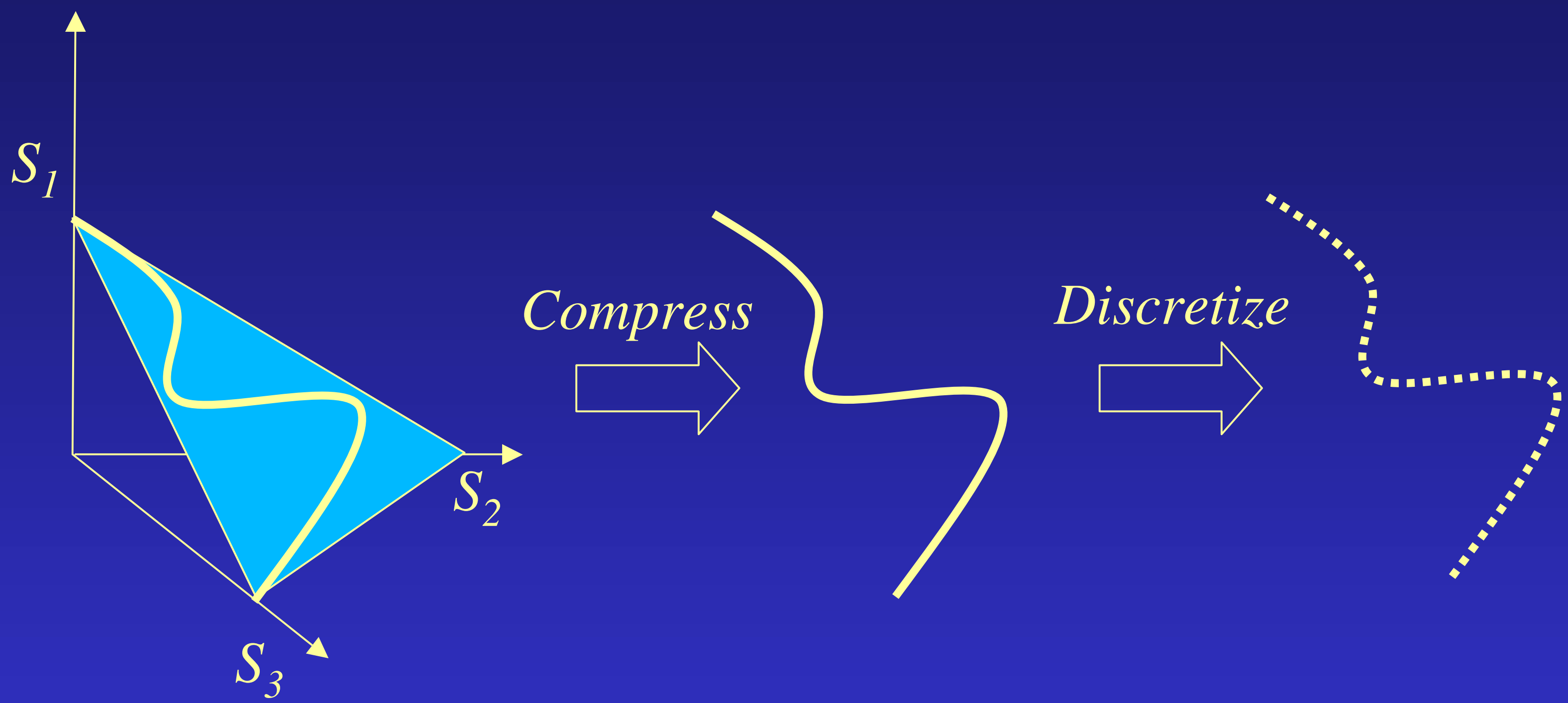
Belief structure







Belief compression



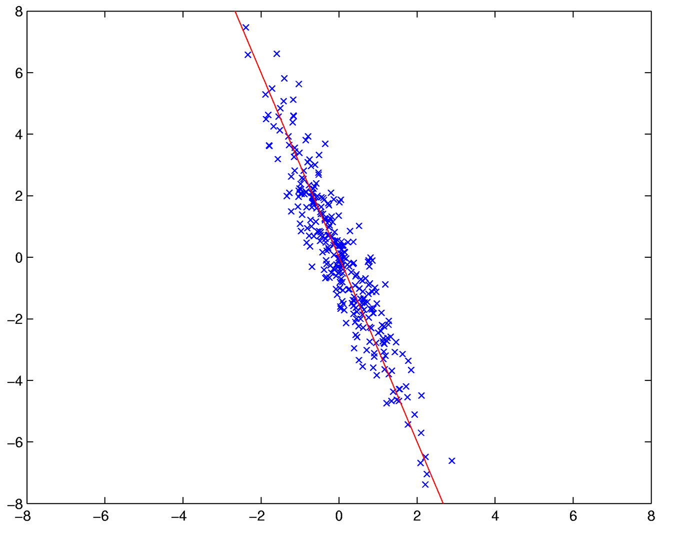
Original POMDP

Low-dimensional belief space $\tilde{B}$

Discrete belief space MDP

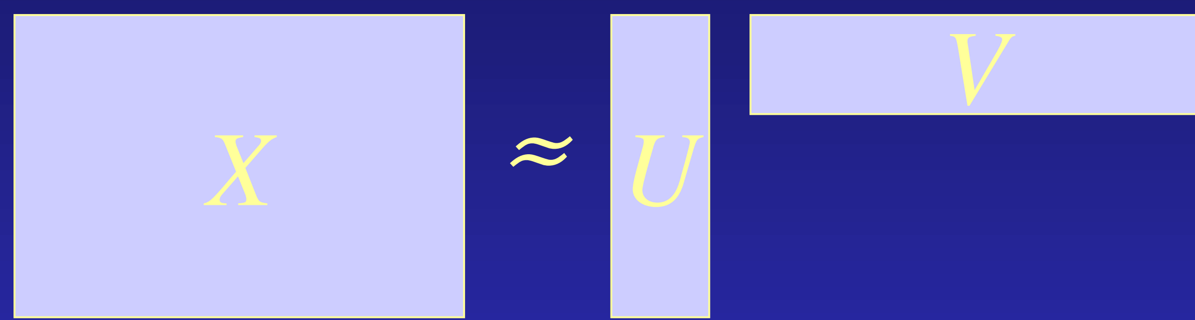
Principal Components

- Pattern recognition algorithm
  - text retrieval, bibliometrics, eigenfaces, compression, ...



- Finds subspace near data
  - features which reconstruct data
  - model: hyperplane plus noise

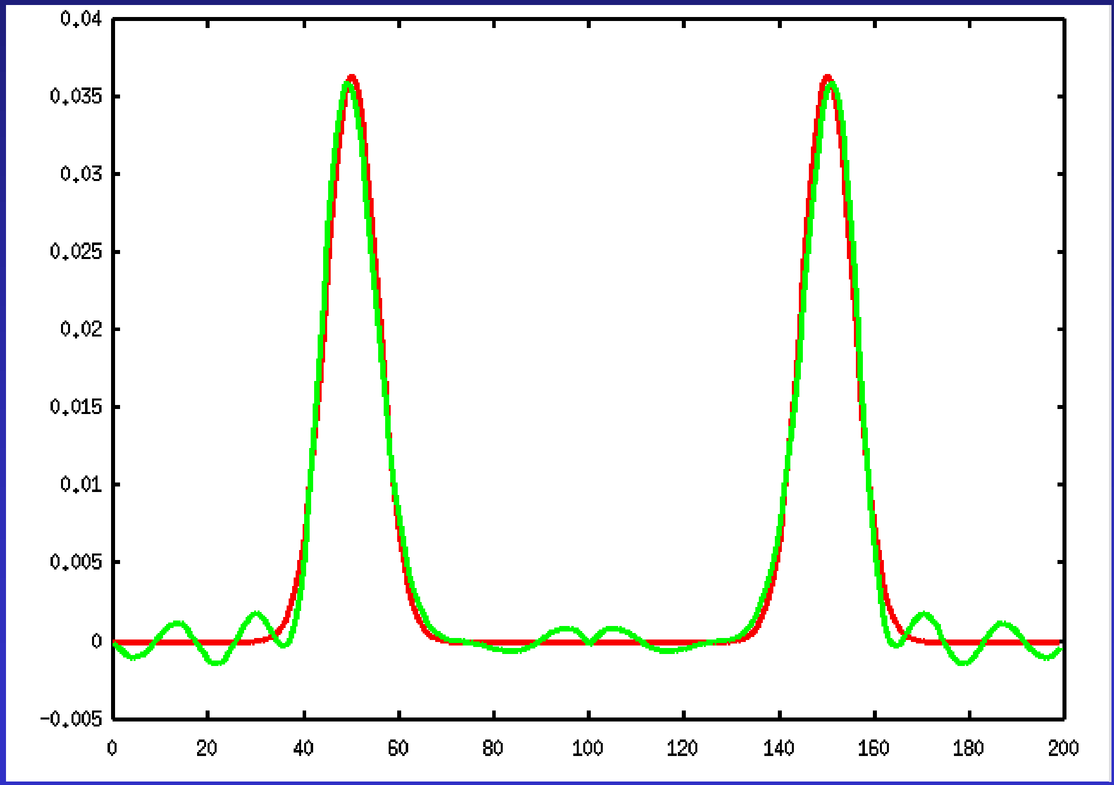
PCA: matrix factorization



X: data (1 per column)
U: feature vectors
V: feature weights

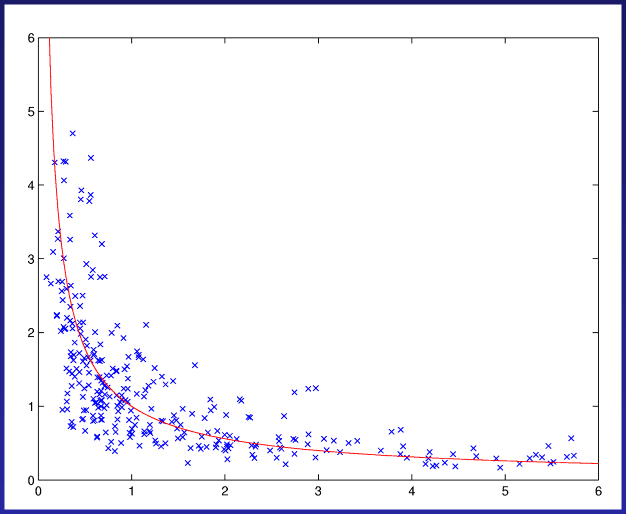
Problem with PCA

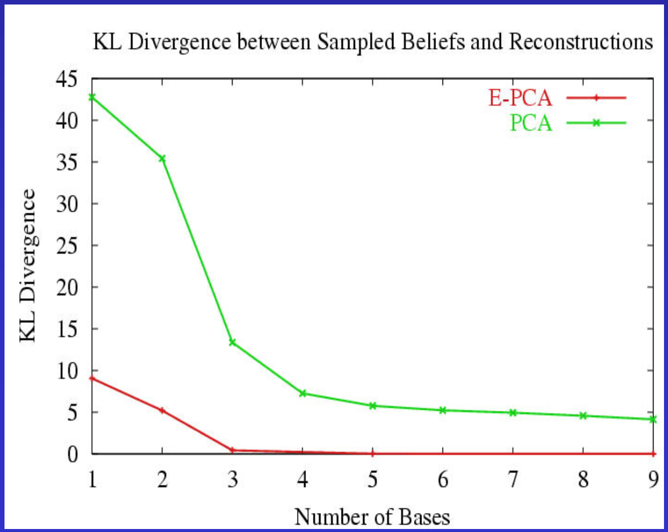
9-dimensional basis



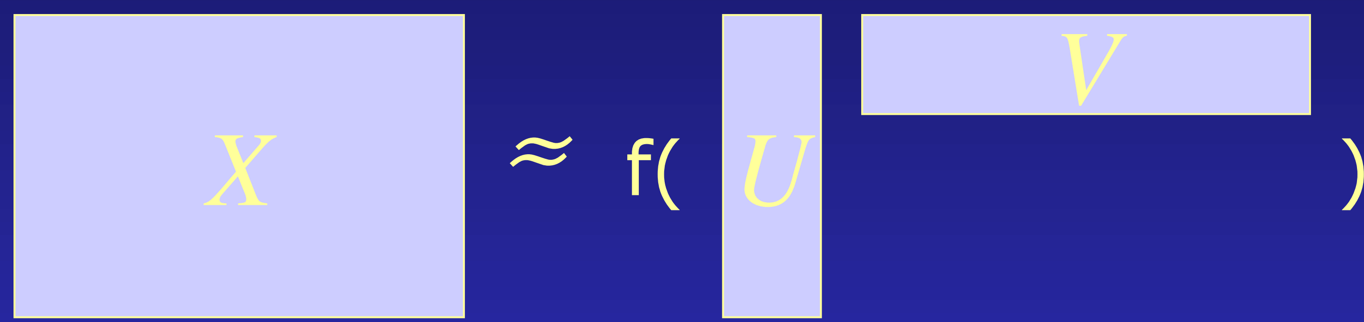
Generalized^2 Linear^2 Model

- Fits submanifold like PCA
- Nonlinear like GLM
- Nonlinearities can allow better match to data with fewer feature vectors





$(GL)^2M$: nonlinear factorization



$$U = g(A) \quad V = h(B)$$

X: data (1 per column)

U: feature vectors

V: feature weights

Link functions

- Link function *f* is arbitrary, componentwise monotone
- For this application: *f* = exp
  - enforces positive probabilities
  - pays more attention to errors near zero
- Inferring sufficient statistics of an exponential family

Comparison of objectives

PCA

$$\|X - UV\|^2$$

$$\|UV\|^2 / 2 - X \circ UV + const$$

(GL)²M w/ exp

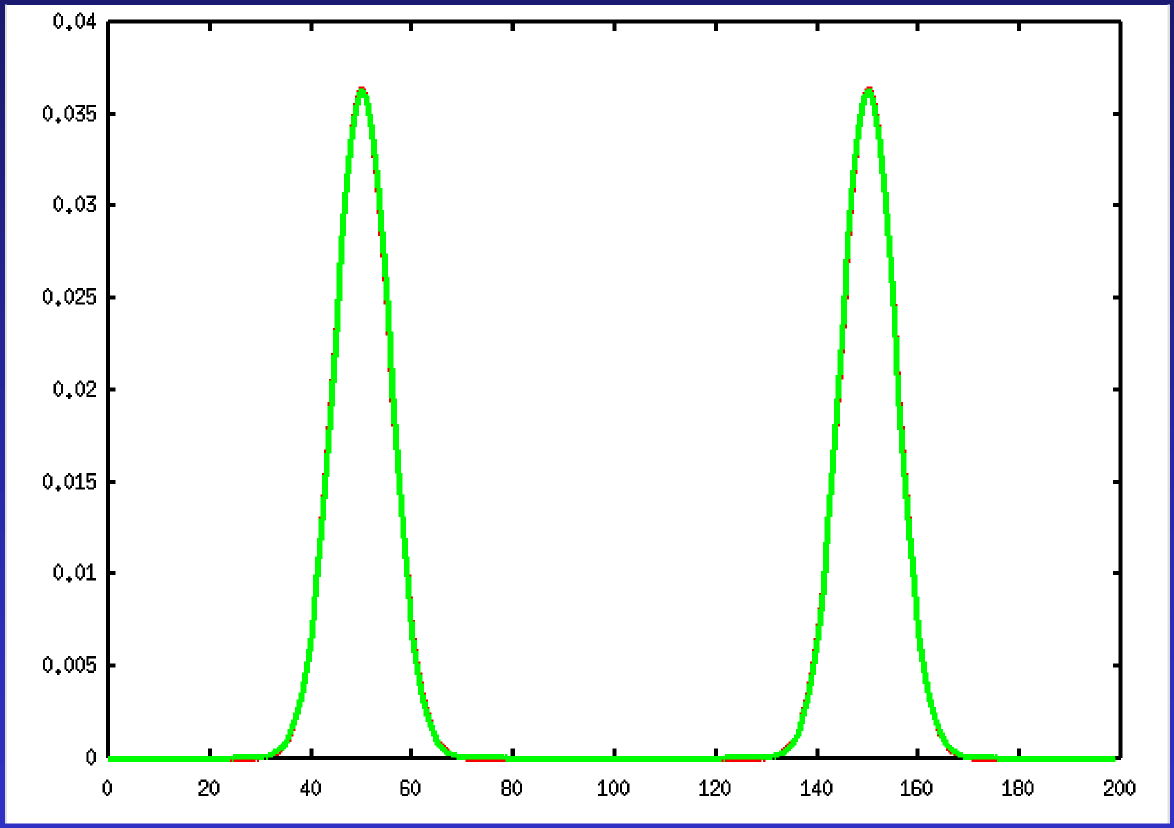
$$D_{KL}(X \mid e^{UV})$$

$$\sum e^{UV} - X \circ UV + const$$

$(GL)^2M$ advantages

- Expressive model
  - PCA, sPCA, ICA, GLIM, NMF, ...
- Efficient algorithm
  - alternating minimization
  - Newton's method
  - similar to IRLS

Belief reconstruction performance

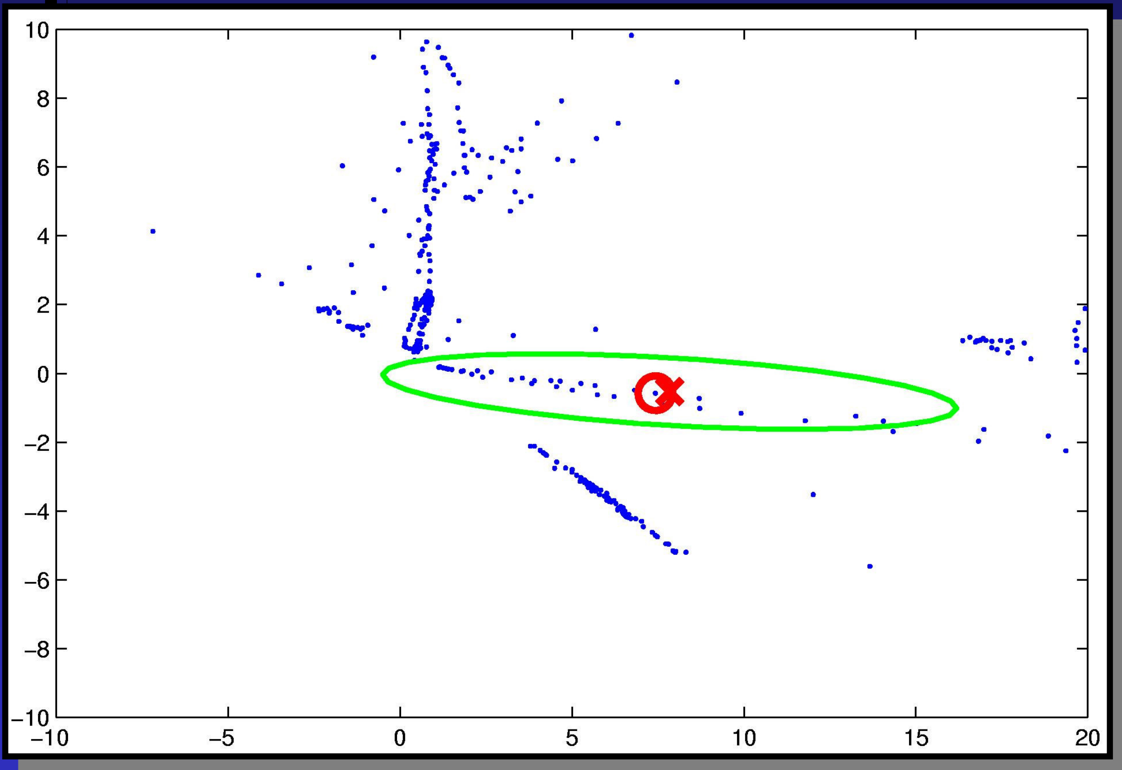


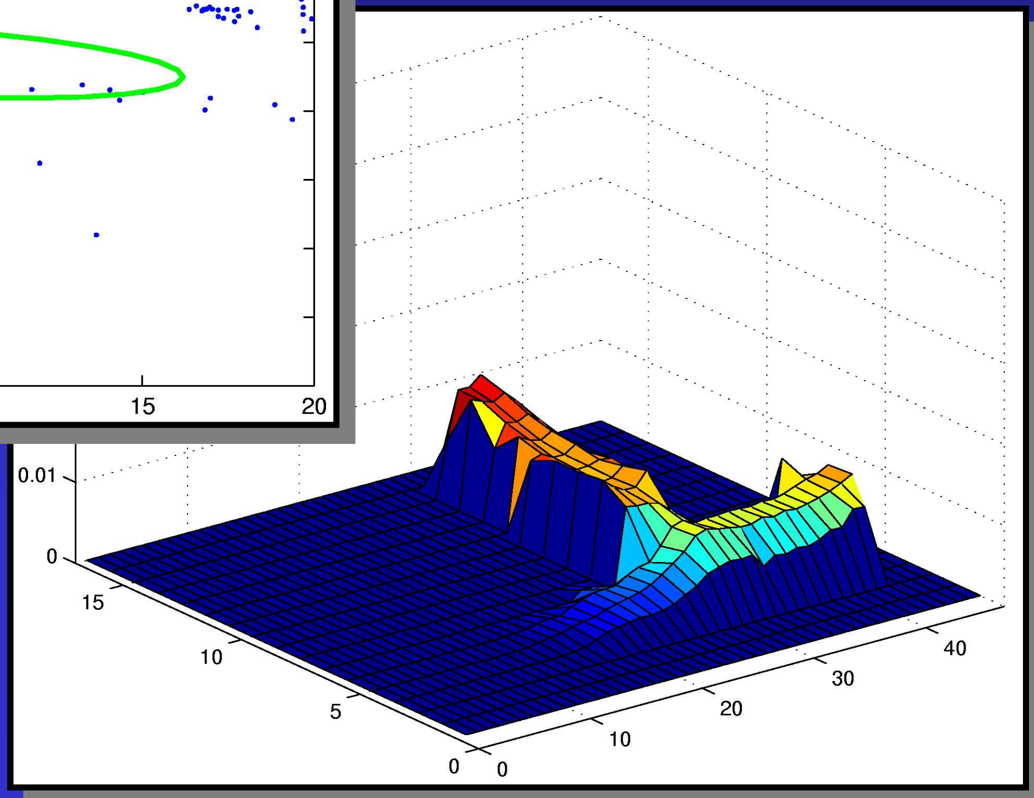
Probability

State index

4 bases

Belief visualization

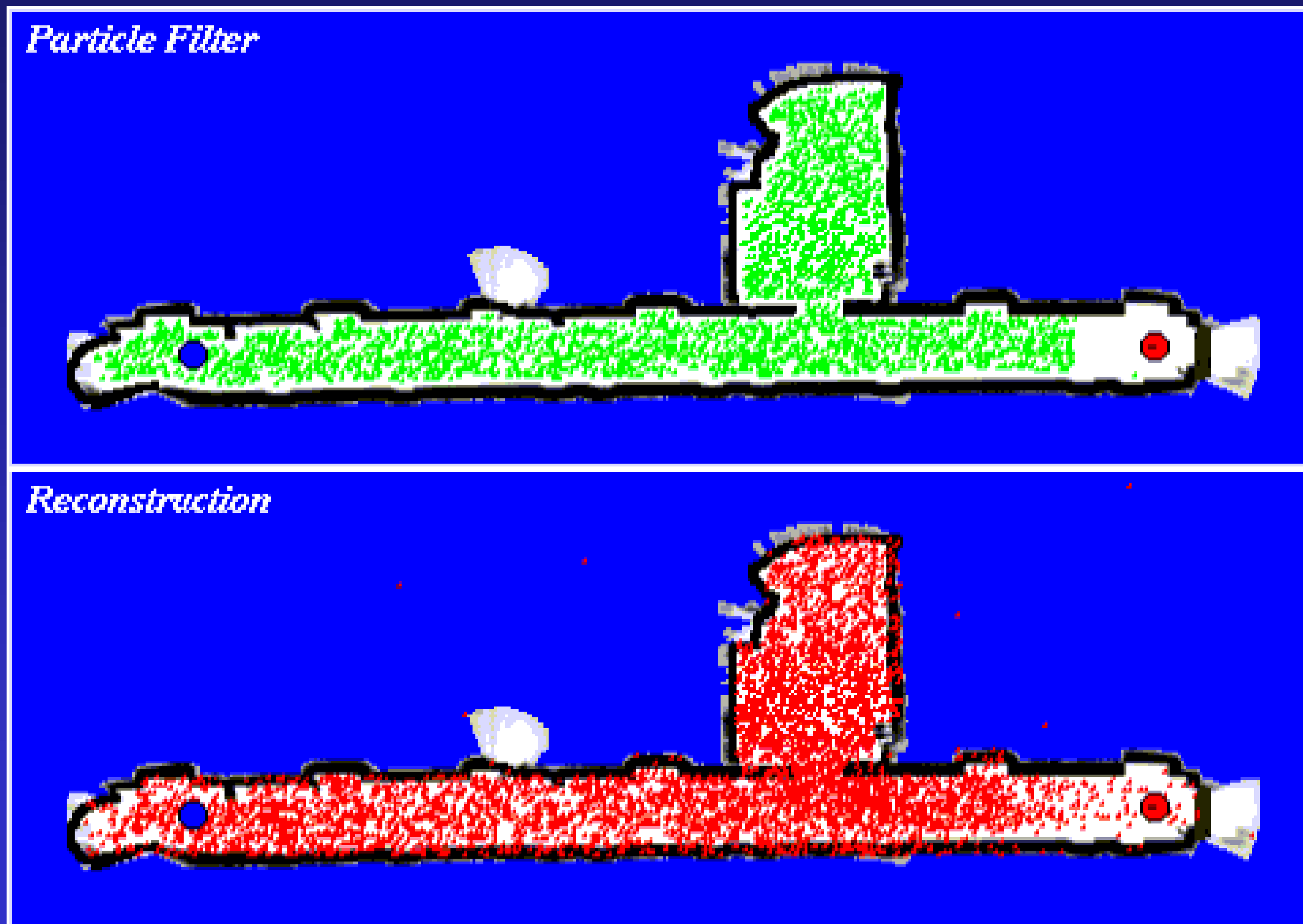




Does it let us plan?

- Drive robot around, collect beliefs
- Learn 6 factors
- Compute approximate value fn
  - one value for each belief sample
  - *k* nearest neighbor
  - nearest in 6D space
- Evaluate greedy policy

Learned policy



Policy Comparison

Belief compression

- Finding low-D representation of high-D belief states
- Nonlinear component analysis model & algorithm
- Discretize & plan in low-D space

[Gordon, NIPS 02]
[Roy, Gordon & Thrun, NIPS 02]

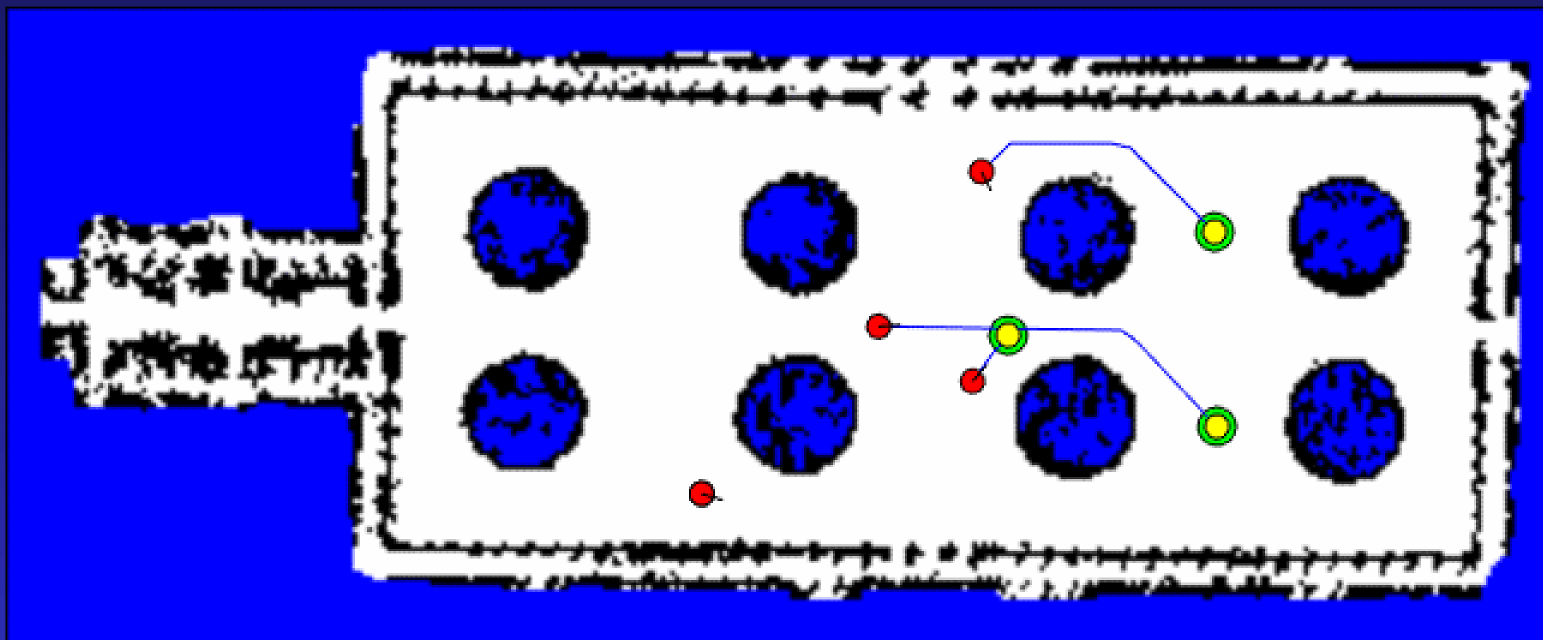
Stanford quad

Stanford quad

600 meters

Map credit: Mike Montemerlo

Robots in the quad



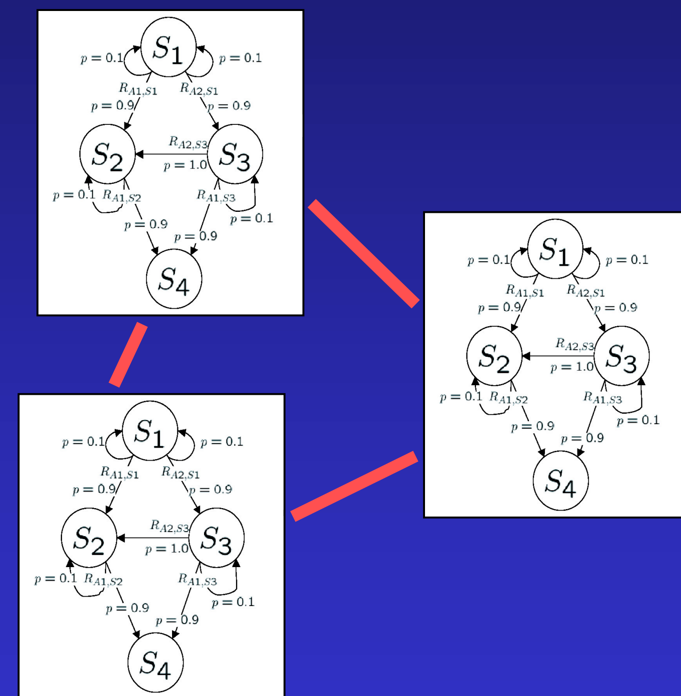
- Team of robots under high-level control by human operator
- Human provides recon goals/rewards, robots decide who goes where

Problem details

- Several possible “flavors”
- All goals known in advance: TSP
- Much faster to move than to serve a goal: queueing problem
- In between: goals arrive often and randomly
  - can’t plan too far ahead
  - “getting out of position”

Loose coupling between agents

- Joint MDP has exponential size
  - in #agents
  - in #goals
- Interaction is limited
- Agents interact only by competing for goals



Market abstraction

- Each agent remembers limited state (own position, current destination)
- Assumes complete control
  - can move, select next goal at will
- Conflict? $\Rightarrow$ define a resouce ← hard part
- Assign a price to each resource

Why markets?

- Flexible:
  - right to visit goal
  - fuel
  - network bandwidth
  - taxi ride
  - right to pick up passenger
  - bridge toll
  - right to explore
  - machine time
  - responsibility to help teammate achieve X
- Help make planning efficient

Basic auction algorithm

- Robot *i* in state x_i considers action or sequence of actions a_i
- Estimates future discounted cost conditioned on a_i: $Q(x_i, a_i)$
- Bids $Q(x_i, a_i)$ for doing a_i
- Auctioneer examines feasible joint actions $(a_1, a_2, \ldots)$
- Chooses joint action w/ lowest sum of bids

hard parts

Hard parts

- Choose state, resources (design):
  - hand pick (criteria on next slide)
- Estimate Q (prediction):
  - hand-designed heuristic
  - ignore future interaction
  - or wait a few slides
- Pick joint action (clearing):
  - break into smaller auctions
  - exhaustive search

How to choose states and resources?

- Market abstraction: when I need a resource I can buy it at a predictable price
- Good resource = efficient market
  - always other agents available to trade (no “thin markets”)
  - price not determined by a single agent’s actions (no monopolies)
- Good state = helps predict prices

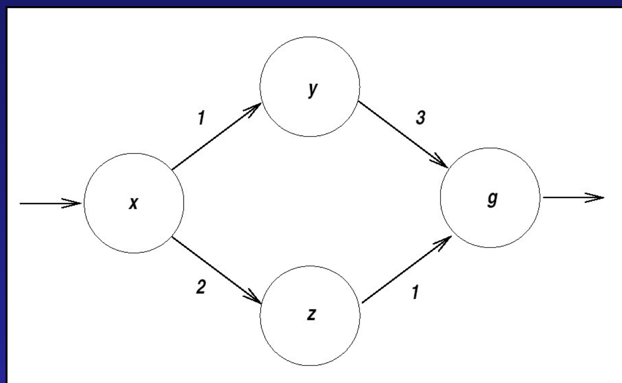
Insight

Market abstraction

$\Rightarrow$

Efficient planning
is possible

MDP as linear program



$$\max_{\mathbf{f}_a} \; \sum_a \mathbf{c}_a \cdot \mathbf{f}_a$$
$$\sum_a \mathbf{f}_a - \gamma \sum_a T_a^T \mathbf{f}_a = \alpha$$
$$\forall a: \; \mathbf{f}_a \geq 0$$

minimize $f_{xy} + 2f_{xz} + 3f_{yg} + f_{zg}$ subject to

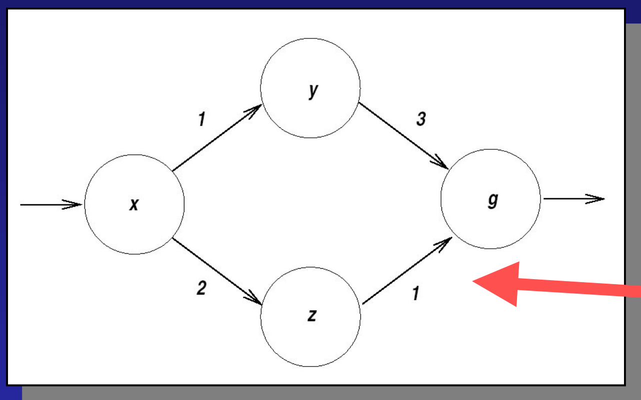
$$\begin{array}{rrrrrrcl}
-f_{xy} & -f_{xz} & & & & +1 & = & 0 \\
f_{xy} & & -f_{yg} & & & & = & 0 \\
& f_{xz} & & -f_{zg} & & & = & 0 \\
& & f_{yg} & +f_{zg} & -f_g & & = & 0 \\
\end{array}$$

$$f_{xy}, f_{xz}, f_{yg}, f_{zg}, f_g \;\; \geq \;\; 0$$

minimization of cost

conservation of probability

Resource constraints



fuel supply: 1 unit per trial

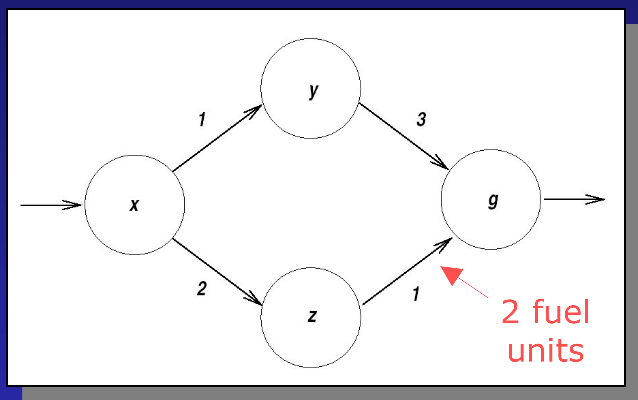
this edge uses 2 units of fuel

$$2 f_{zg} \leq 1$$

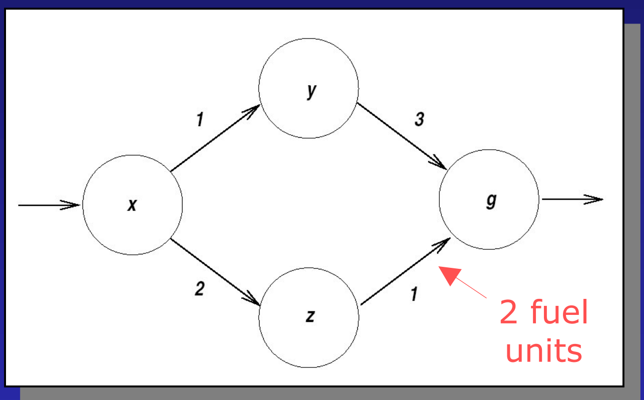
- W/o constraint: use lower path
- With constraint: randomize 50-50

With two agents

Robot 1



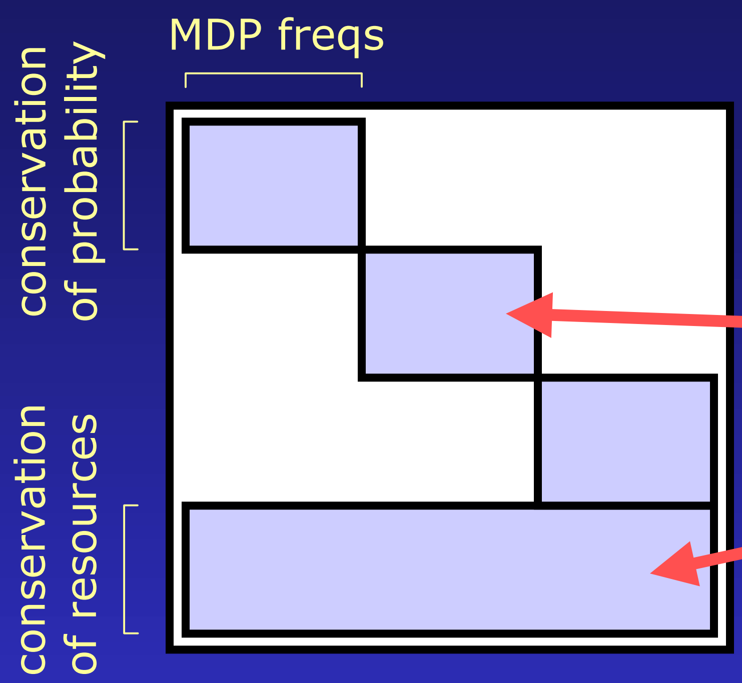
Robot 2



$$2\,f_{zg,1} + 2\,f_{zg,2} \leq 1$$

- Each agent uses better path 25% of time

In general



$$\max_{\mathbf{f}_i} \sum_i \mathbf{c}_i \cdot \mathbf{f}_i$$
$$\forall i: \quad A_i \mathbf{f}_i = \mathbf{b}_i$$
$$(*) \quad \sum_i C_i \mathbf{f}_i = \mathbf{d}$$
$$\forall i: \quad \mathbf{f}_i \geq \mathbf{0}$$

- Mostly independent single-agent MDPs
- Coupling via resource constraints

Dantzig-Wolfe decomposition

master

$$
\begin{aligned}
& \max_{\mathbf{q}_i} \ \textstyle\sum_i \mathbf{c}_i^{\mathrm{T}} F_i \mathbf{q}_i \\
(*) \quad & \textstyle\sum_i C_i \left(F_i \mathbf{q}_i\right) = \mathbf{d} \\
& \forall i: \quad \mathbf{q}_i \geq 0 \\
& \forall i: \quad \textstyle\sum_j q_{ij} = 1
\end{aligned}
$$

slave

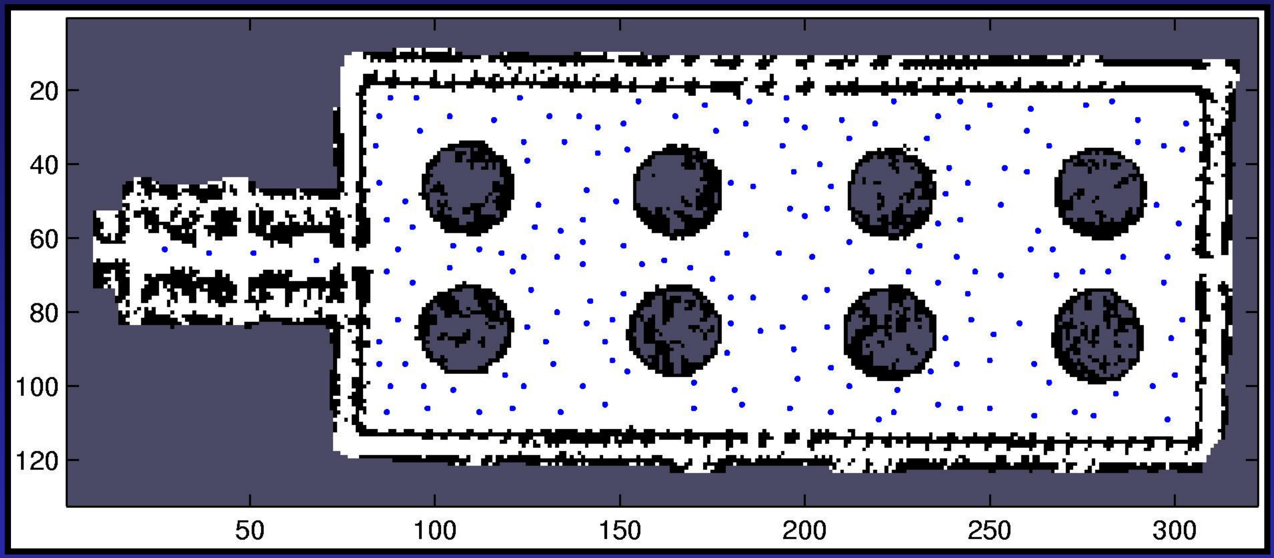
$$
\begin{aligned}
& \max_{\mathbf{f}_i} \ (\mathbf{c}_i^{\mathrm{T}} - \mathbf{p}^{\mathrm{T}} C_i)\mathbf{f}_i \\
& A_i \mathbf{f}_i = \mathbf{b}_i \\
& \mathbf{f}_i \geq \mathbf{0}
\end{aligned}
$$

- Divide into master and slave problems
- Slave program: plan for one robot given resource prices
- Repeatedly solve slave, plug resulting policy into master
- Master decides how to combine single-robot solutions, produces new prices

Factorized planner

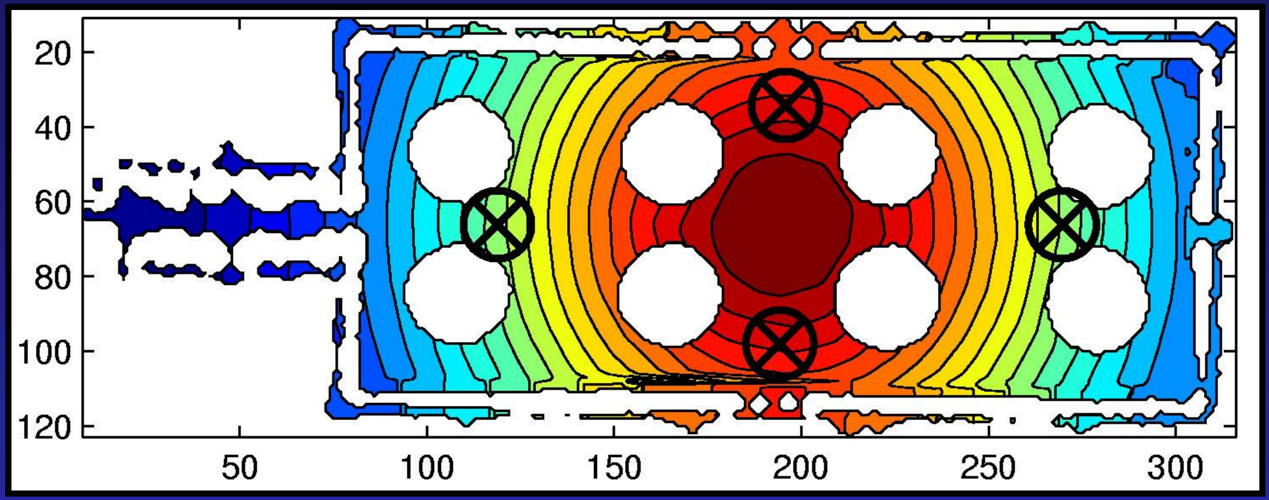
- Repeat until no change:
  - send prices p to robots
  - robot i plans w/ costs c_i - $C_i'p$
  - frequencies $f_{it}(s,a)$, values $v_{it}(s)$
  - send usage $C_i f$, cost $c_i f$ to master
  - solve master for new prices p and weights q_{it}
- robot i uses values $\sum q_{it} v_{it}(s)$ for lookahead in basic auction

Problem dynamics



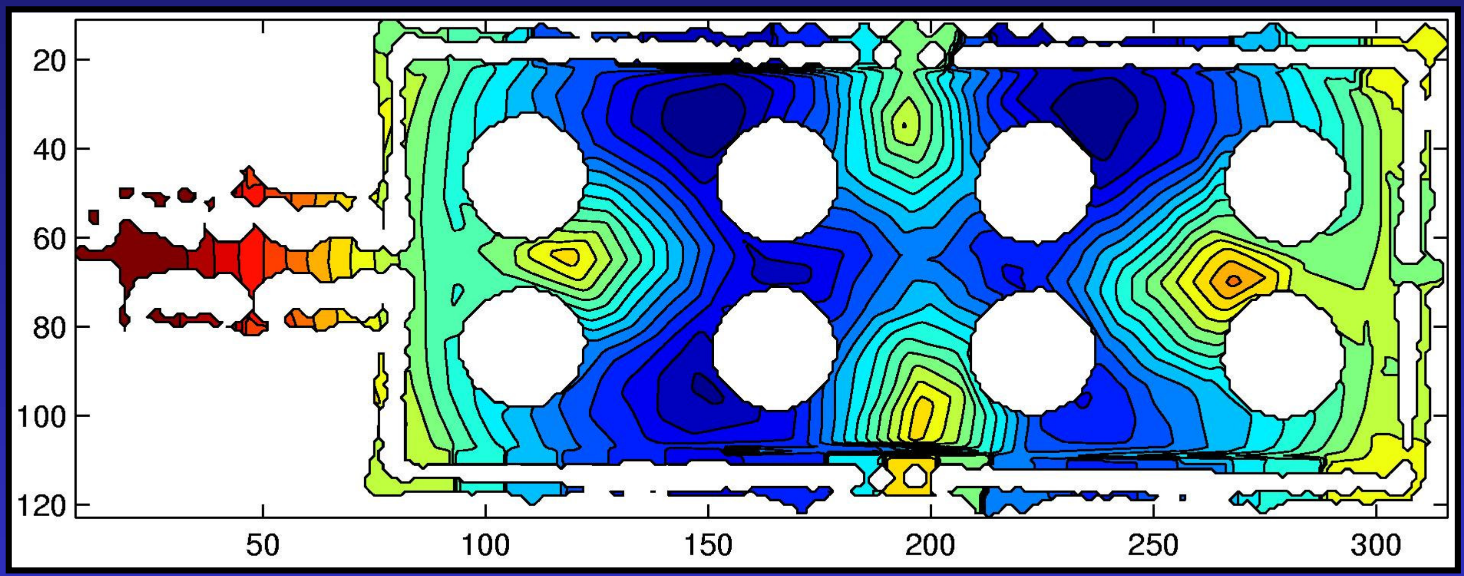
- State: position (80cm grid), destination (~150 points)
- Actions: move, buy goal, declare satisfied
- Reward: -1.25 per meter, +10 per goal, $(price) for buying goal

Constraints and prices

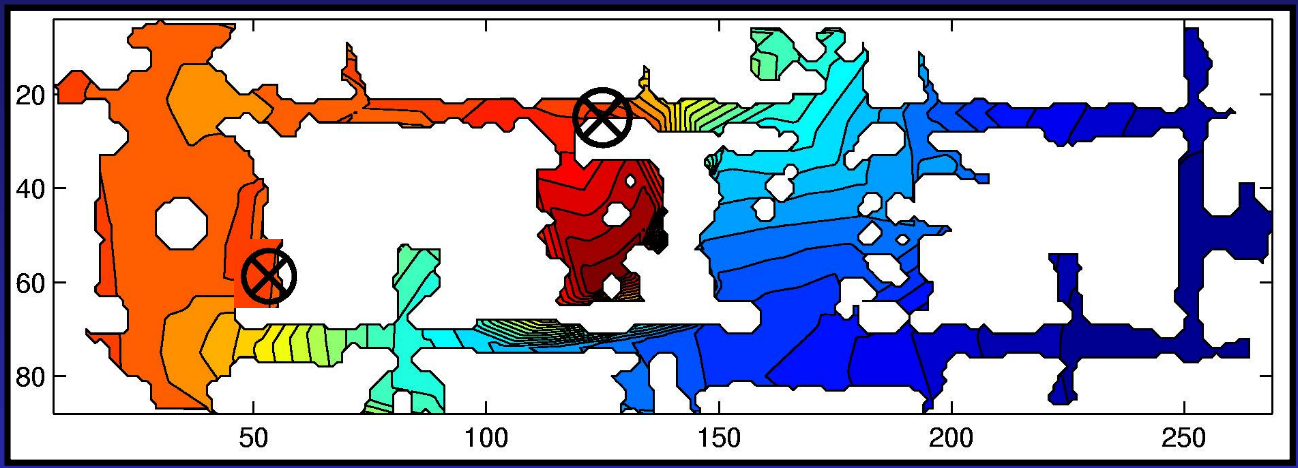


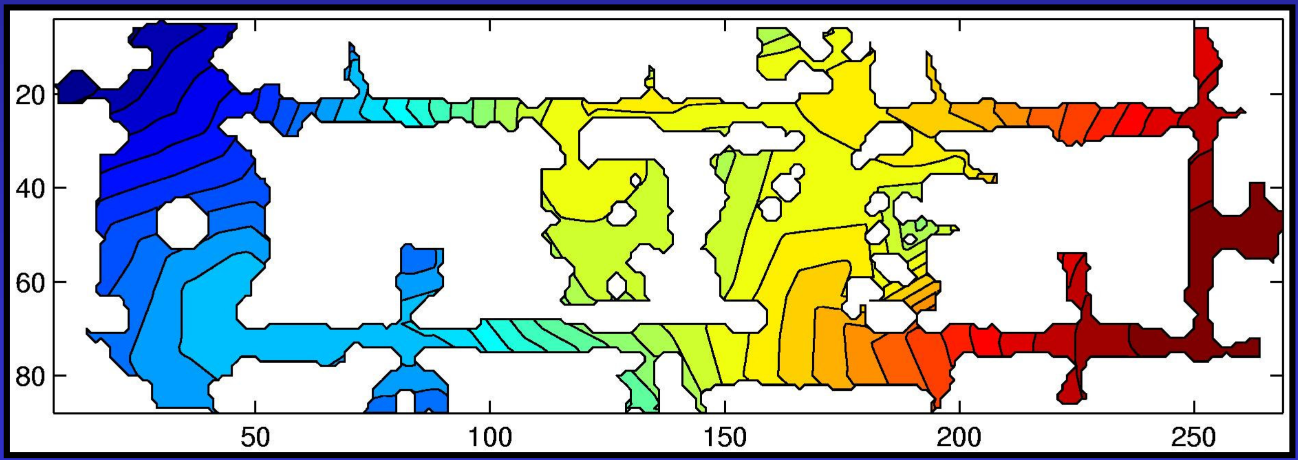
- Goals arrive at rate *g(x)*
- Frequency of *satisfy-goal(x)* must match *g(x)*
- So, learn one price per goal
- Individual planners think goal *x* is worth $10-*price(x)*

Learned value function

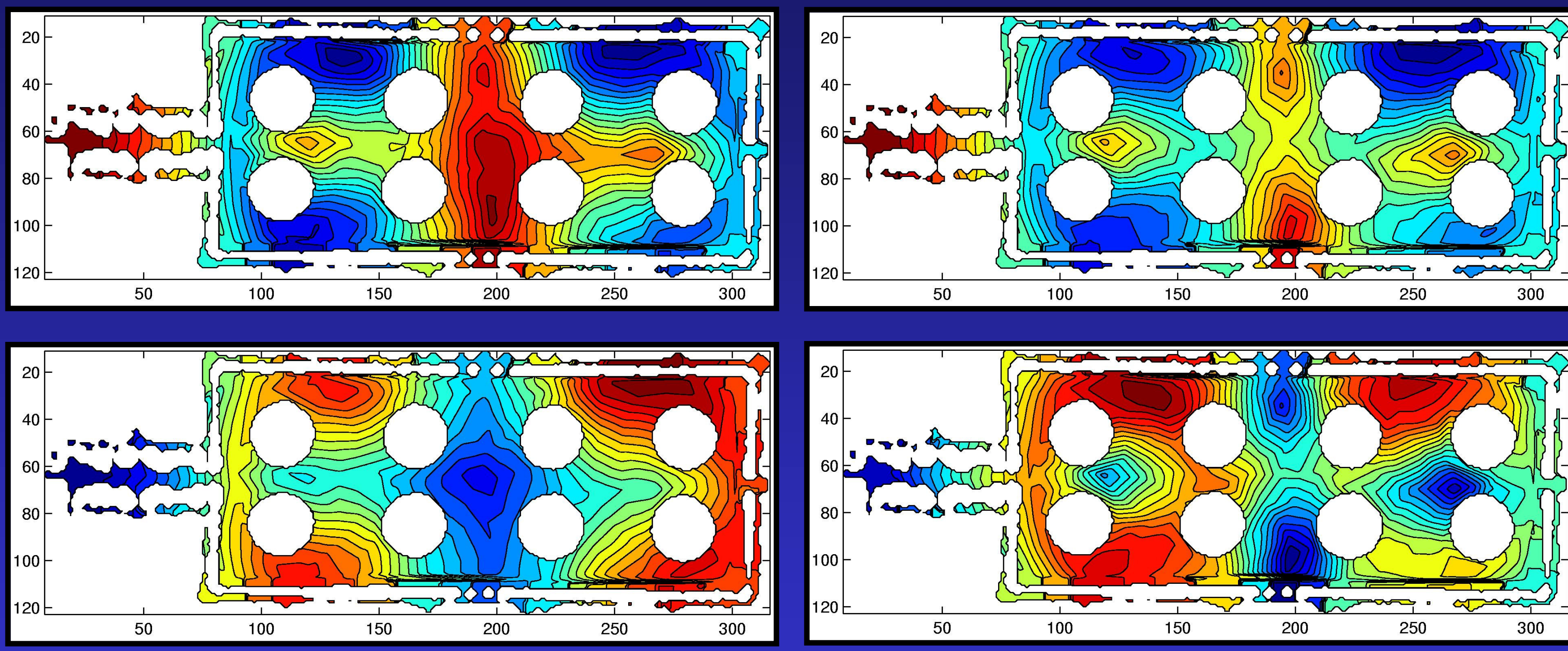


Roles





Roles



Market-based planning

- Approximate robot interactions by defining resources
- Estimate resource prices $\Rightarrow$ decouple planning problems
- Robots learn roles, effect of future uncertainty

[Guestrin & Gordon, UAI 02]
[Bererton, Gordon, Thrun & Khosla, NIPS 03]

Conclusion

- Solve multi-robot planning problems by factorizing and approximating
- Belief compression
- Market-based planning