Learning to behave intelligently

or

How to stop worrying and love being wrong

Leslie Pack Kaelbling
MIT CSAIL

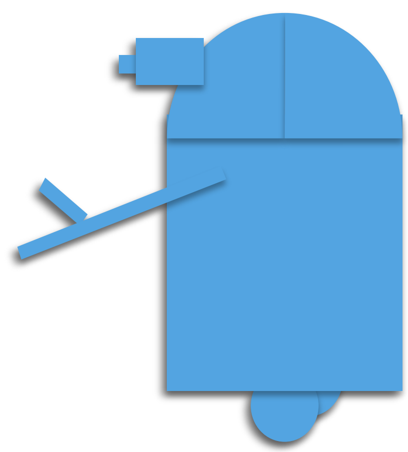
Flexible intelligence in complex domains

[1] What To do First?

Our problem

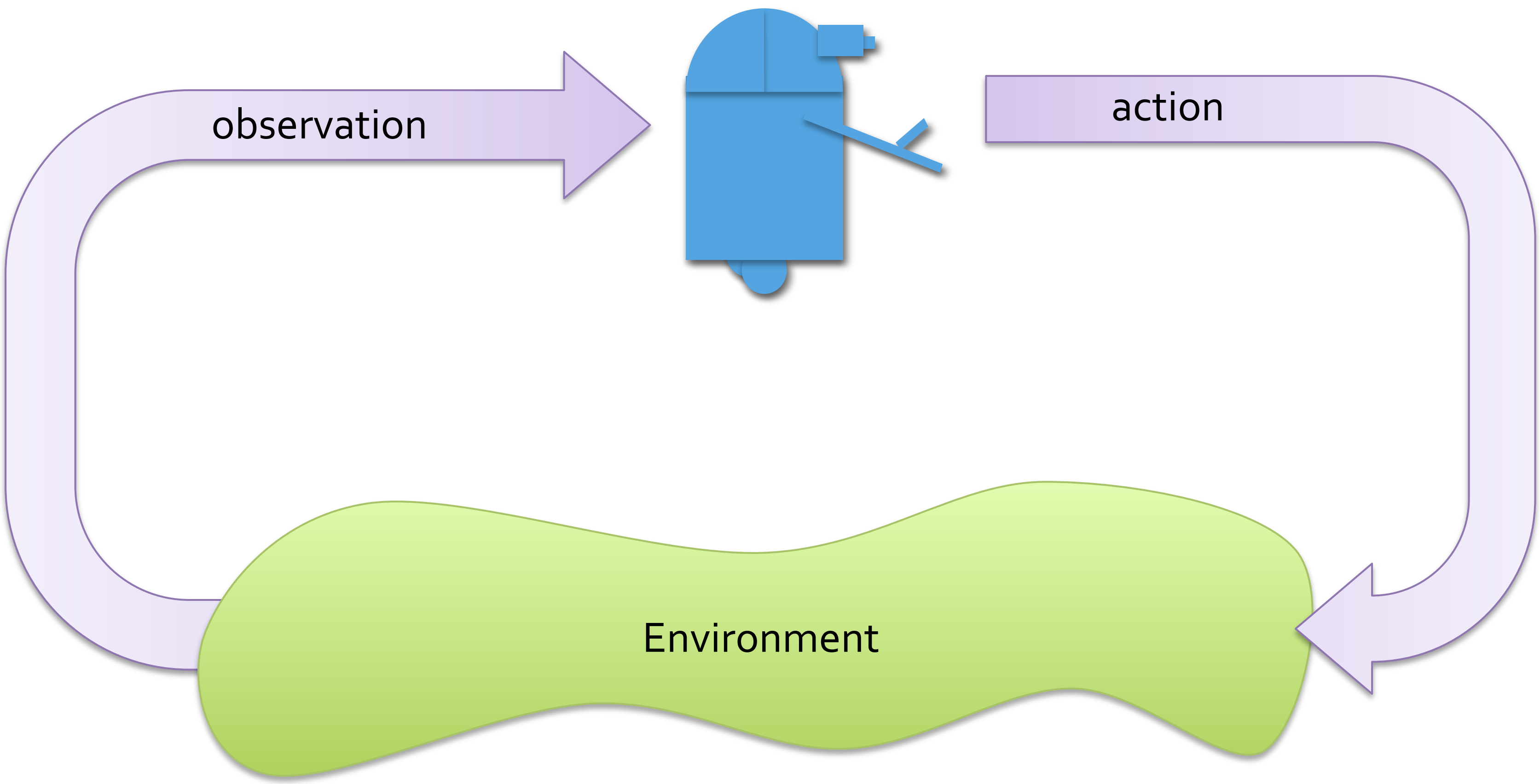
How to build the **'central' computational mechanisms** for

- **closed-loop control** of a system with
- **sensors and actuators** that has
- **long-term goal-directed** interactions with
- a **complex**
- **imperfectly predictable** external environment

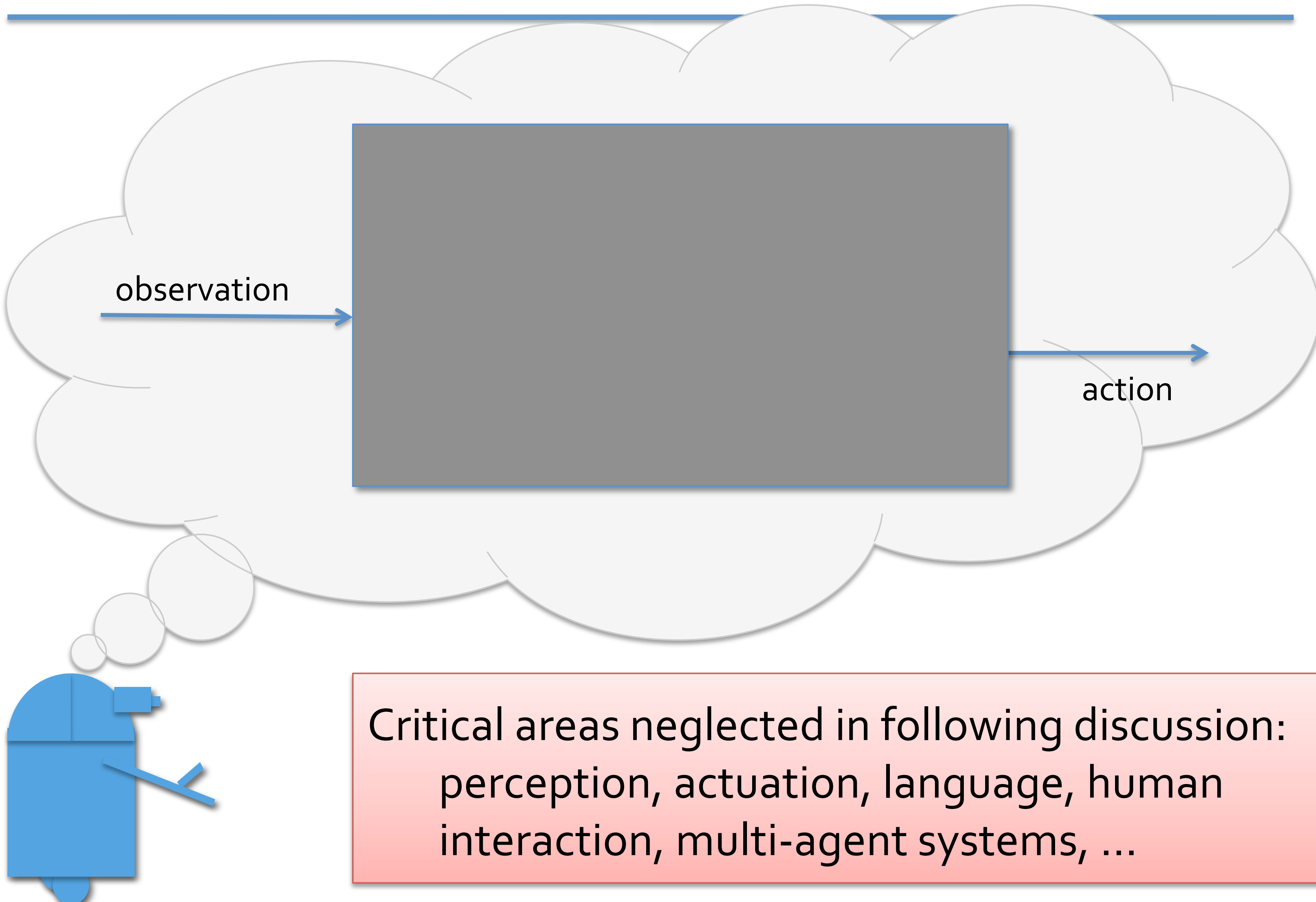


What is the role of learning?

Interaction with an external environment



What to learn? What to build in?



Critical areas neglected in following discussion:
perception, actuation, language, human interaction, multi-agent systems, …

Structures we could learn

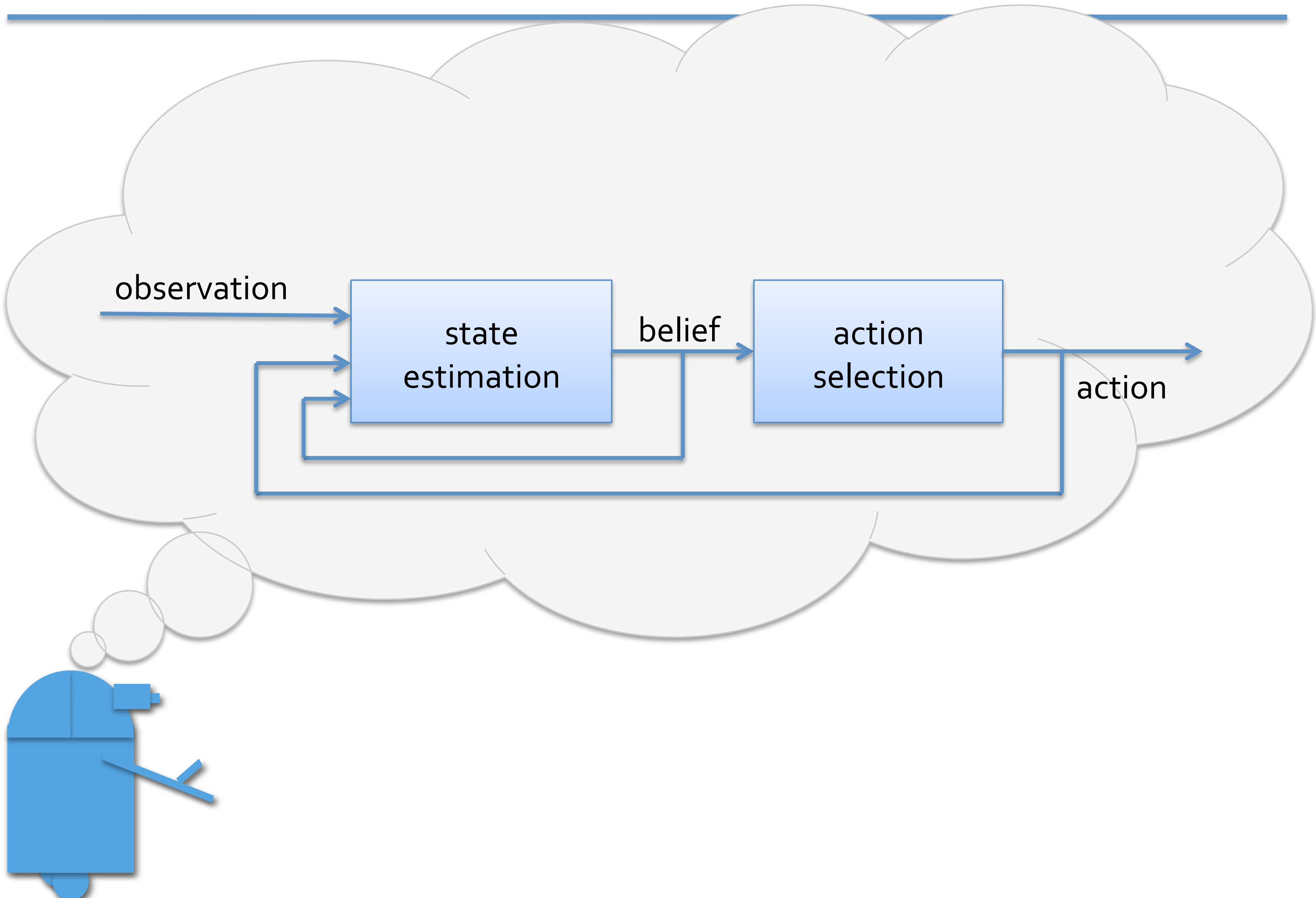
	Independent of R	Easy to factor	Easy to apply
Policy			✔
Value function			✔
Transition model	✔	✔	
Observation model	✔	✔	

Structures we could learn

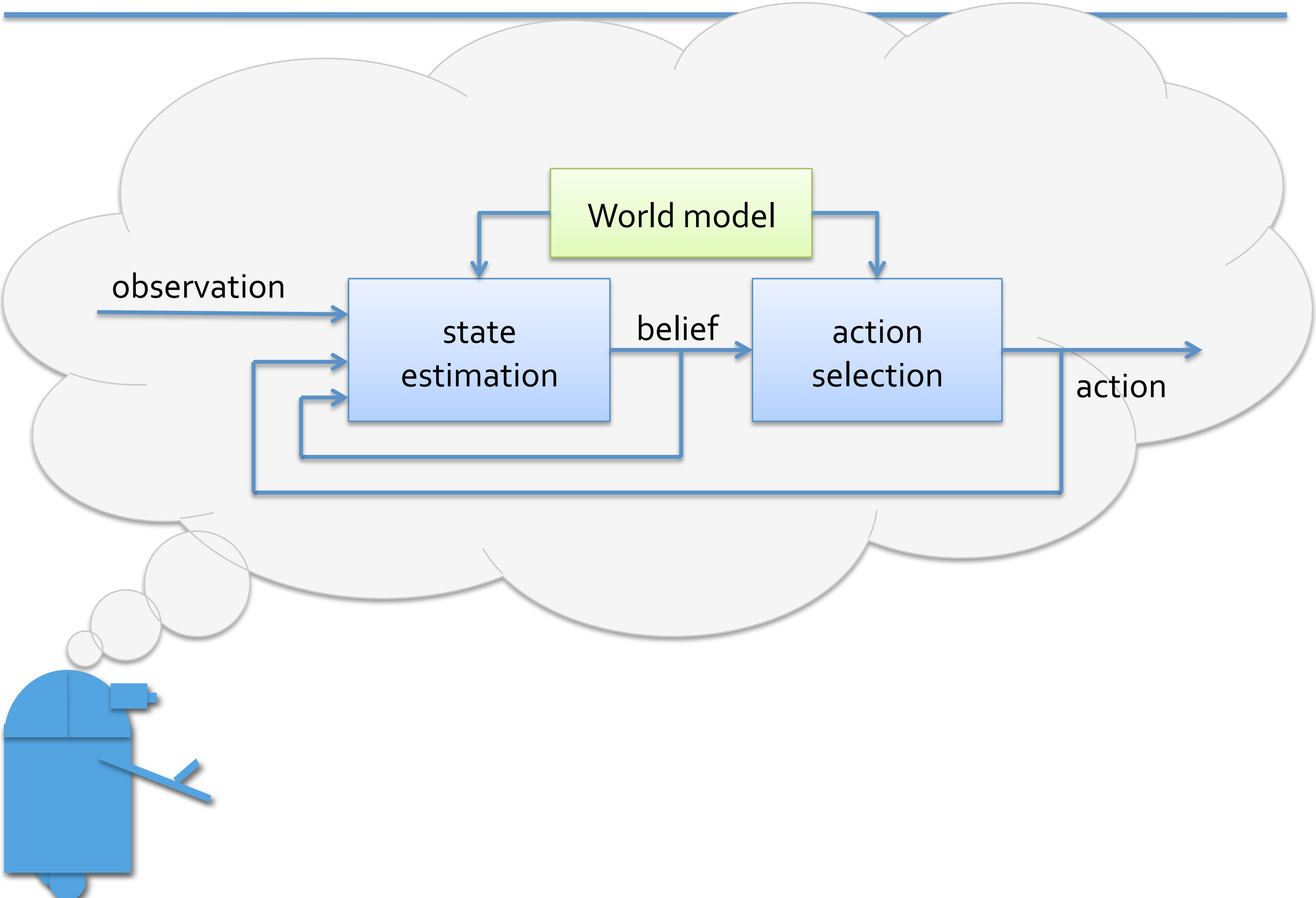
	Independent of R	Easy to factor	Easy to apply
Policy			✔
Value function			✔
Transition model	✔	✔	
Observation model	✔	✔	

Let's try to make transition and observation models
easy to apply and easy to learn

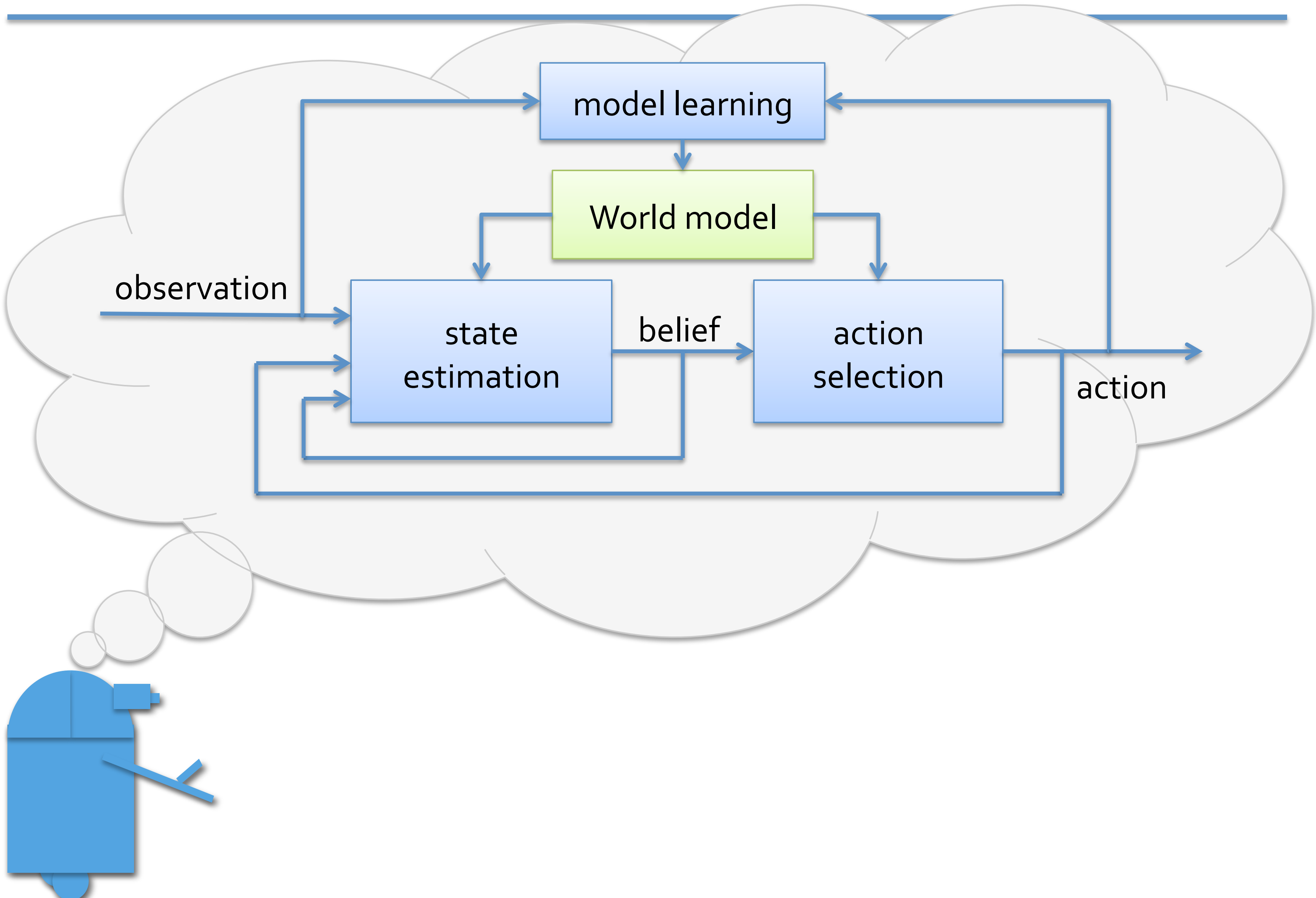
Internal architecture



Internal architecture



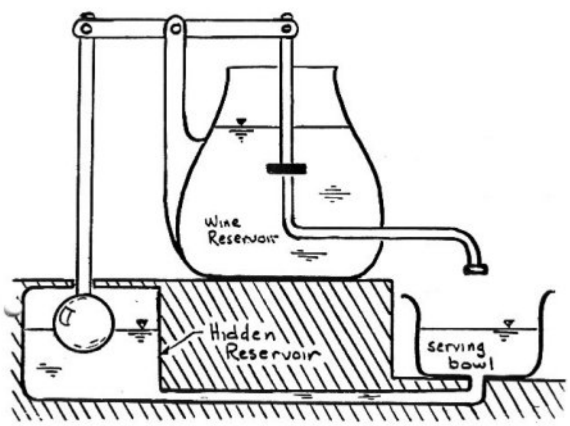
Internal architecture



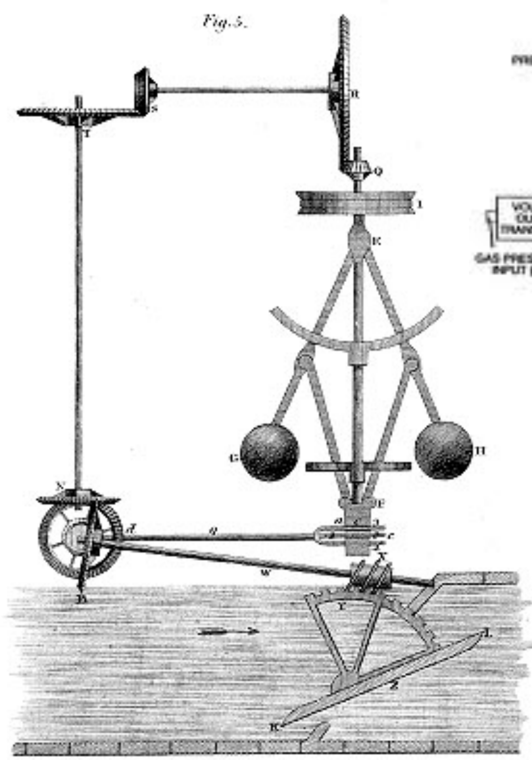
Feedback control

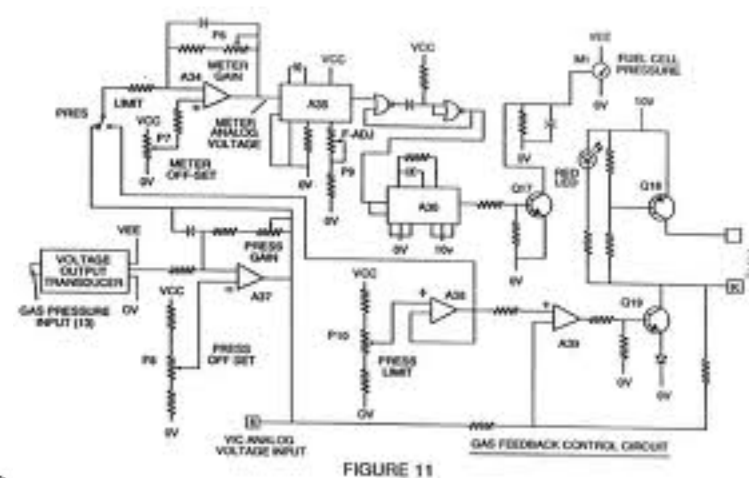
Loop:

- select an action based on (estimated) world state
- see what effect it has in the world

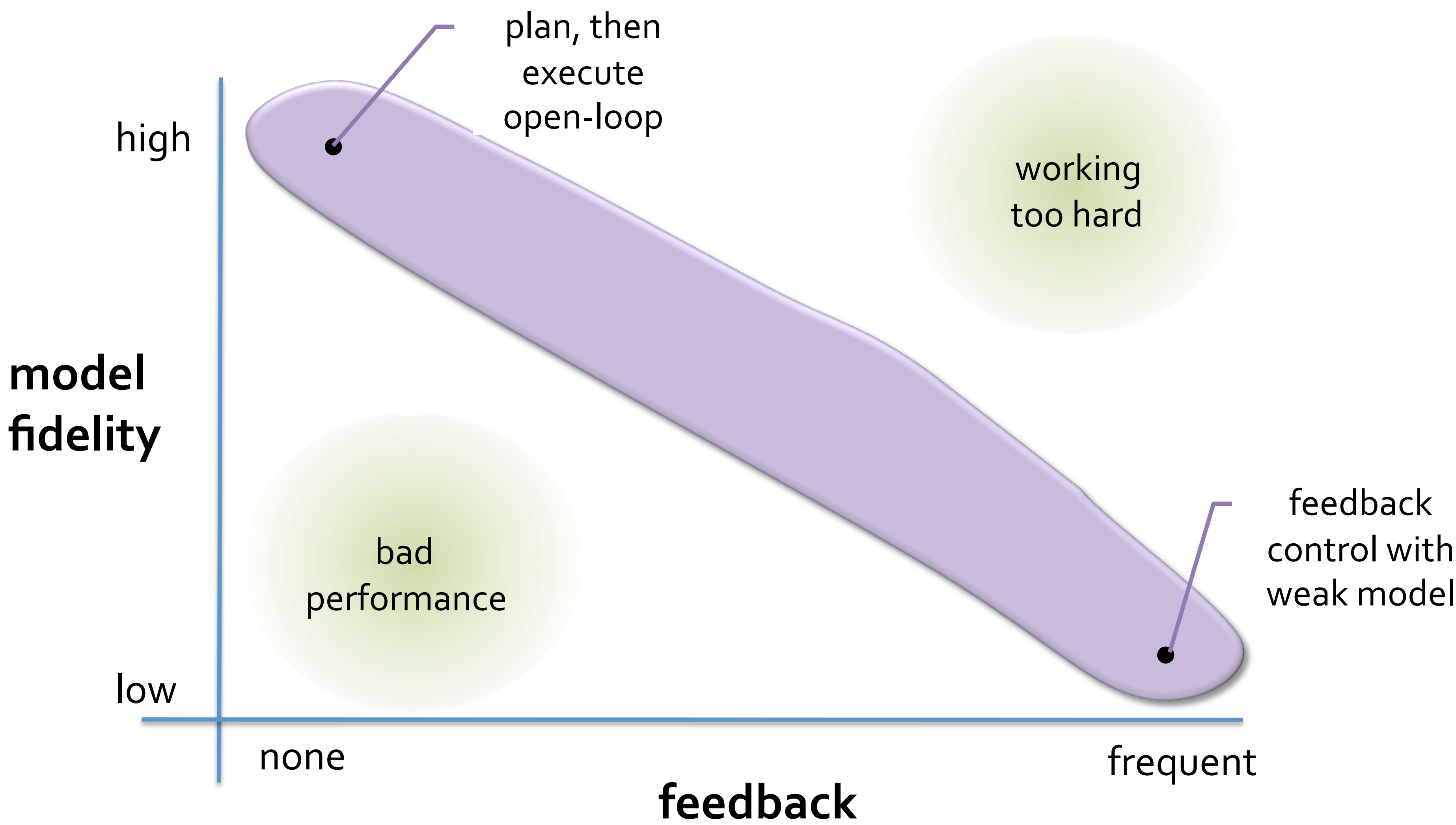


HERO'S SELF-LEVELING BOWL
ca. 30 B.C.





All models are wrong; but some are useful. - Box



Plans versus policies

Number of states in an interesting environment:

- maybe continuous
- maybe 10^4 predicates applied to
  - one or more objects drawn from
    - 10^5 objects (books, shoes, cans of tomatoes)
    - or 10^8 objects (pieces of pasta, pages of books)

Number of states true right now: 1

Using simplified models for action selection

ICAPS 2004: First probabilistic planning competition

Entries: Many sophisticated MDP planning algorithms

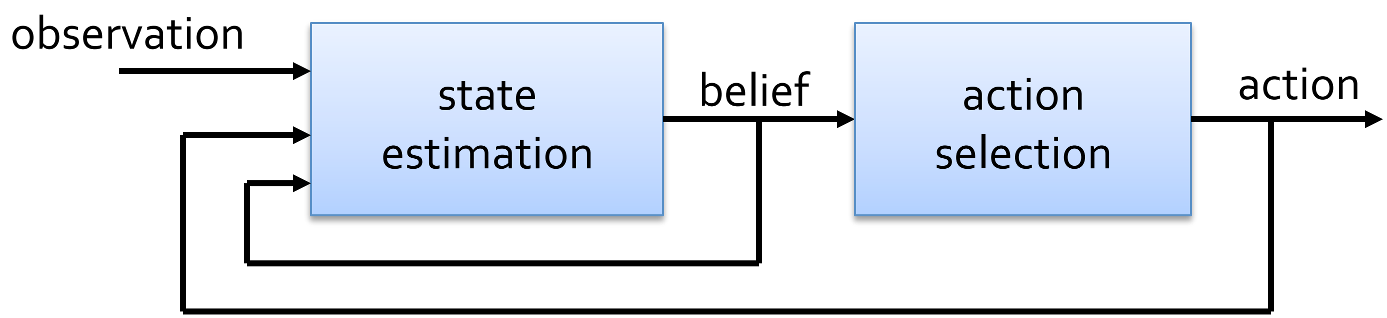
Winner: FF-Replan

- determinimized model + classical forward planner
- replan on unexpected outcomes

Result: New definition of probabilistically interesting problems

- can't be solved effectively by FF-Replan

Action selection with partial observability



Plan in belief space:

- every action gains information and changes the world
- changes are reflected in new belief via estimation
- goal is to believe that the environment is in a desired state

Using simplified models for action selection

Three examples:

Continuous control with state-dependent observation noise:

- deterministic dynamics
- most likely observation

Robot grasping with tactile sensing:

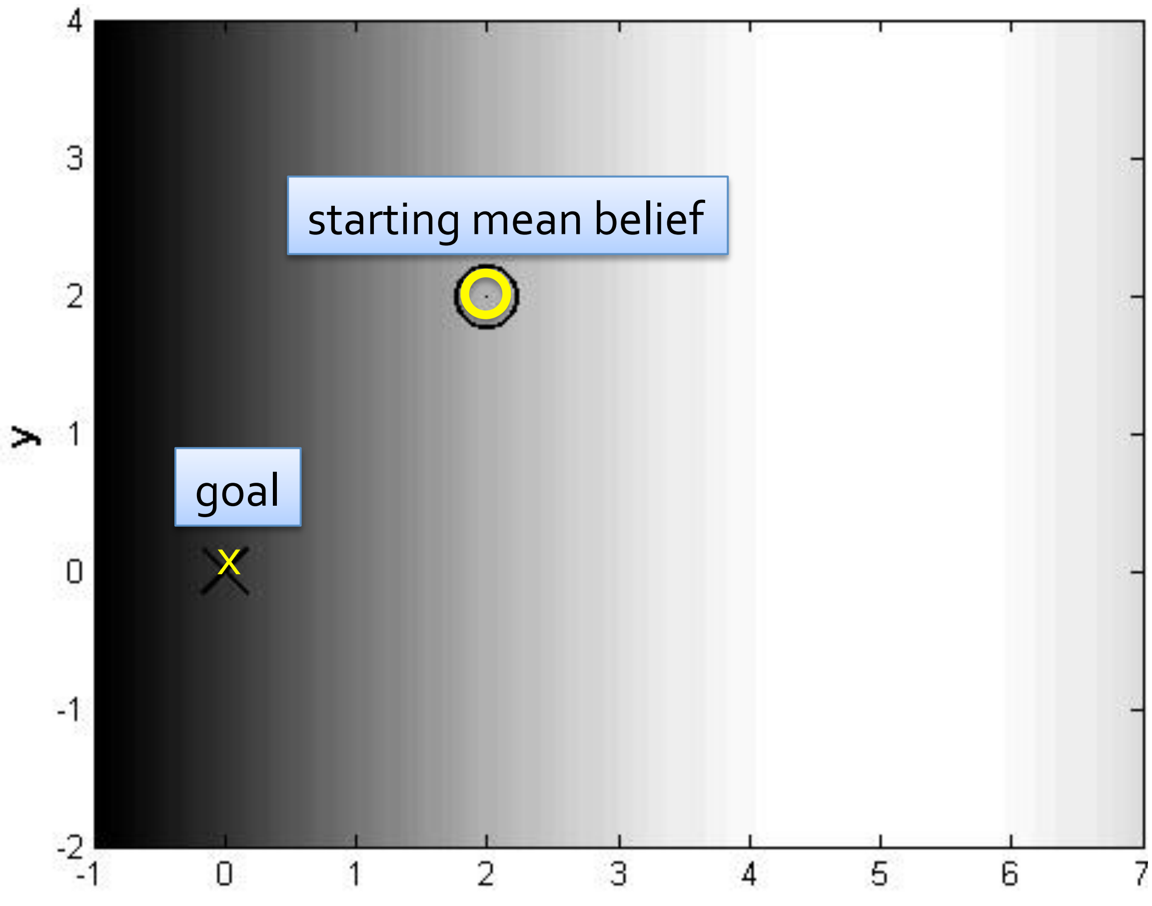
- shortened horizon
- reduced action space

Household robot with local sensing:

- assume subtask serializability
- assume desired observations

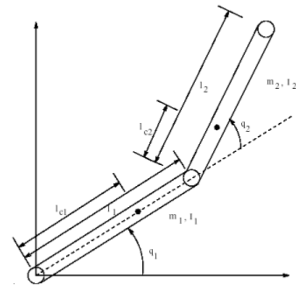
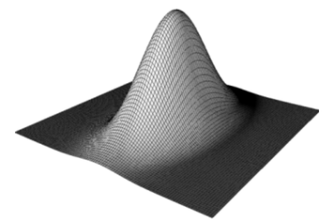
State-dependent observation noise

- robot in x, y space
- good position sensing in light regions; poor in dark



Joint work with Rob Platt, Russ Tedrake and Tomás Lozano-Pérez

Control in belief space: underactuated

	Acrobot	Gaussian belief:
		
State space:	$x = \begin{pmatrix} \theta \\ \dot{\theta} \end{pmatrix}$	$b = \begin{pmatrix} m \\ \Sigma \end{pmatrix}$
Planning objective:	$x_g = \begin{pmatrix} \pi \\ 0 \end{pmatrix}$	$b_g = \begin{pmatrix} x_g \\ 0 \end{pmatrix}$
Underactuated dynamics:	$\ddot{\theta} = f(\theta, \dot{\theta}, u)$	???

Belief space dynamics

Dynamics specify next belief state, as a function of previous belief state and action

- state update: generalized Kalman filter

$$(\mu_{t+1}, \Sigma_{t+1}) = \mathrm{GKF}(o_t, a_t, \mu_t, \Sigma_t)$$

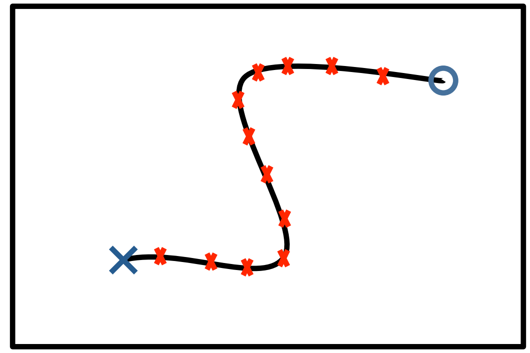
- substitute expected observation in for actual one add Gaussian noise

$$\begin{aligned}(\mu_{t+1}, \Sigma_{t+1}) &= F(a_t, \mu_t, \Sigma_t) + N \\ &= \mathrm{GKF}(\bar{o}(\mu_t), a_t, \mu_t, \Sigma_t) + N\end{aligned}$$

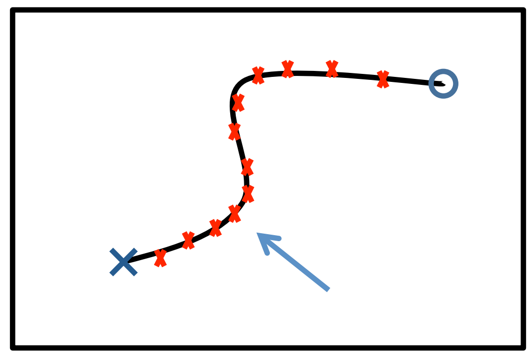
- continuous Gaussian non-linear dynamics: apply tools from control theory

Planning by local optimization

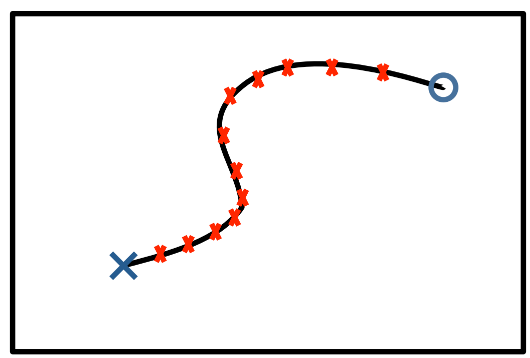
1. Parameterize initial trajectory by "via" points



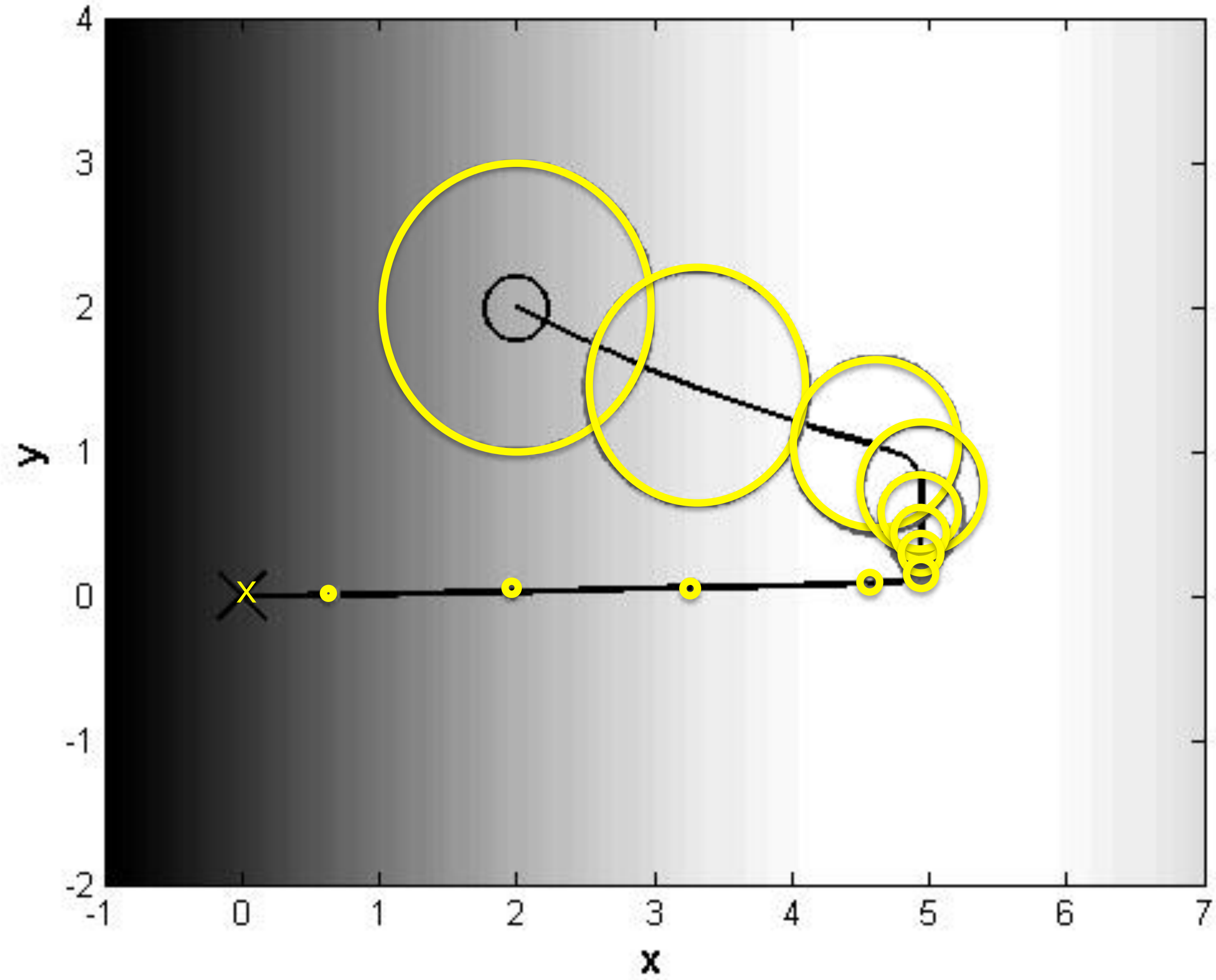
2. Shift "via" points while enforcing dynamic constraints



3. Stop when local minimum is reached

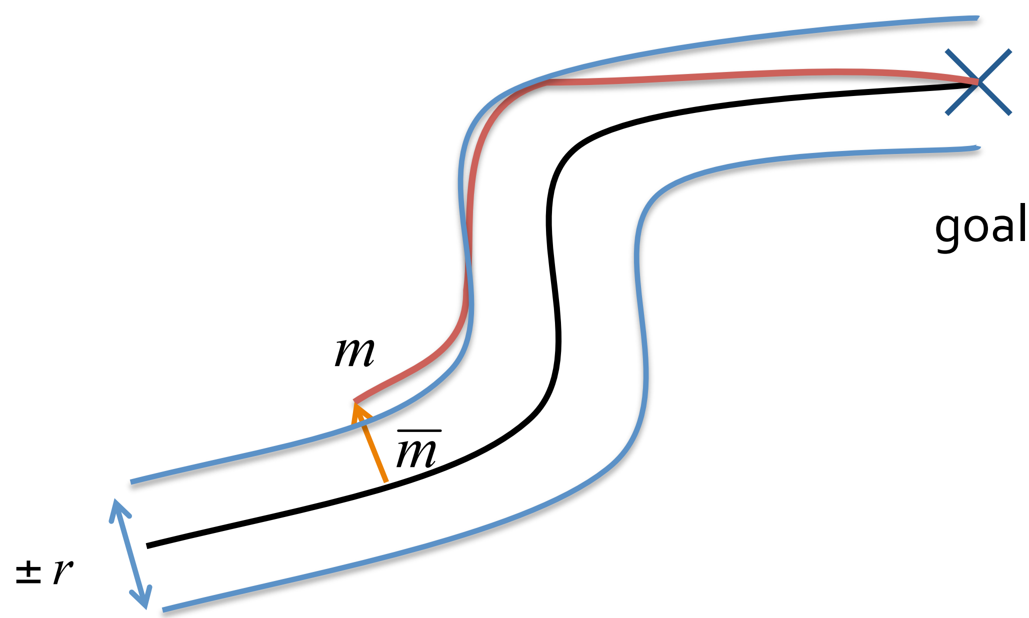


Light-dark plan

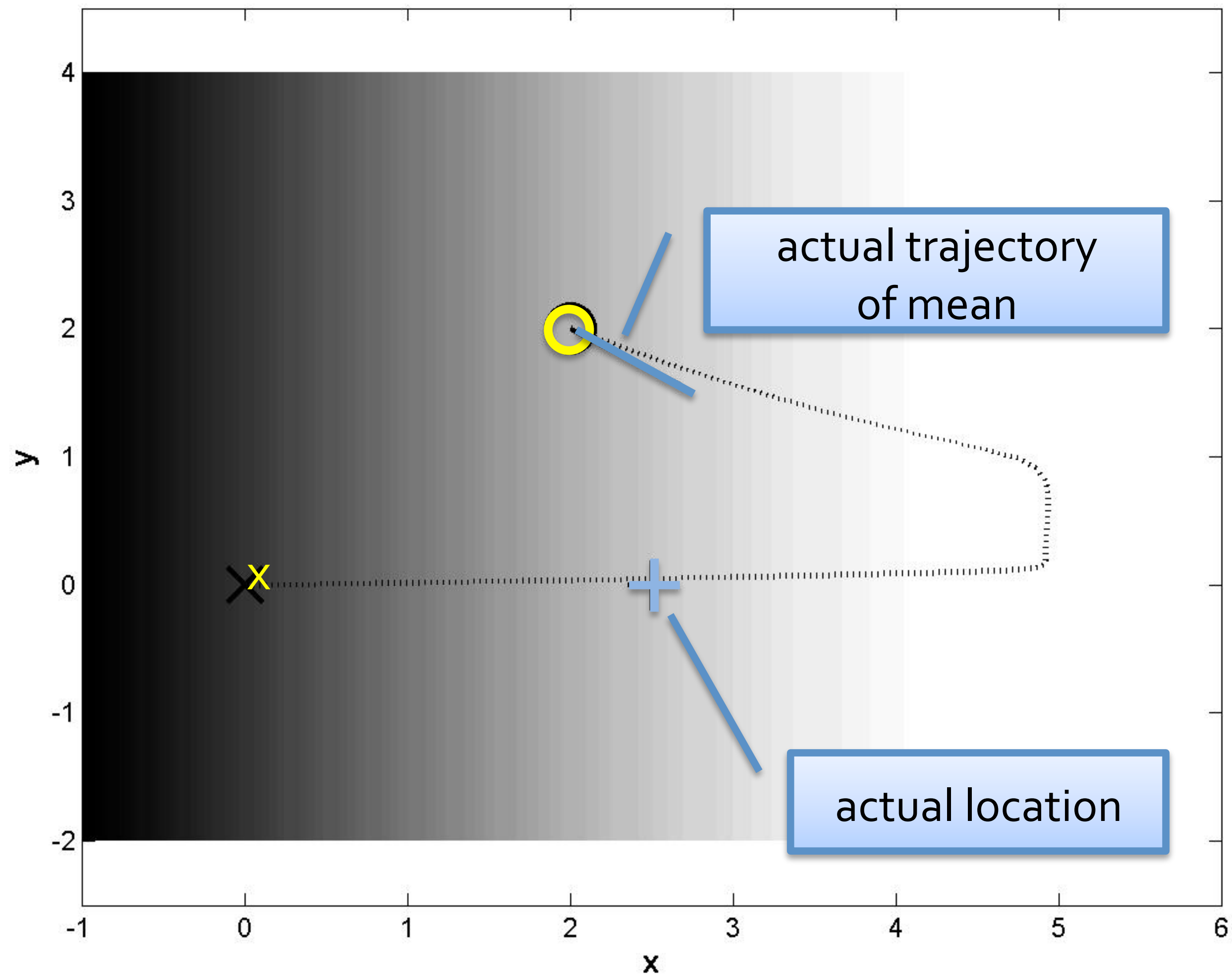


Replanning

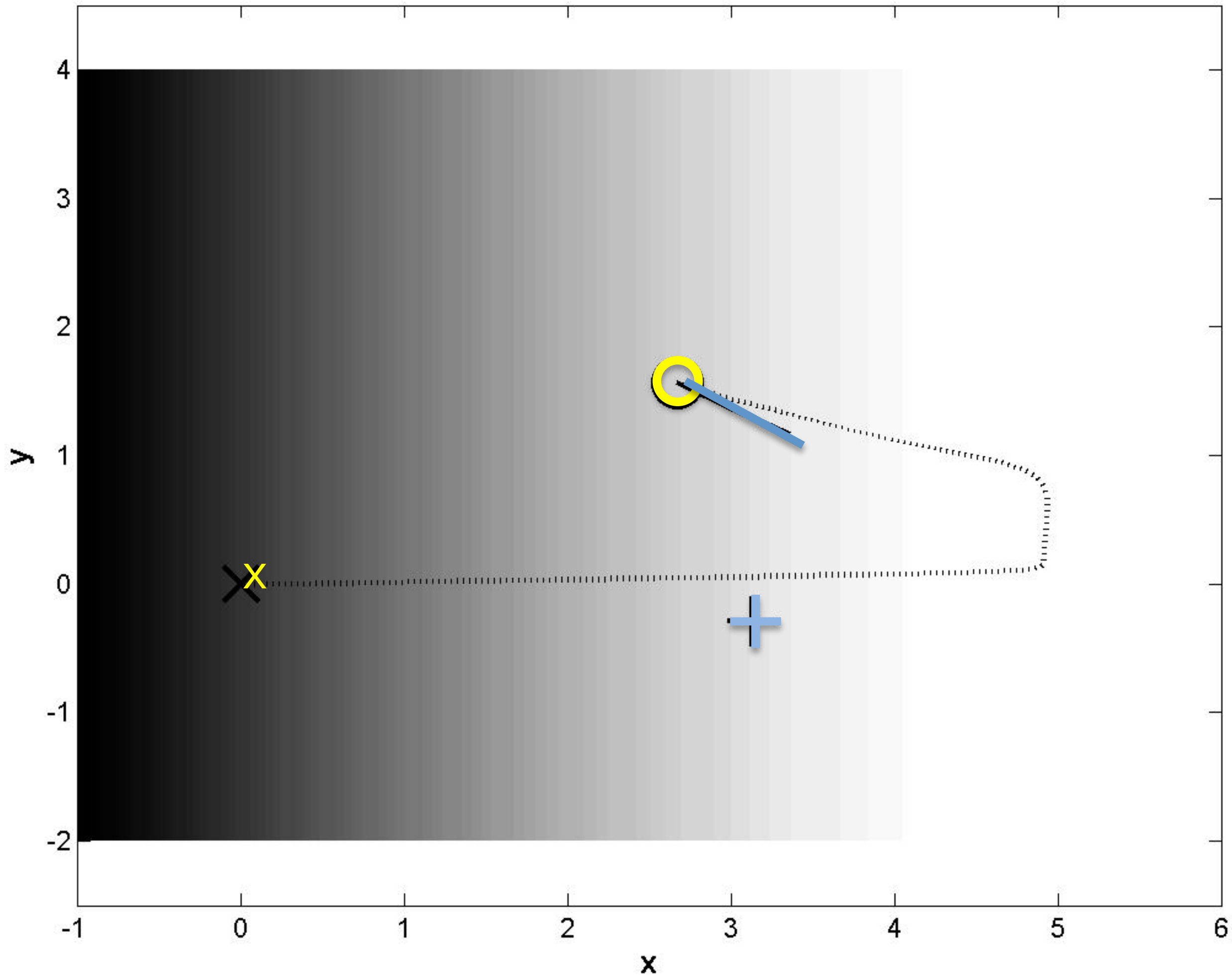
Replan when new belief state deviates too far from planned trajectory



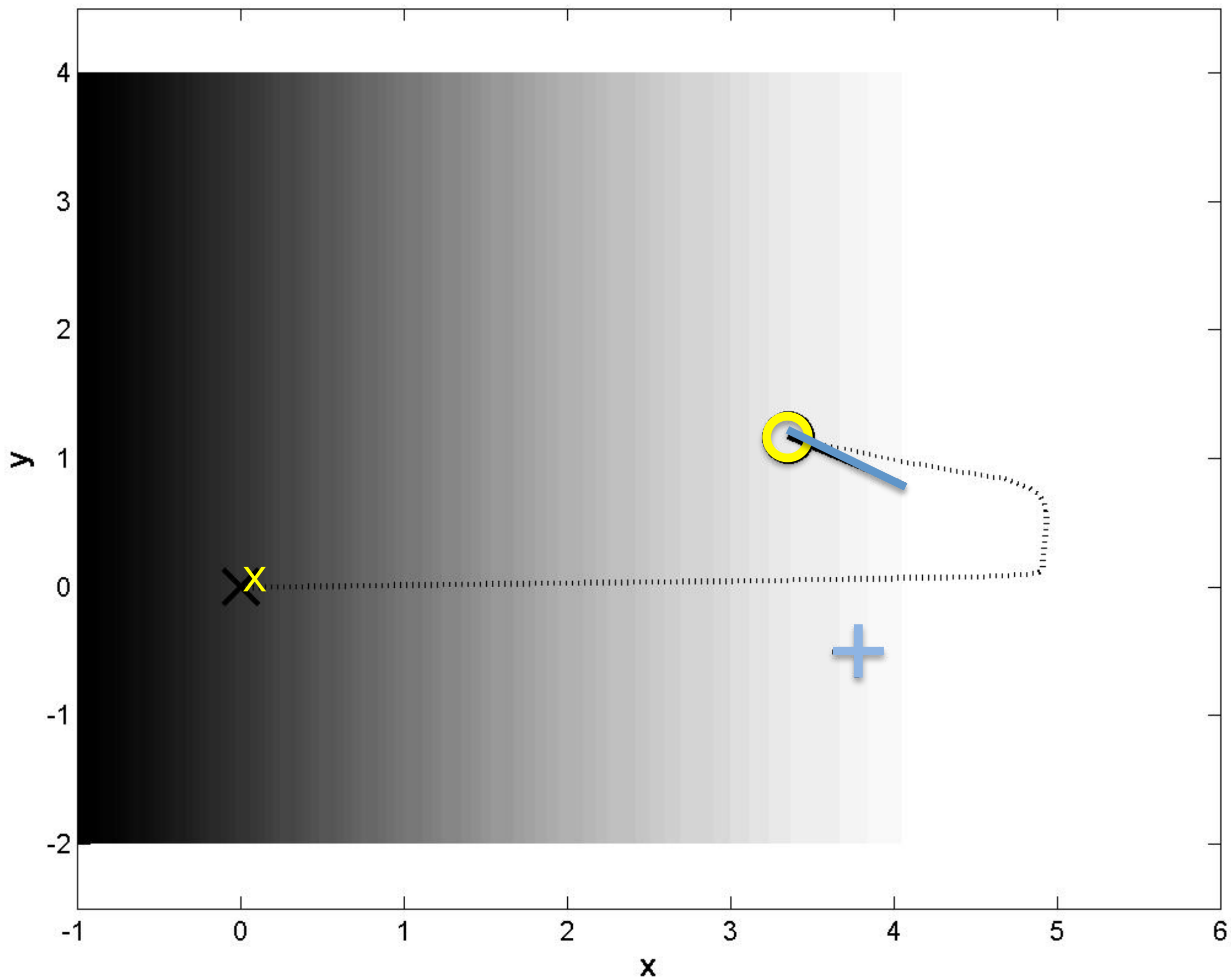
Replanning: light-dark problem



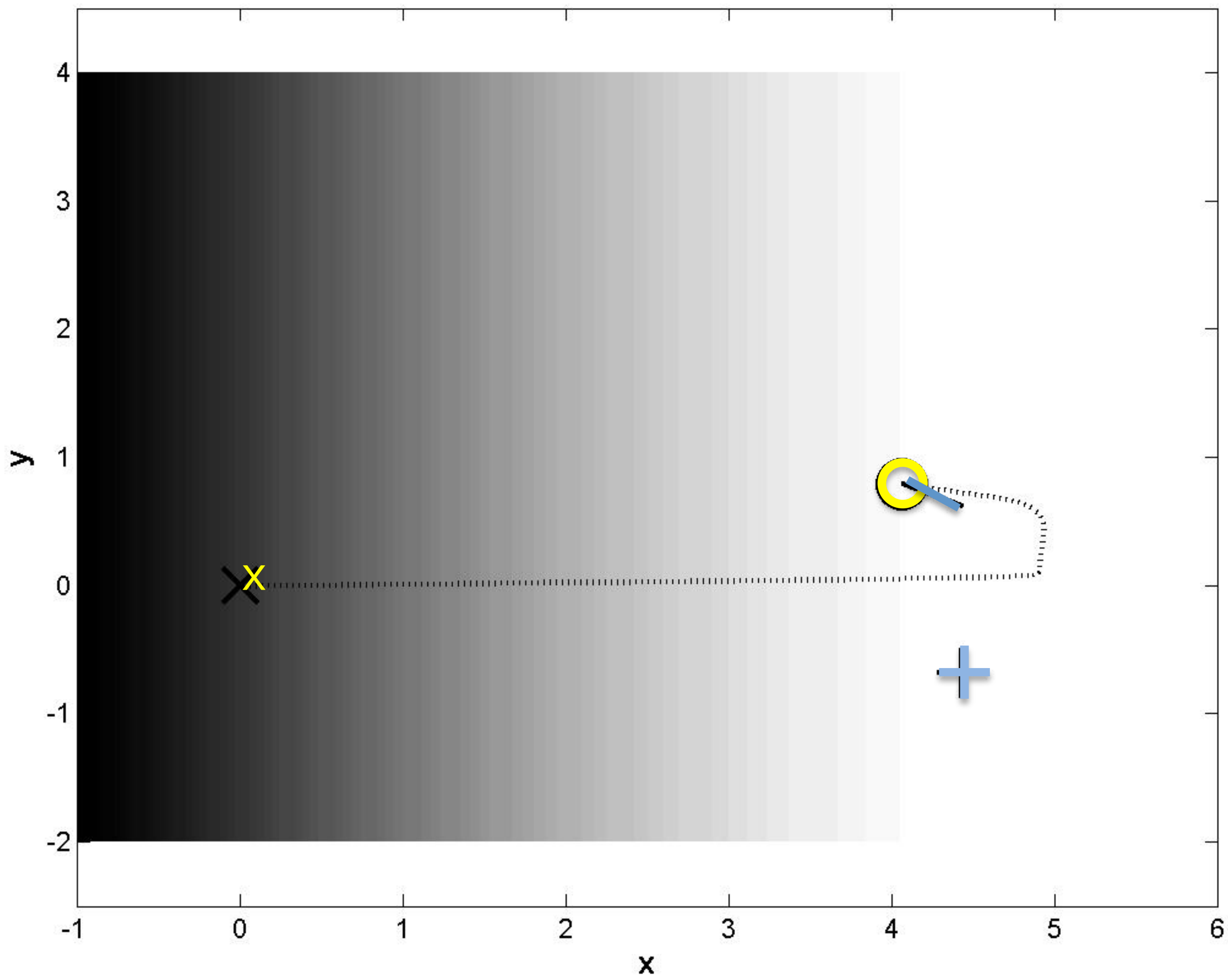
Replanning: light-dark problem



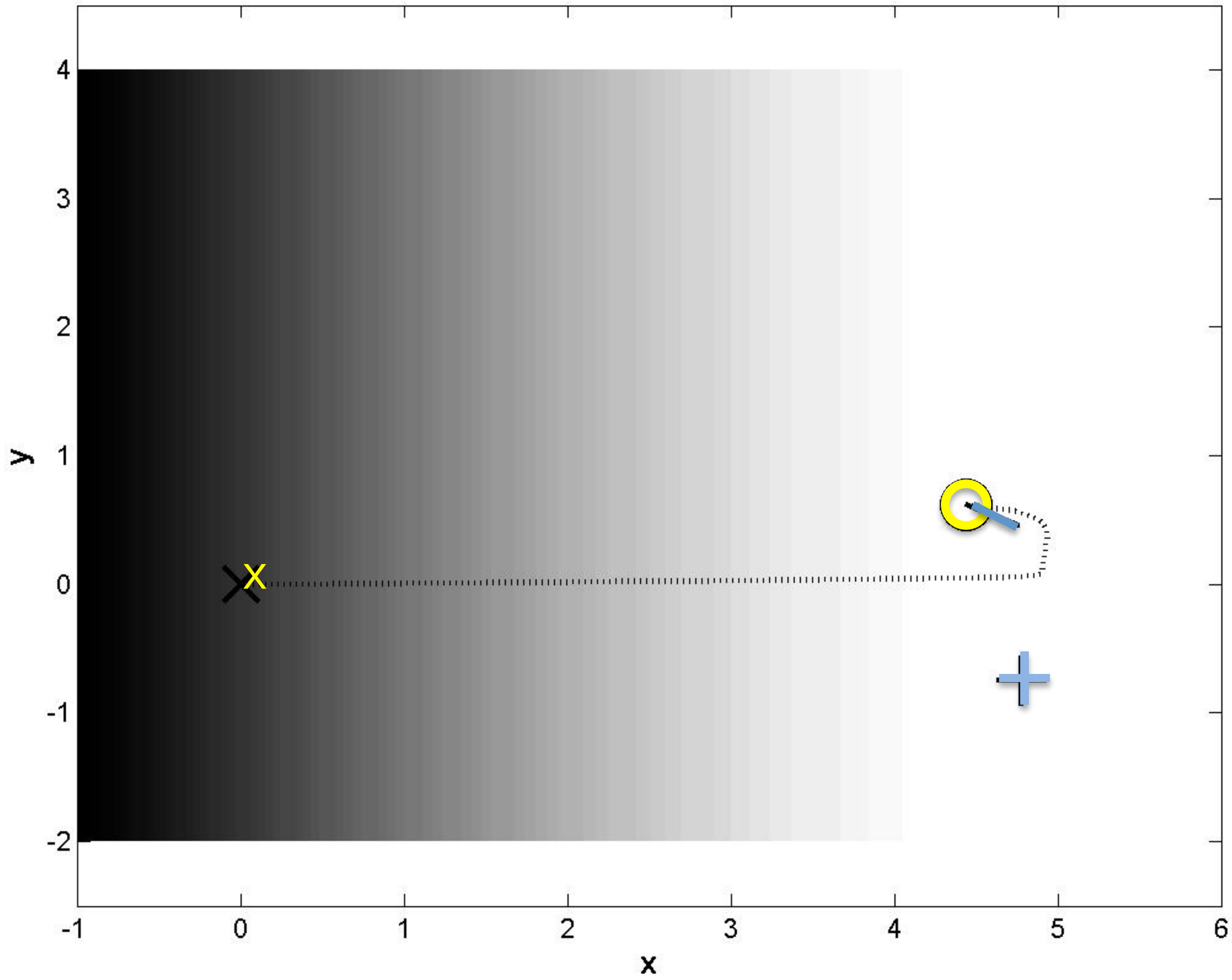
Replanning: light-dark problem



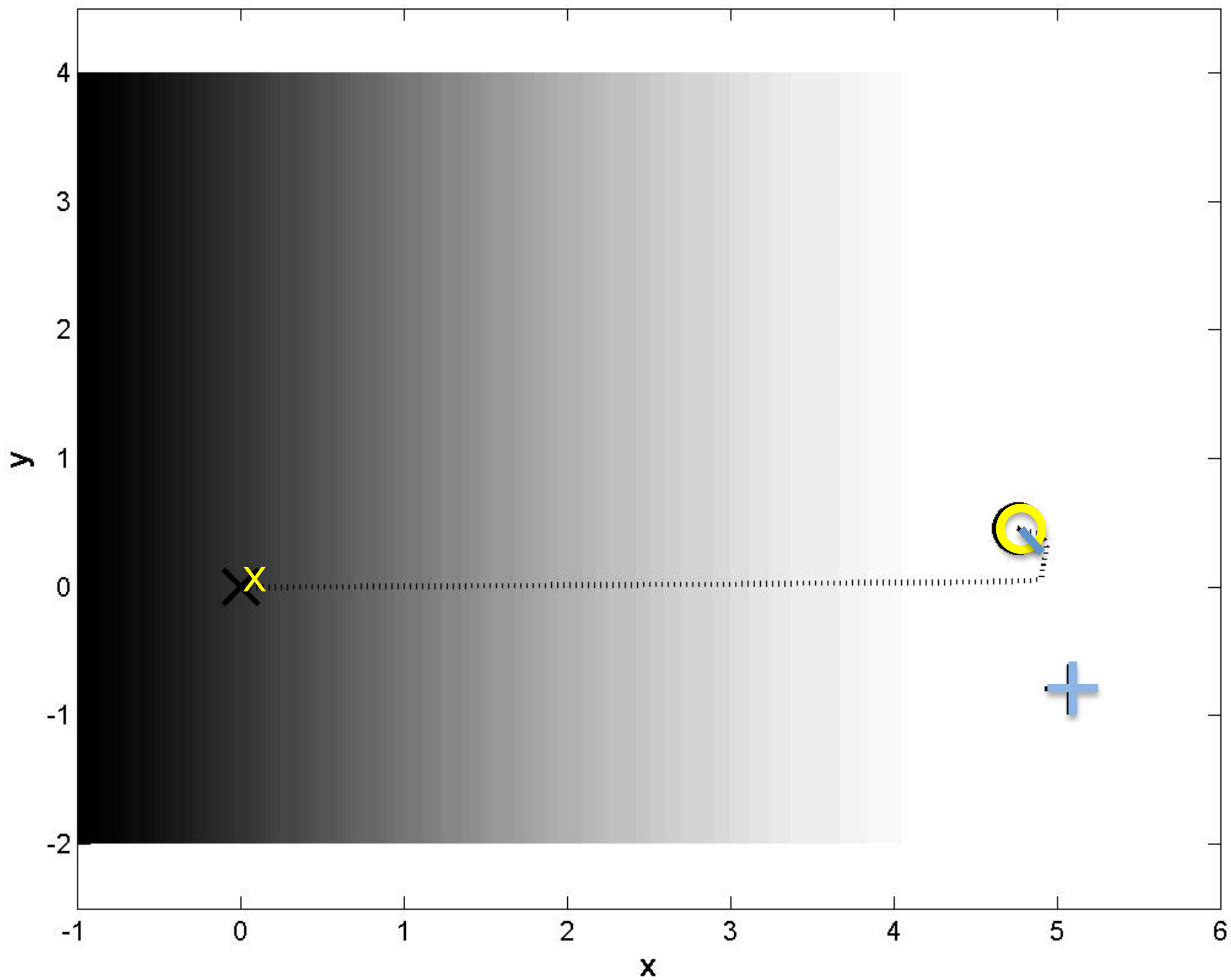
Replanning: light-dark problem



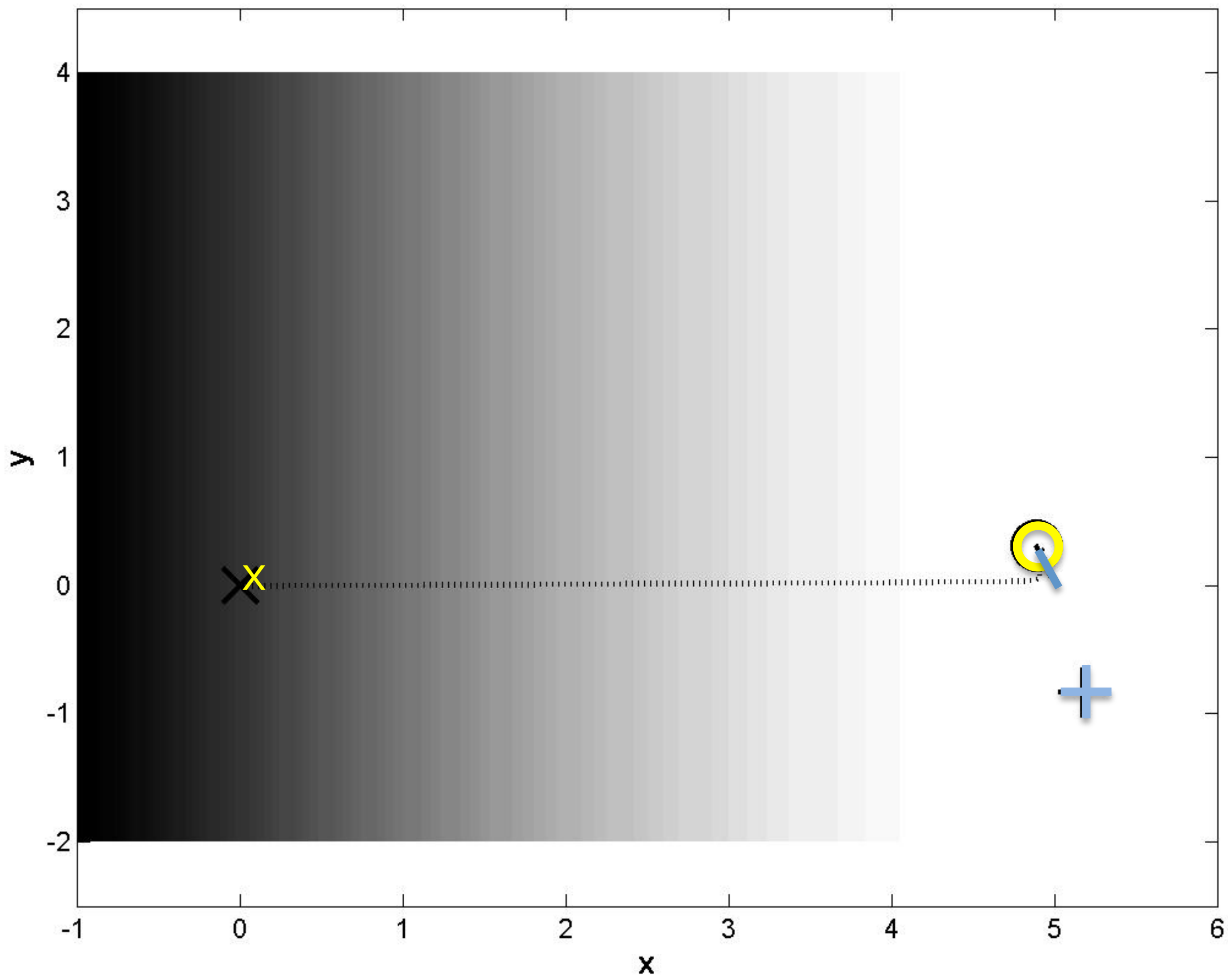
Replanning: light-dark problem



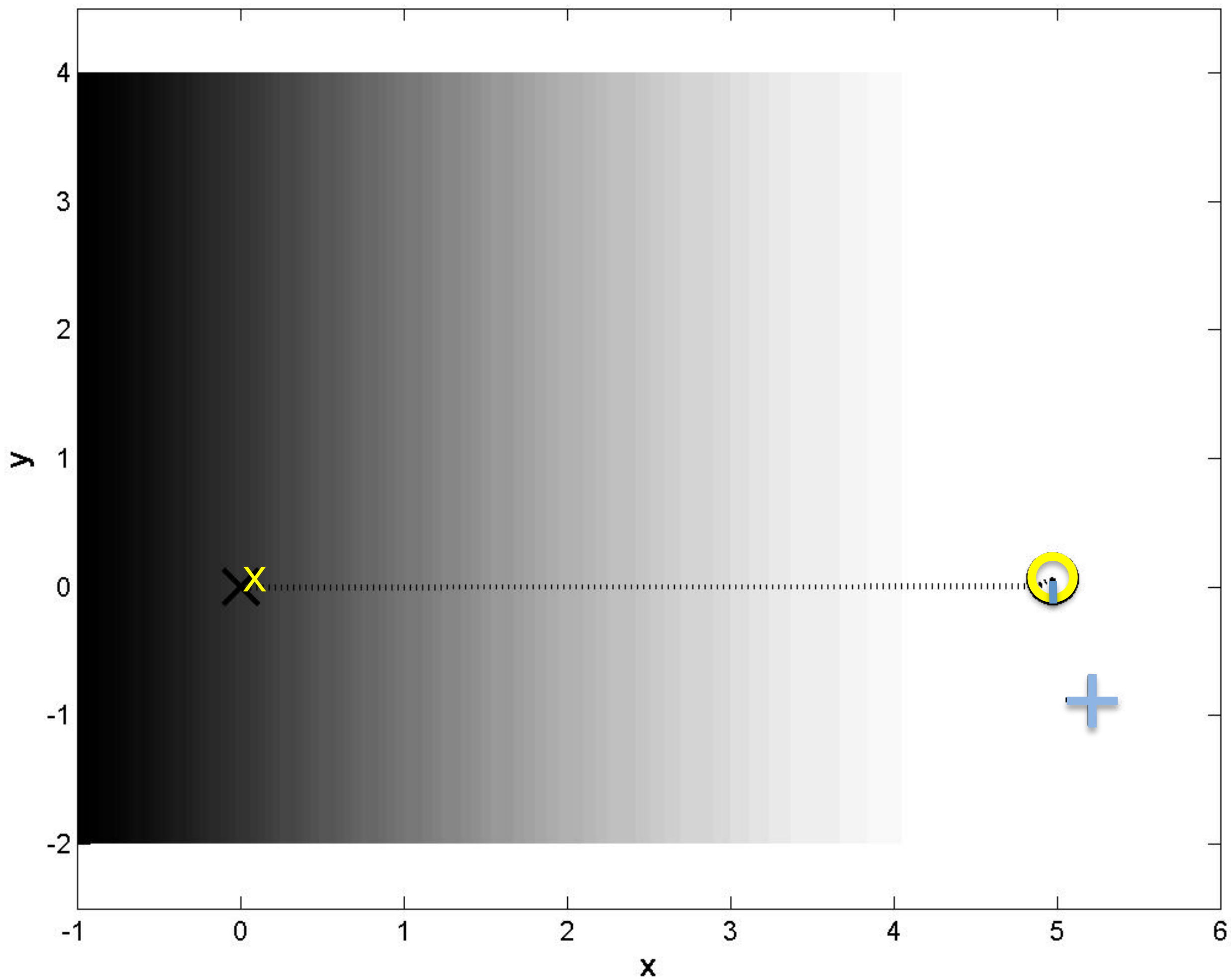
Replanning: light-dark problem



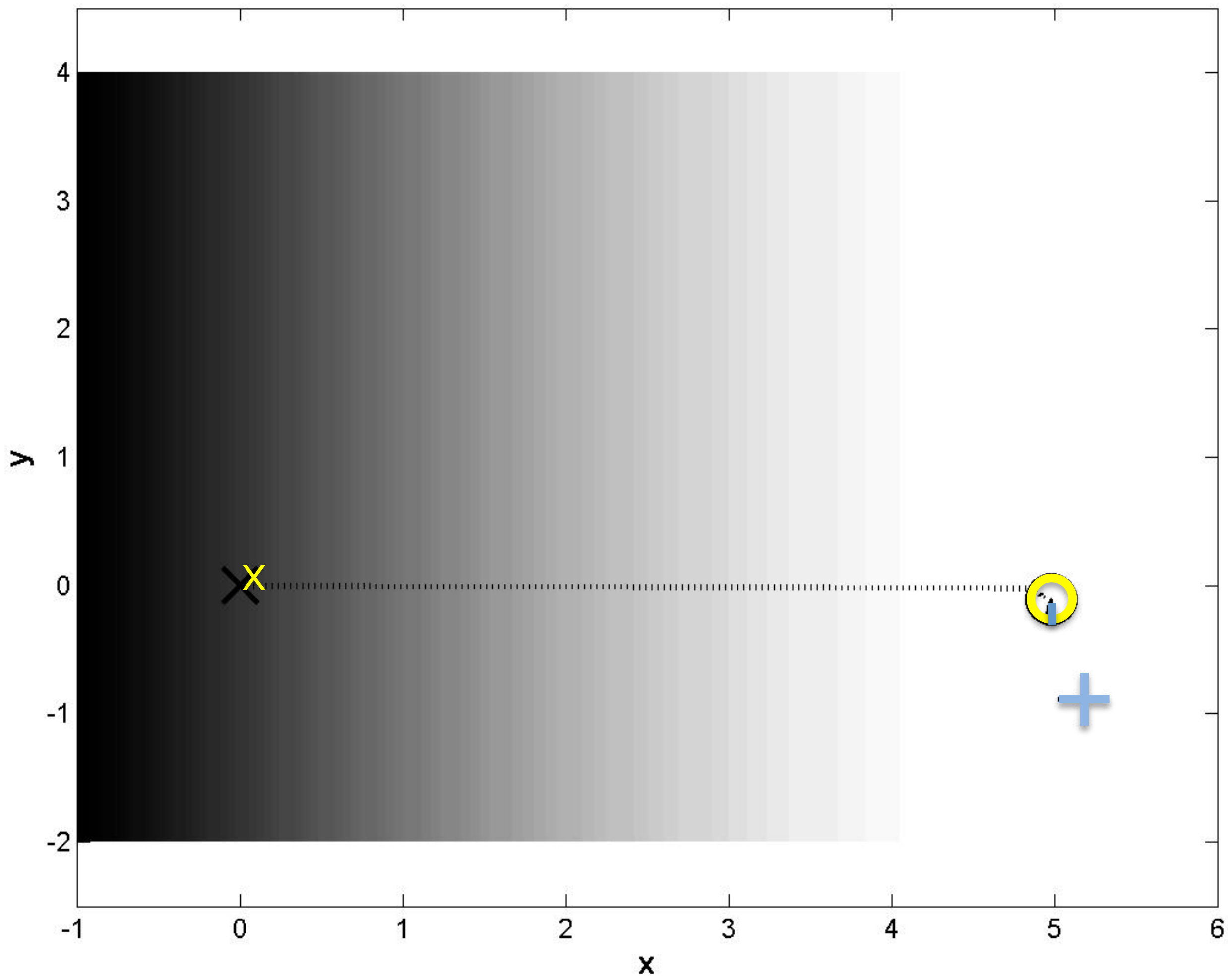
Replanning: light-dark problem



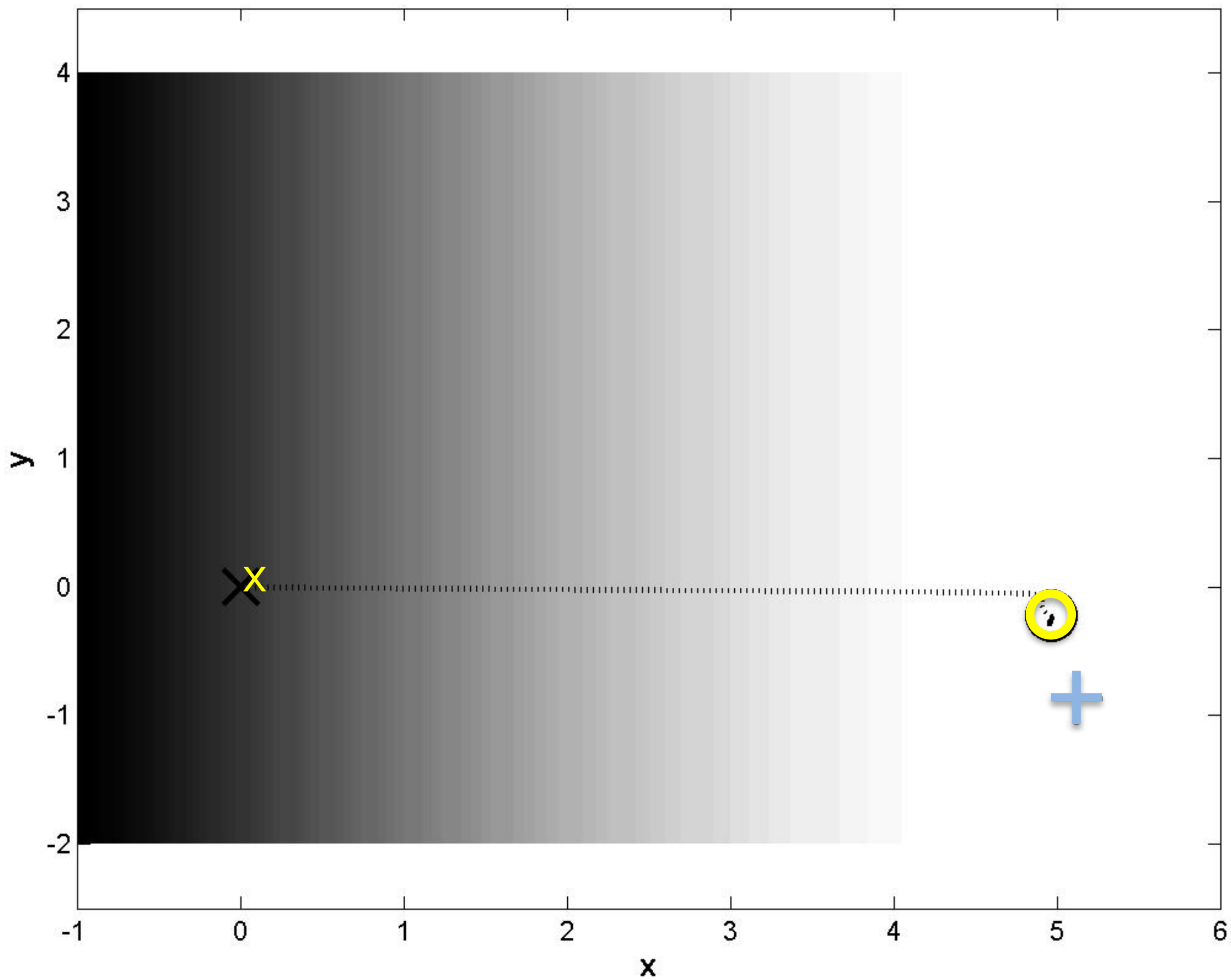
Replanning: light-dark problem



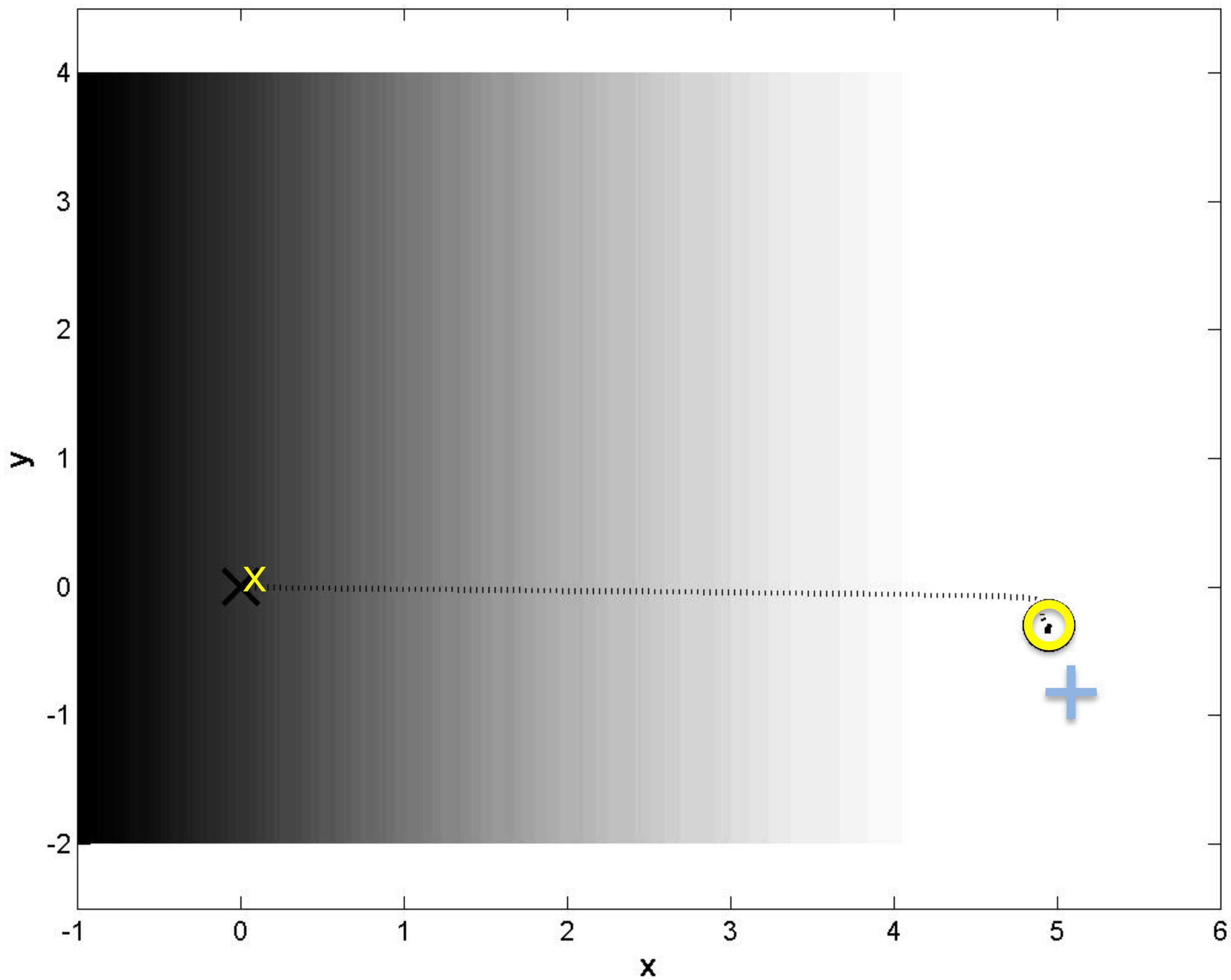
Replanning: light-dark problem



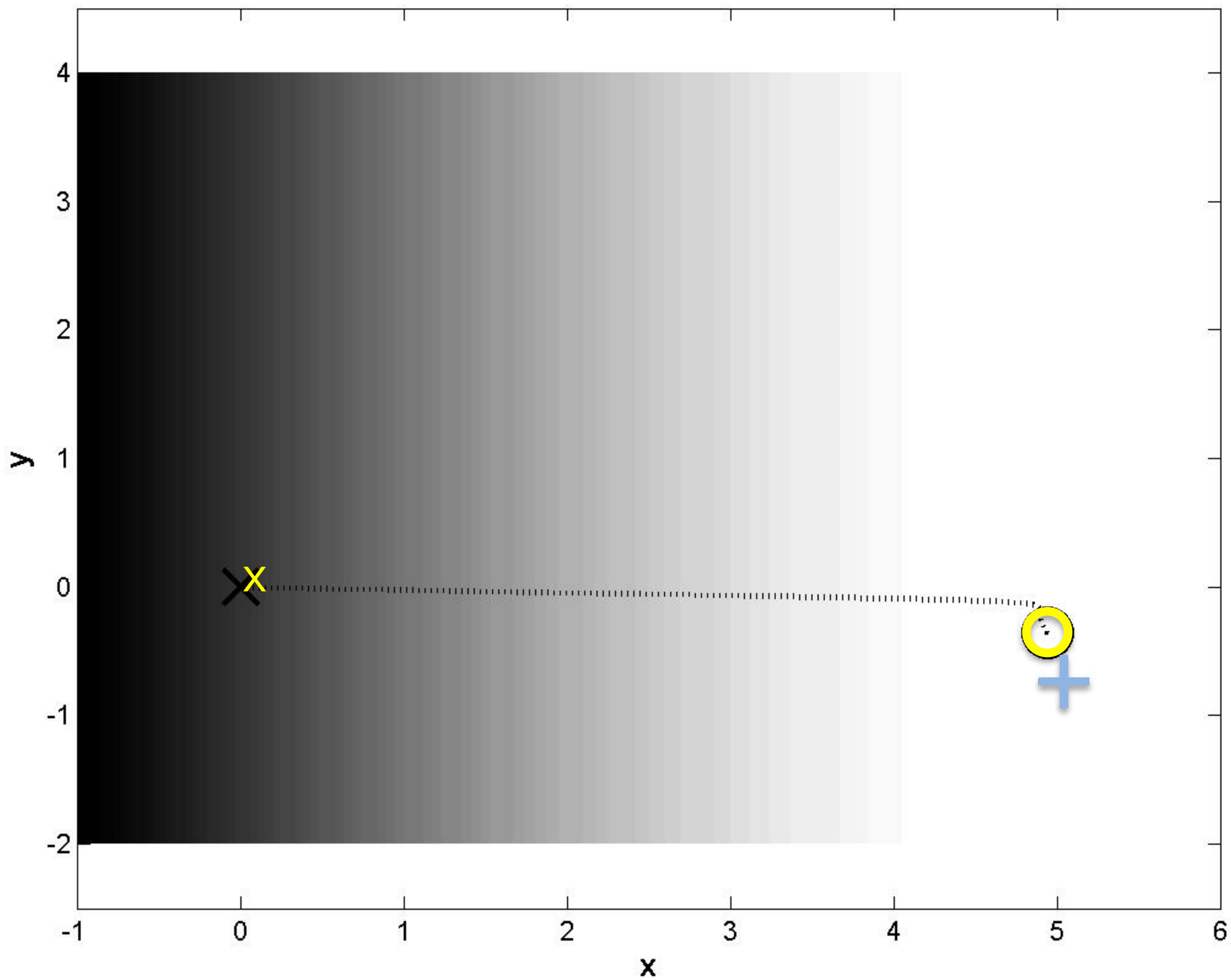
Replanning: light-dark problem



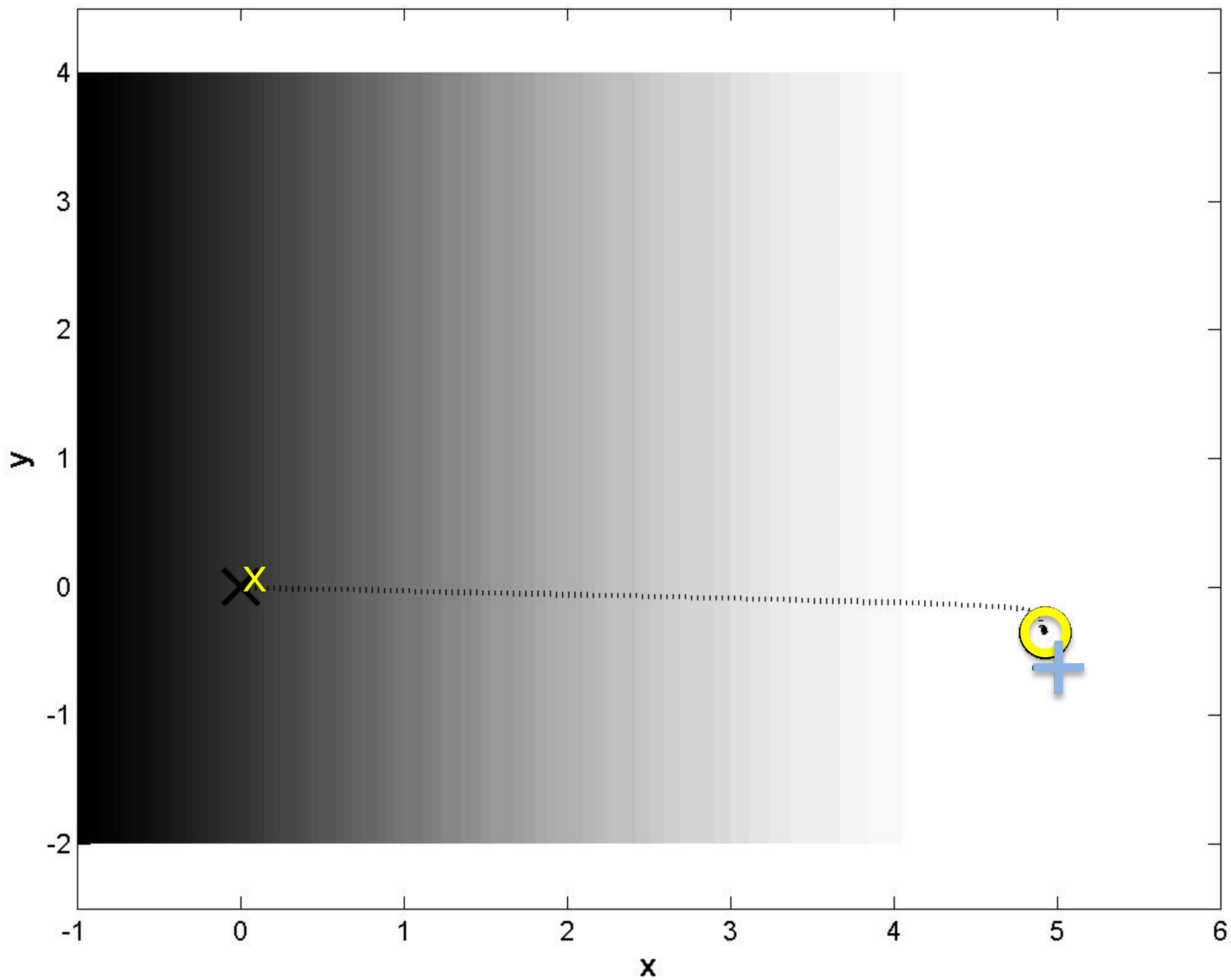
Replanning: light-dark problem



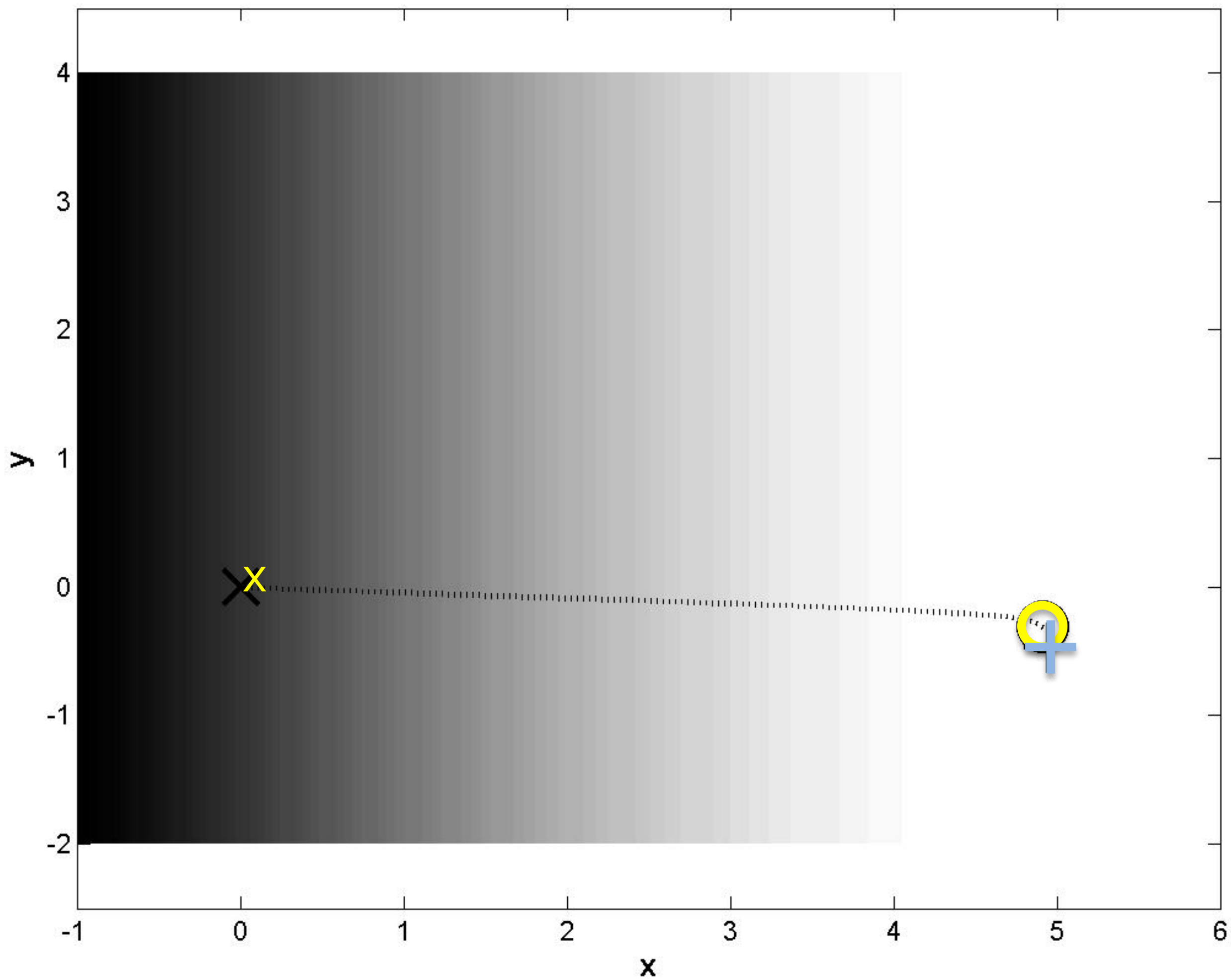
Replanning: light-dark problem



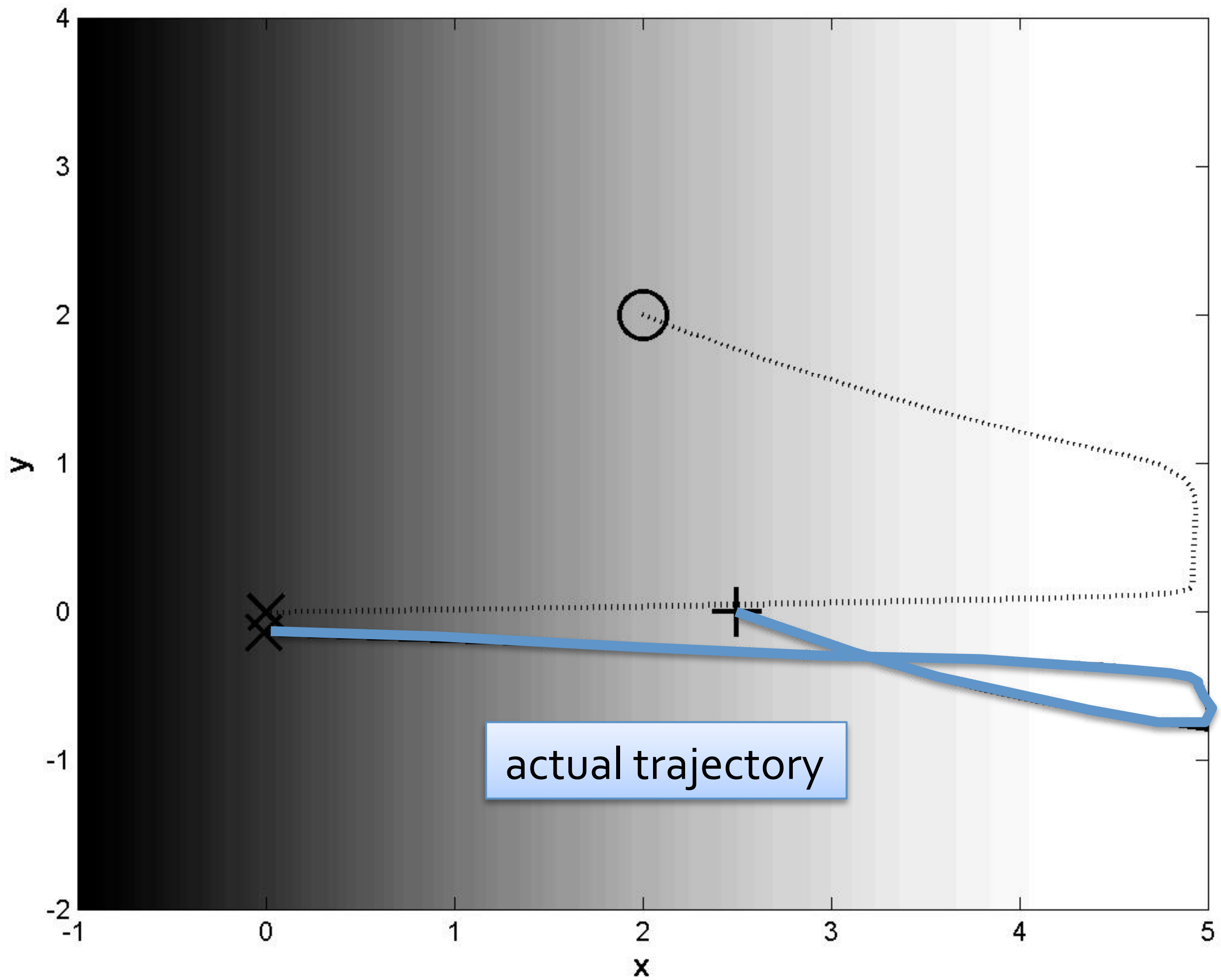
Replanning: light-dark problem



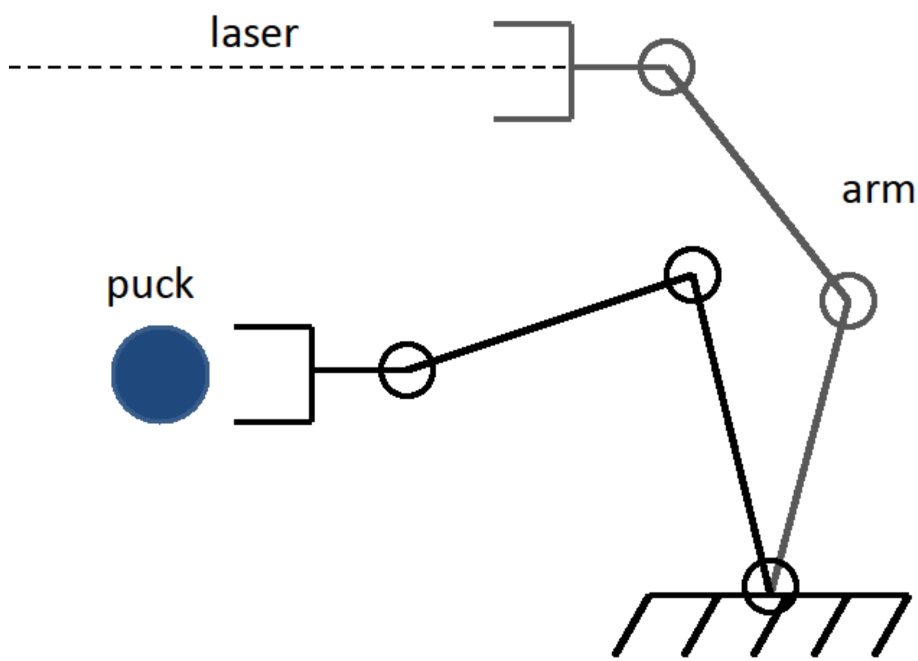
Replanning: light-dark problem



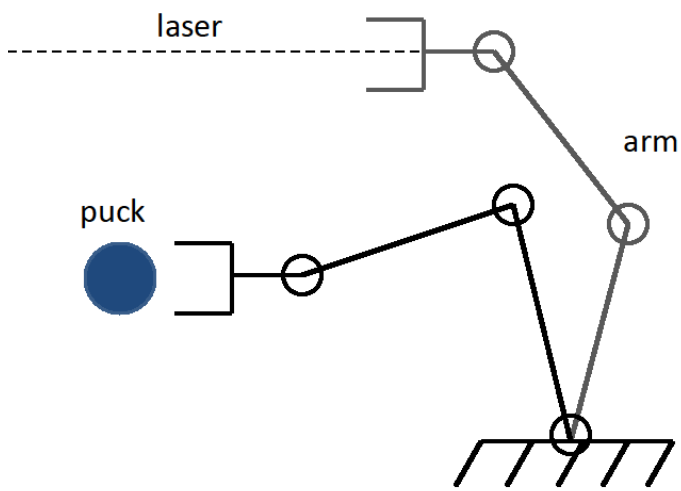
Replanning: light-dark problem

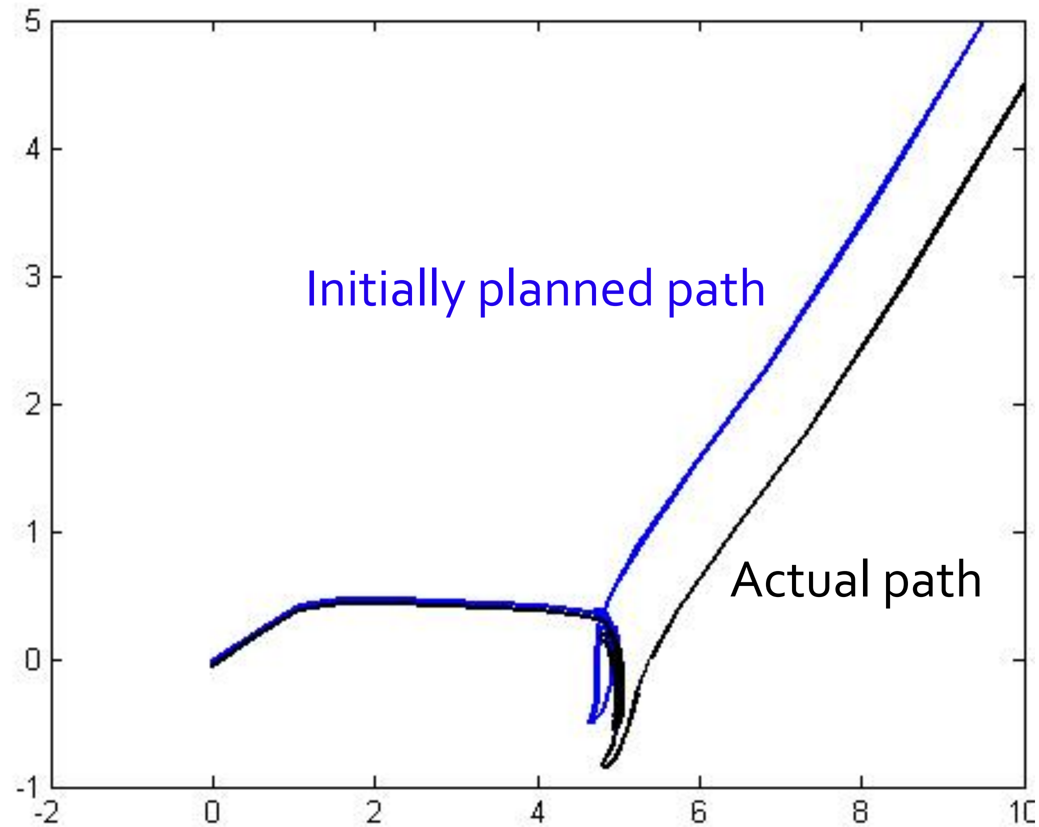


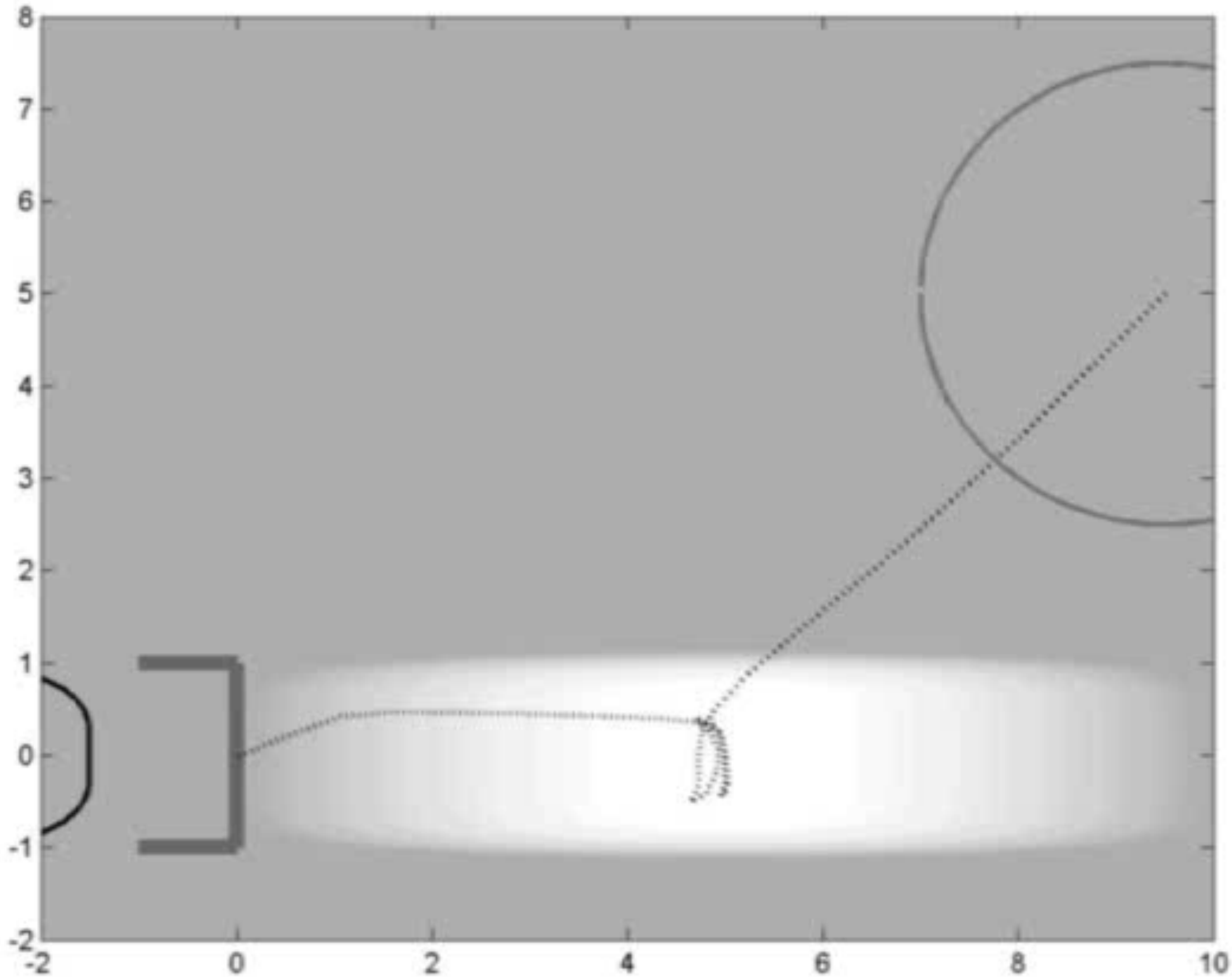
Laser-grasp domain



Laser-grasp: reality







8
7
6
5
4
3
2
1
0
-1
-2
-2
0
2
4
6
8
10

Using simplified models for action selection

Three examples in partially observable domains

Continuous control with state-dependent observation noise:

- deterministic dynamics
- most likely observation

Robot grasping with tactile sensing:

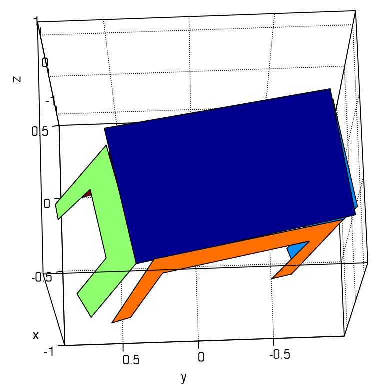
- shortened horizon
- reduced action space

Household robot with local sensing:

- assume subtask serializability
- assume desired observations

Goal: pick up object of known shape with specific grasp

Visual localization and detection works moderately well…



Joint work with Kaijen Hsiao and Tomás Lozano-Pérez

Powerdrill: 10 / 10 successful grasps

25x
speed

Using simplified models for action selection

Three examples in partially observable domains

Continuous control with state-dependent observation noise:

- deterministic dynamics
- most likely observation

Robot grasping with tactile sensing:

- shortened horizon
- reduced action space

Household robot with local sensing:

- assume subtask serializability
- assume desired observations

Classes of robotics problems in which:

- Problems are huge:
  - long horizon
  - many continuous dimensions
  - combinatoric discrete aspects
- No terrible outcomes
- Geometry is not intricate
- Partial observability: local but fairly reliable

Joint work with Tomás Lozano-Pérez

Symbols to Angles

Initial state known in geometric detail

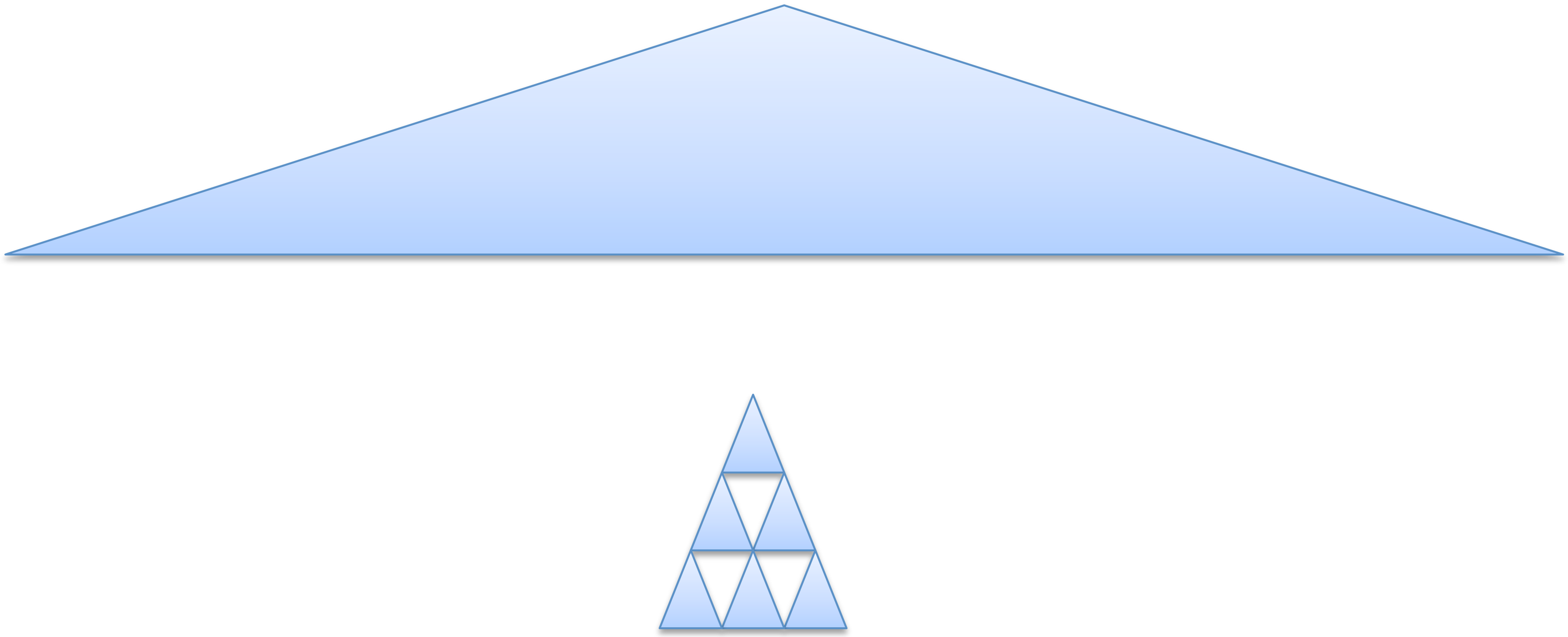
Goal set is abstract, symbolic

tidy(house) ^ charged(robot)

Operator descriptions:

- STRIPS-like, with continuous values
- procedures suggest values for existential vars
- geometric reasoning

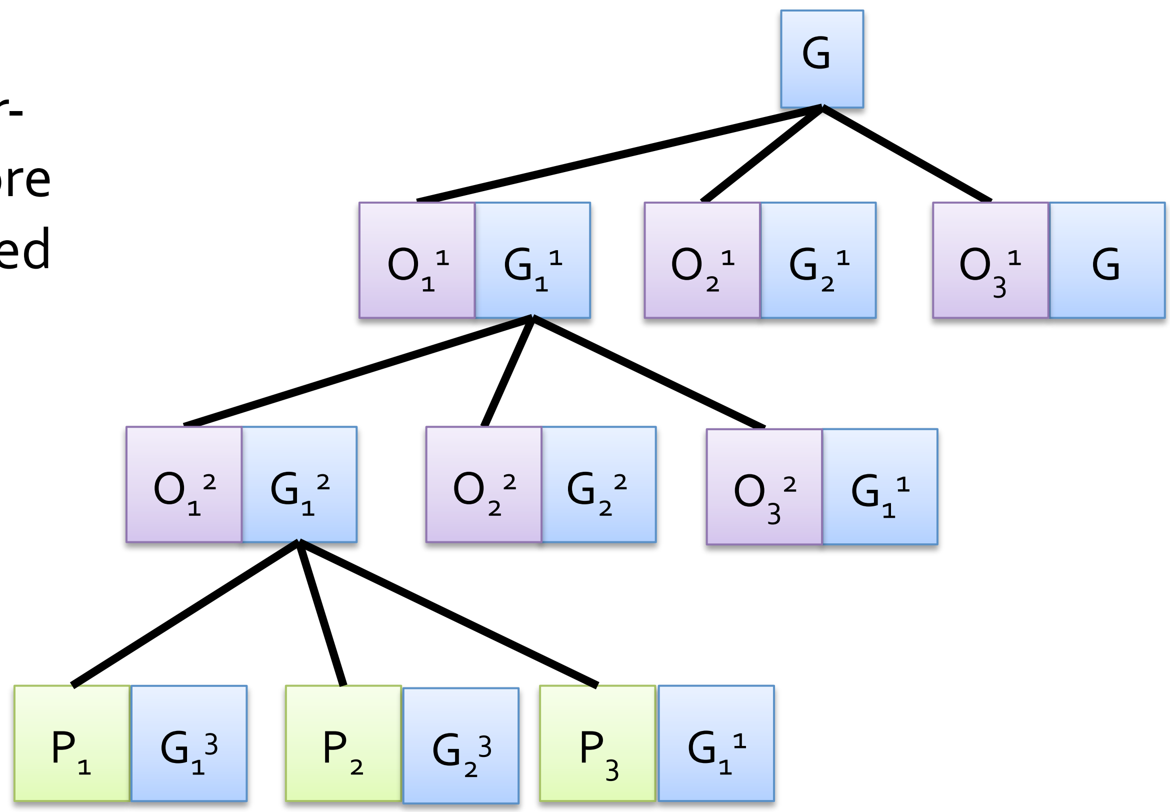
Hierarchy crucial for large problems



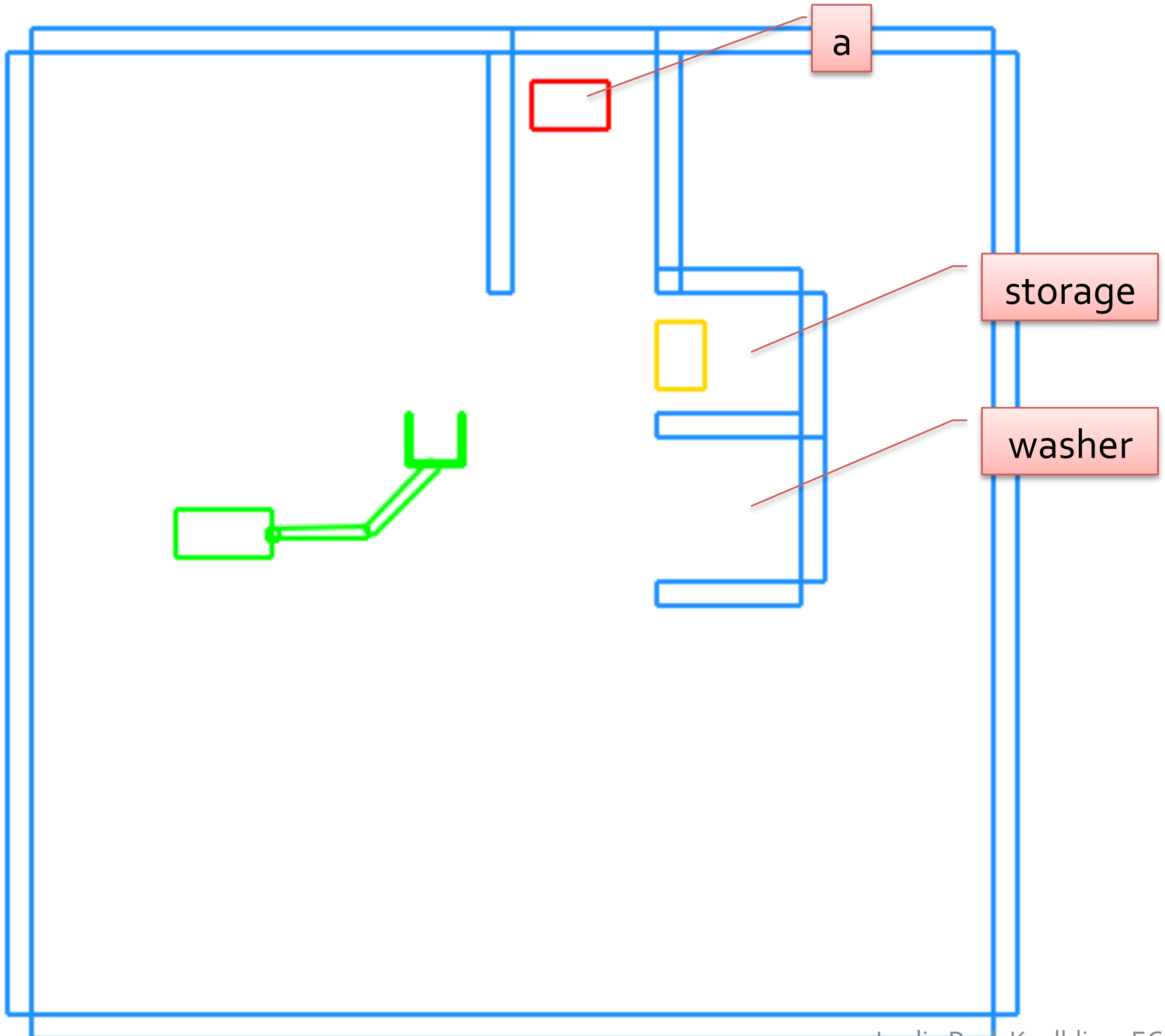
Subtrees represent **serialized subtasks**

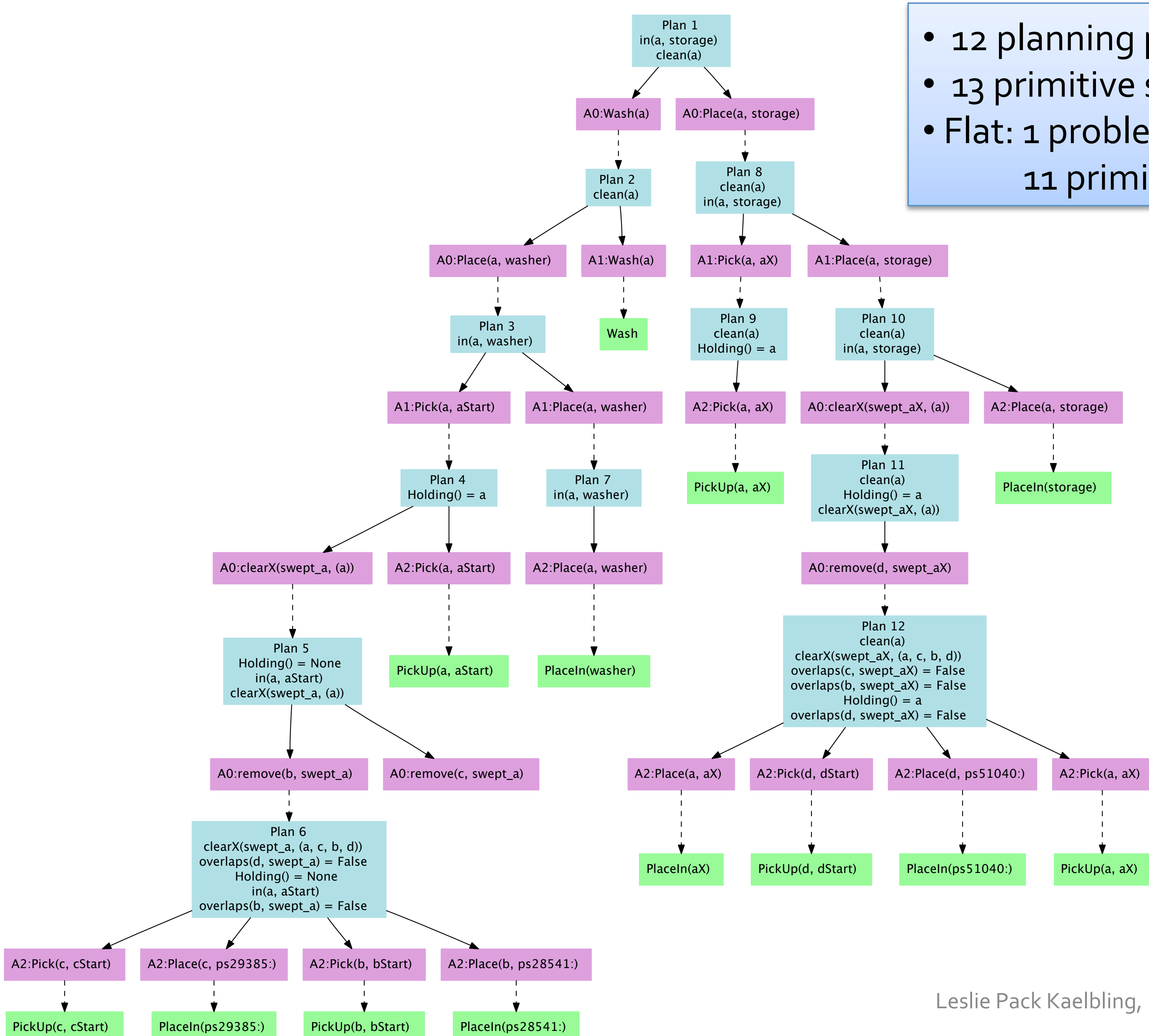
Planning in the now

- maintain left expansion of plan tree
- each level uses a higher-fidelity model: with more preconditions elaborated
- keep track of weakest preconditions for each operation in each plan
- recursively plan to achieve those preconditions
- execute primitives
- replan if preconditions of a plan step are ever violated



Wash a block and put it away

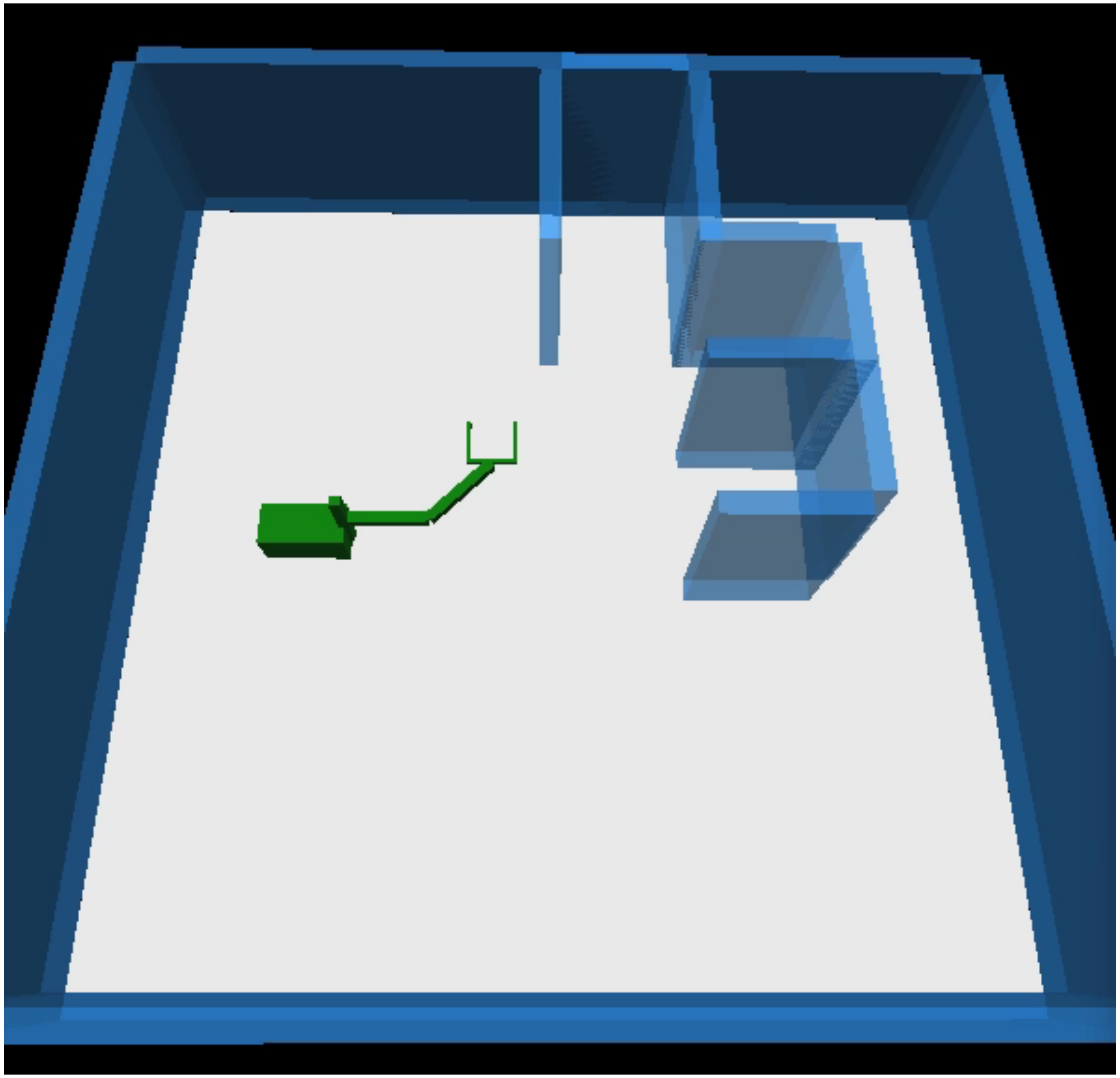




- 12 planning problems
- 13 primitive steps
- Flat: 1 problem,
  11 primitive steps

Leslie Pack Kaelbling, ECML/PKDD-2010

Wash a block and put it away



Planning in the Know

Plan in the **now in belief space**:

- Plan can depend on obtaining particular observations
- Construct single plan that will succeed with high probability
- Replan on unexpected observations

Plan at the "**knowledge level**"

- Traditional to plan in the powerset of the state space
- We have potentially infinite state space
- Use explicit logical representation of knowledge and lack of knowledge

Knowledge fluents

Fluent: $\phi = v$

$$\mathrm{Loc}(\mathrm{vacuum}) = \mathrm{livingRoom}$$

Knowing **the** value: $K_\epsilon(\phi = v) \equiv \Pr(\phi = v) > 1 - \epsilon$

$$K(\mathrm{Loc}(\mathrm{vacuum}) = \mathrm{livingRoom})$$

Knowing **a** value: $KV_\epsilon(\phi) \equiv \exists v. K_\epsilon(\phi = v)$

$$KV(\mathrm{Loc}(\mathrm{vacuum}))$$

Operators in knowledge space

Standard operator descriptions automatically extended:

- require preconditions to be known
- add knowledge effects

Grasp

$$\begin{aligned} \text{pre} &: \text{in}(\text{robot}, \mathsf{R}) = \mathsf{T} \wedge \text{in}(\mathsf{O}, \mathsf{R}) = \mathsf{T} \\ \text{post} &: \text{holding}(\mathsf{O}) = \mathsf{T} \end{aligned}$$

Grasp

$$\begin{aligned} \text{pre} &: \mathsf{K}(\text{in}(\text{robot}, \mathsf{R}) = \mathsf{T}) \wedge \mathsf{K}(\text{in}(\mathsf{O}, \mathsf{R}) = \mathsf{T}) \\ \text{post} &: \text{holding}(\mathsf{O}) = \mathsf{T} \wedge \mathsf{K}(\text{holding}(\mathsf{O}) = \mathsf{T}) \end{aligned}$$

Observation probabilities

Given an operator with knowledge effect,
result can be any desired value, with cost: $-\log \Pr(\phi = v)$

Look

$$\begin{aligned} \text{pre} &: \mathsf{K}(\text{in}(\text{robot}, \mathsf{R}) = \mathsf{T}) \\ \text{post} &: \mathsf{KV}(\text{in}(\mathsf{O}, \mathsf{R})) \end{aligned}$$

C_0

Look

$$\begin{aligned} \text{pre} &: \mathsf{K}(\text{in}(\text{robot}, \mathsf{R}) = \mathsf{T}) \\ \text{post} &: \mathsf{K}(\text{in}(\mathsf{O}, \mathsf{R}) = \mathsf{T}) \end{aligned}$$

$C = -\log \Pr(\text{in}(\mathsf{O}, \mathsf{R}) = \mathsf{T}) + C_0$

Going on a tiger hunt

move(Room):

post: robotLoc = Room

listen:

pre: robotLoc = hall

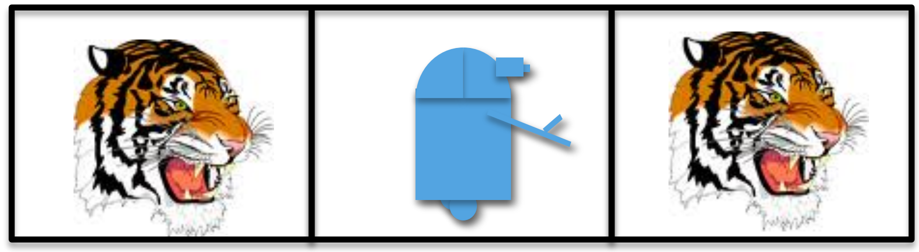
post: KV(tigerLoc)

shoot:

pre: robotLoc = tigerLoc

post: deadTiger

P(tigerLoc = leftRoom) = 0.8



Going on a tiger hunt: regression search tree

move(Room):

post: robotLoc = Room

listen:

pre: robotLoc = hall

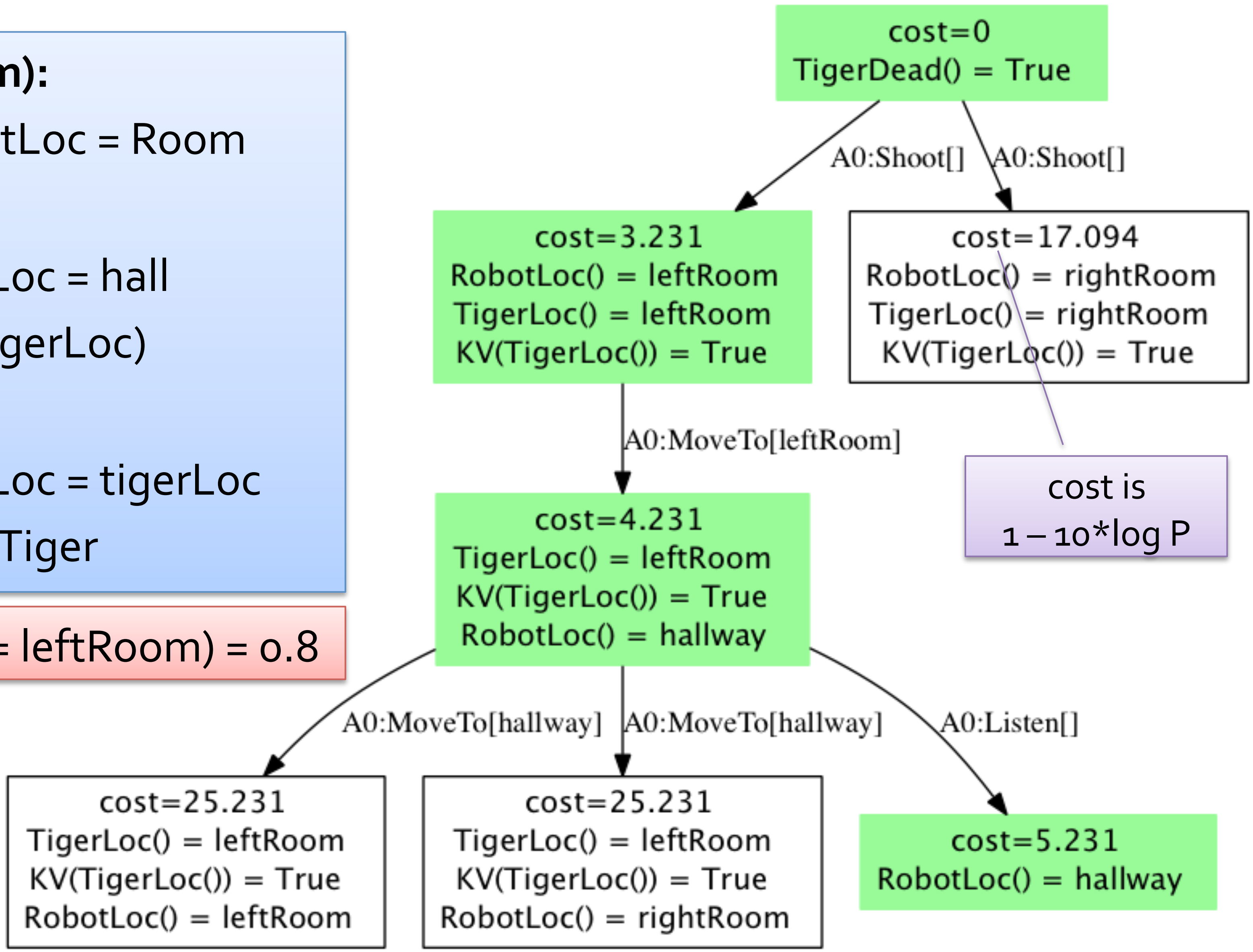
post: KV(tigerLoc)

shoot:

pre: robotLoc = tigerLoc

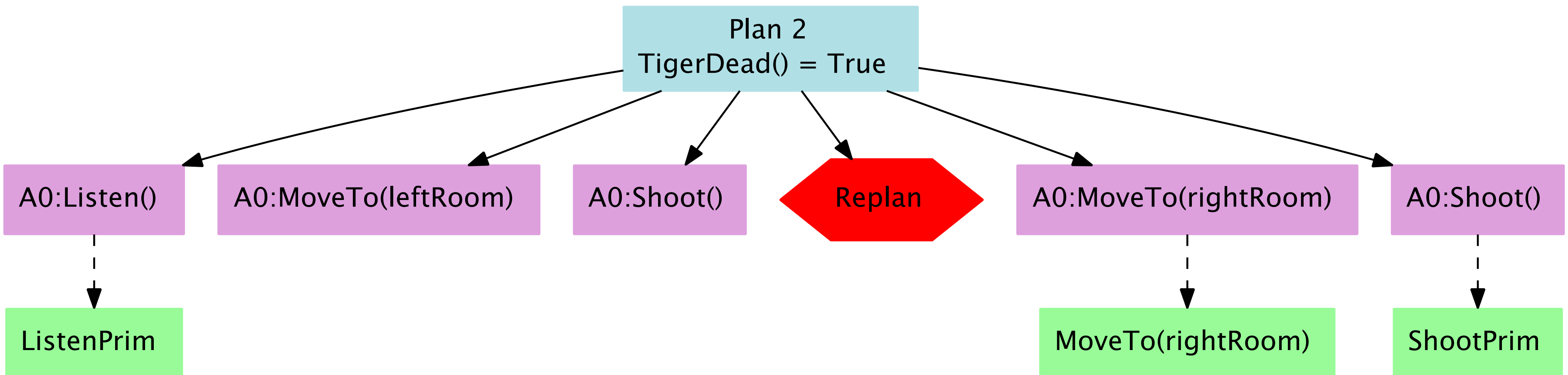
post: deadTiger

P(tigerLoc = leftRoom) = 0.8



Monitor execution and replan

- Listen, expecting to hear tiger on the left
- Hear tiger on right
- Replan

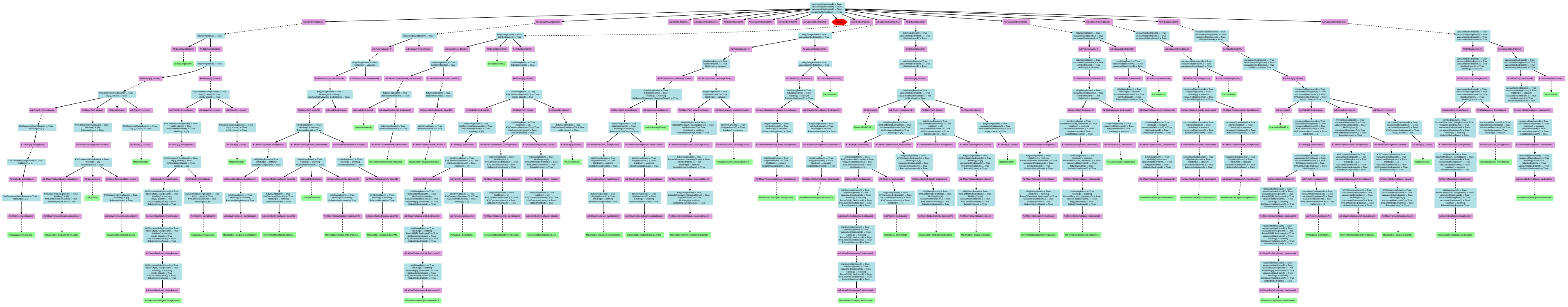


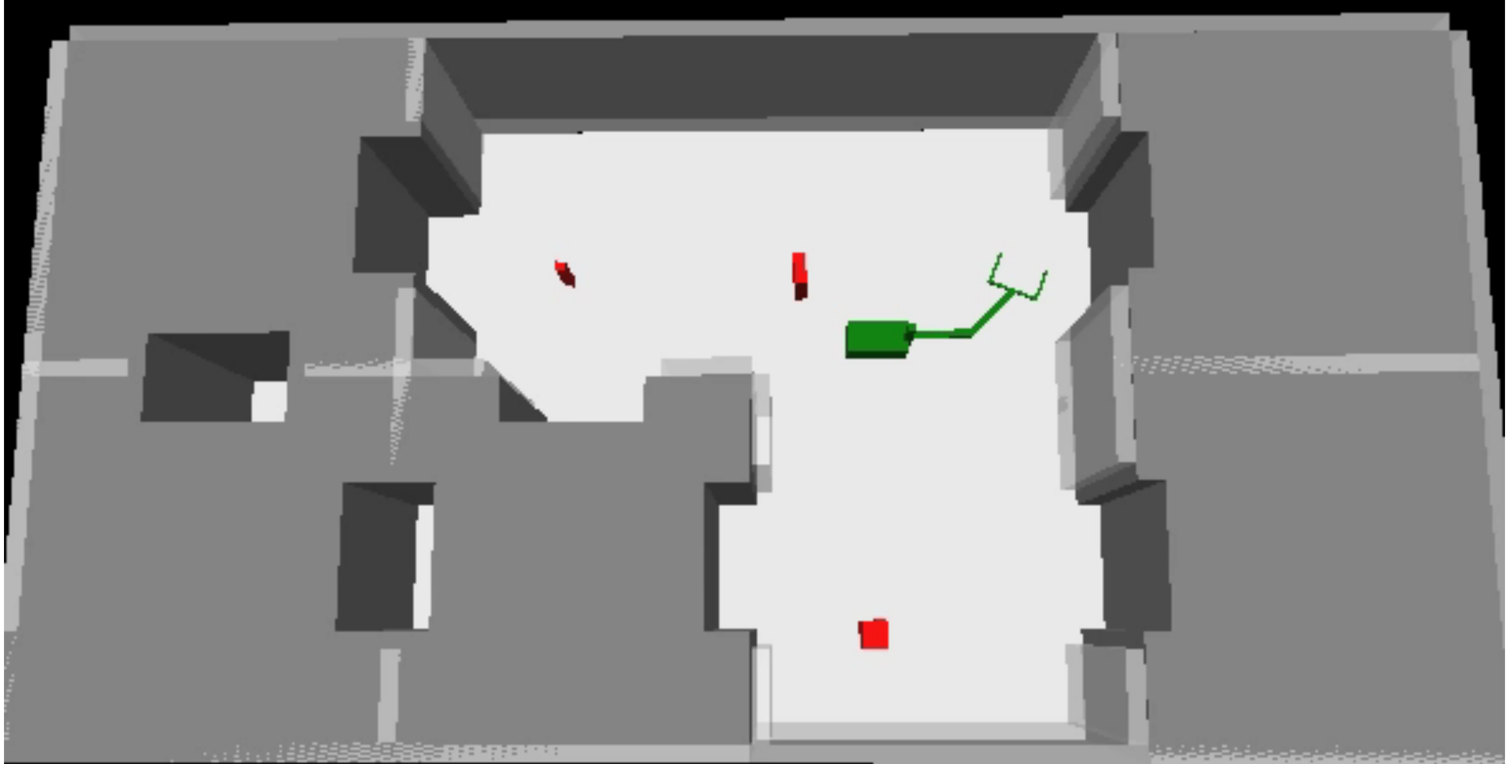
Cleaning house

Goal: vacuum four of the rooms in the house

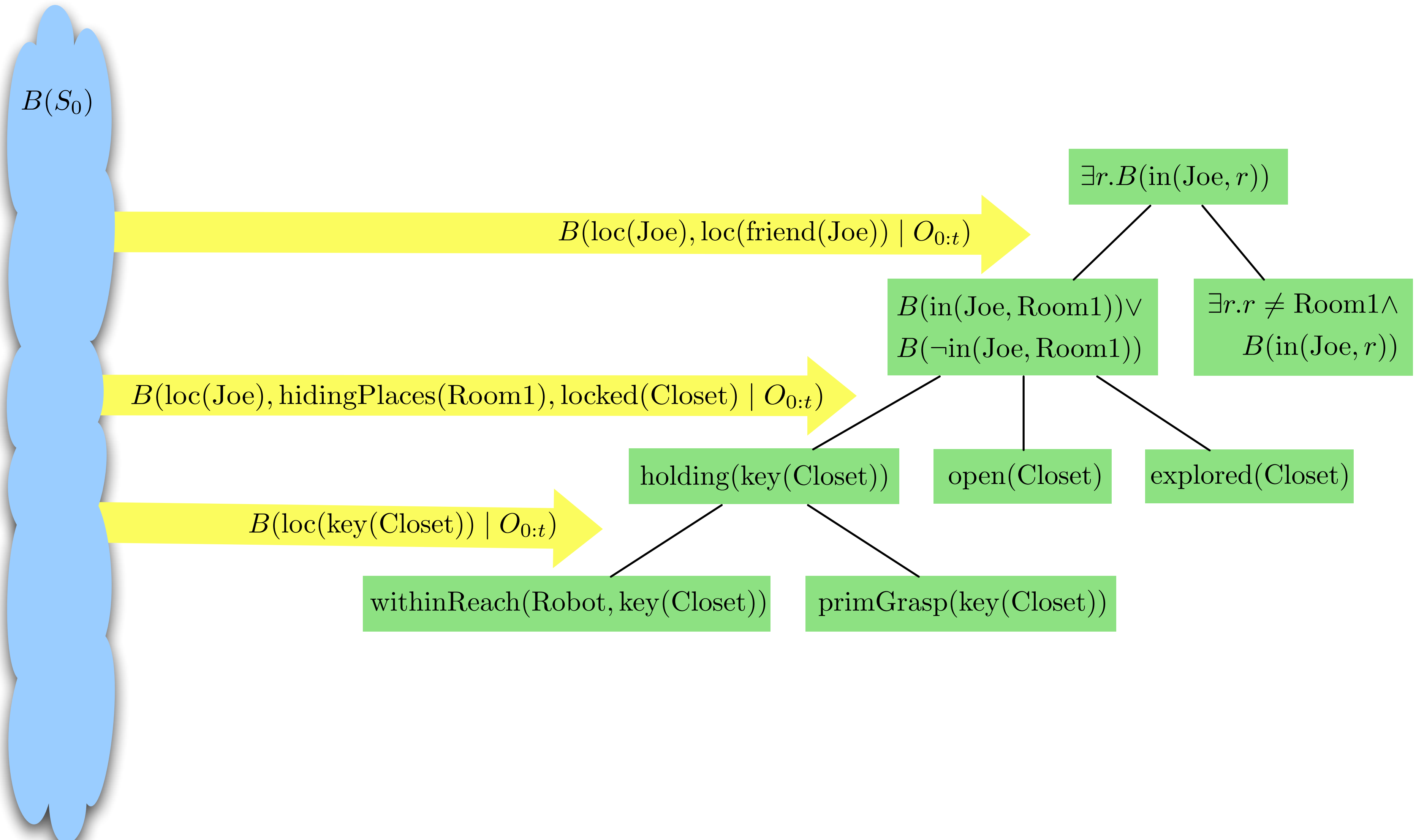
- have to put away junk items before vacuuming
- location of junk is unknown
- location of vacuum is unknown

Plans are made assuming likely belief; replan as necessary

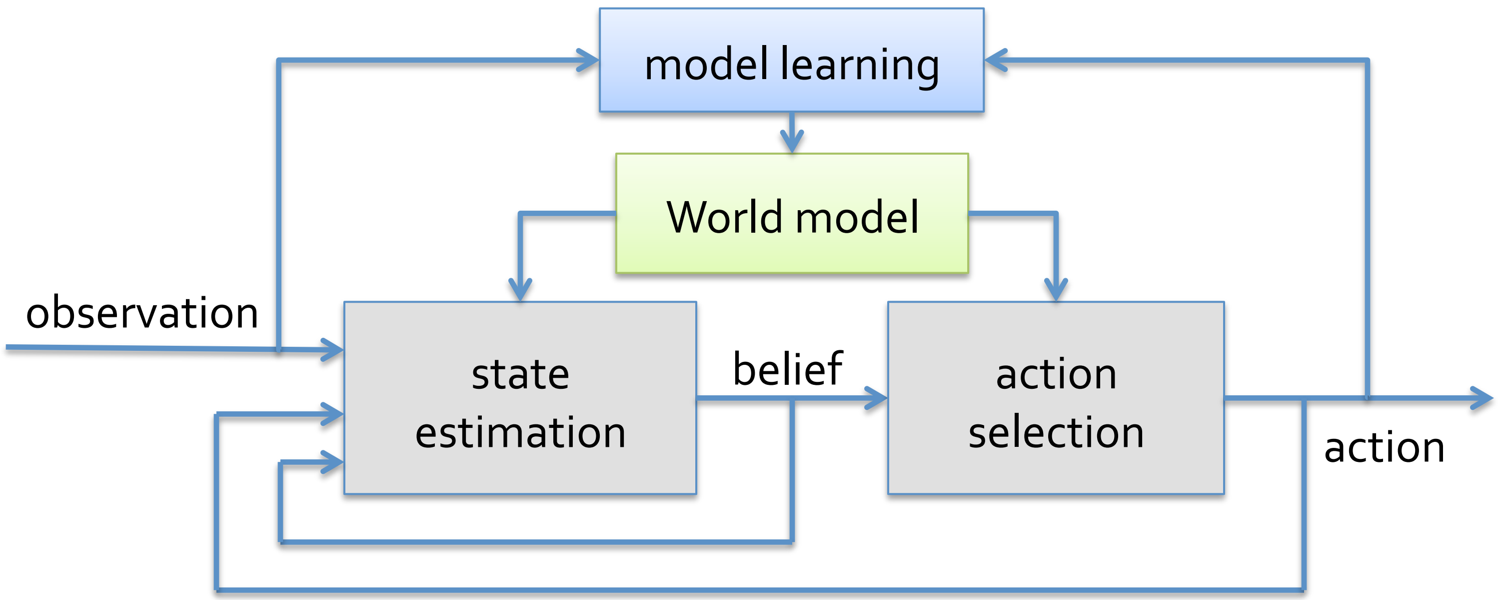




Plan hierarchy can pose small filtering problems

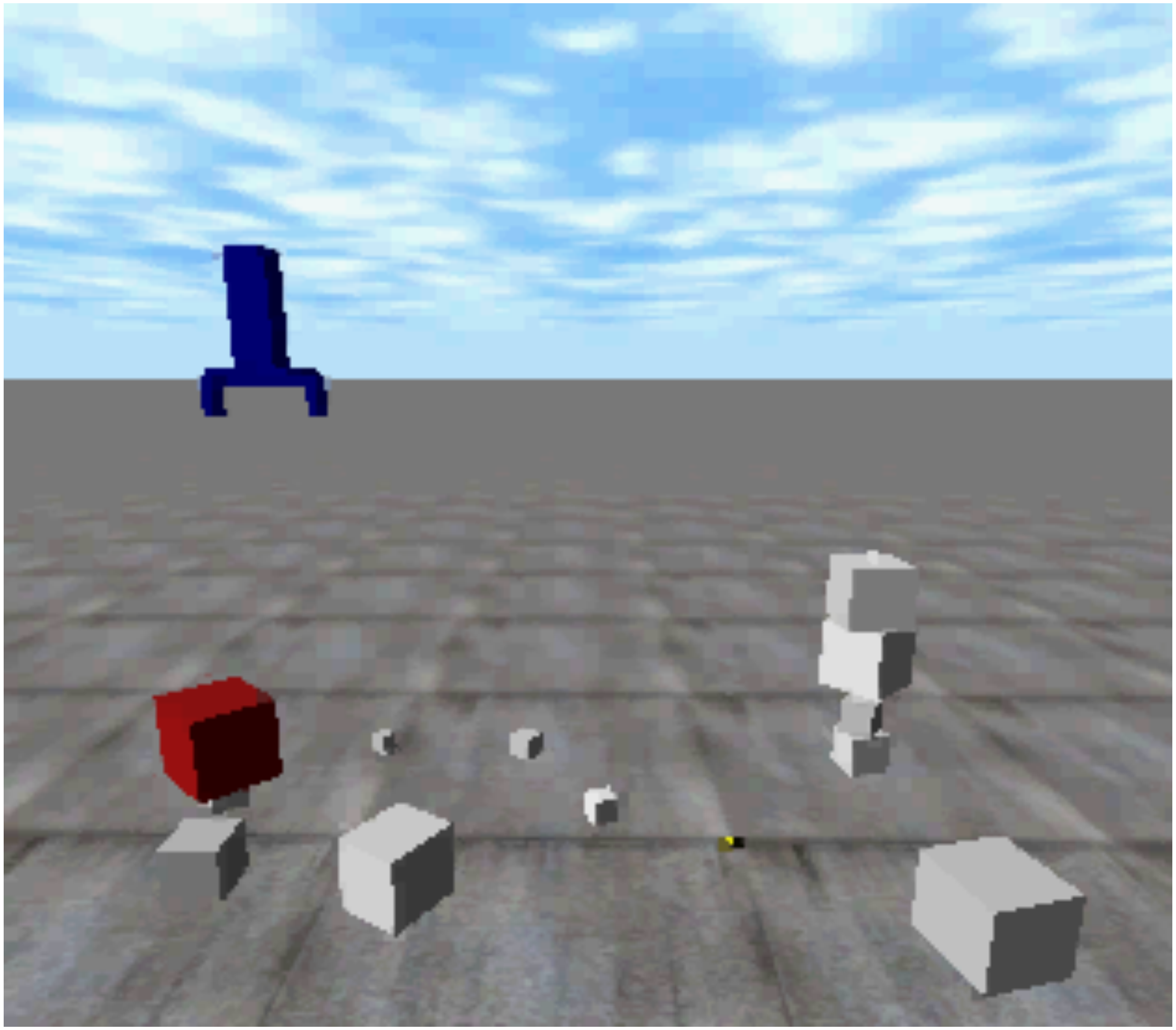


Learning models



- Factoring, lifting are crucial
- Fidelity doesn't have to be perfect
- Ultimately, partially observable

Blocks with physics



Joint work with Hanna Pasula and Luke Zettlemoyer

Representing a world model

Representation should:

- allow effective generalization
- be useful for planning
- be efficiently learnable

High fidelity model: detailed physical dynamics equations

Low fidelity model: probabilistic state transition dynamics over discretized state space

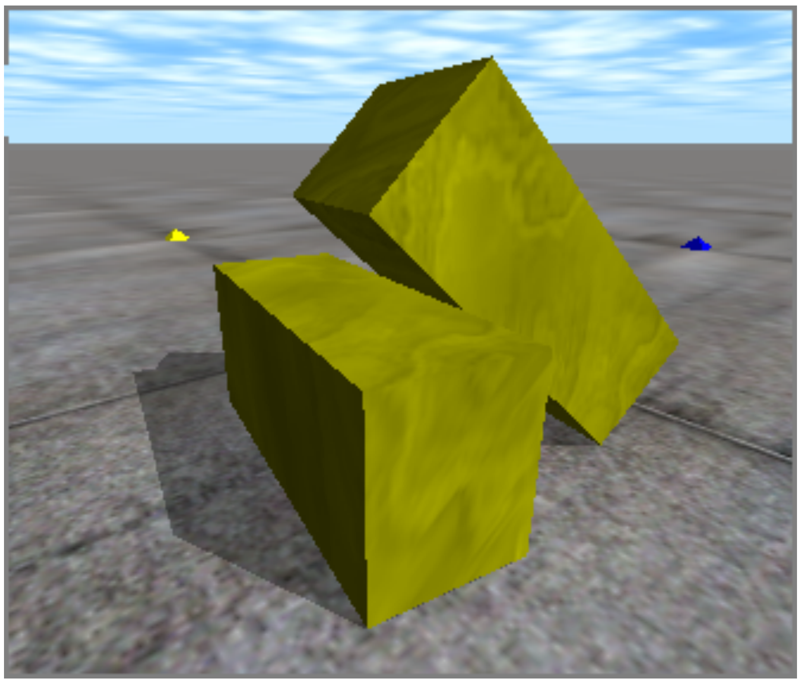
$$\Pr(s_t \mid s_{t-1}, a)$$

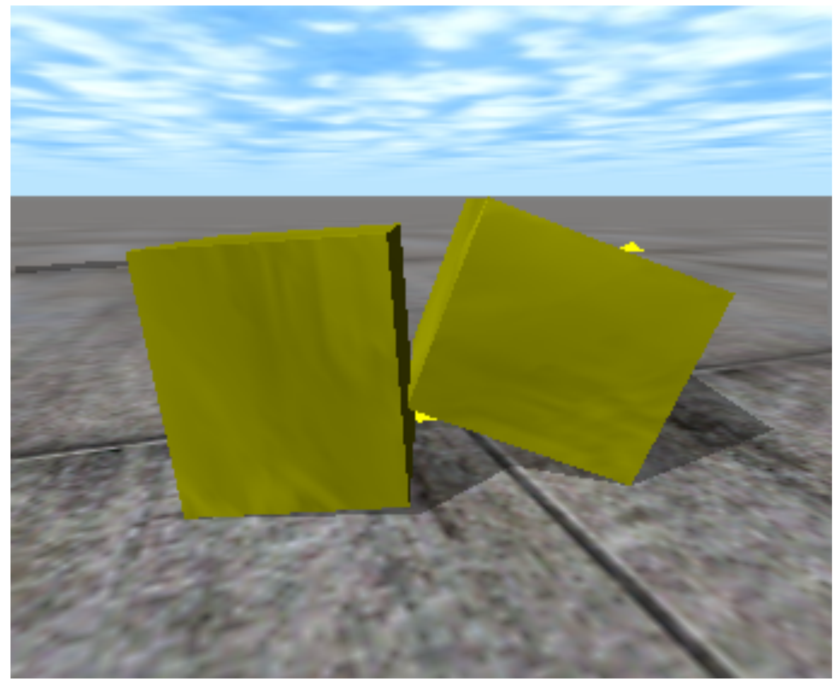
Probabilistic dynamic rules

Combine logic and probability to model effects of actions in complex, uncertain domains

```
pickup(X): {Y: on(X,Y)}
    clear(X), inhand-nil, size(X)>2, size(X)<7 →
      0.803 :¬on(X,Y)
      0.093 : no change
```

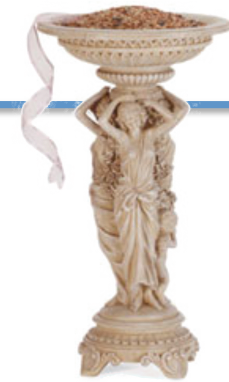
Is X *on Y*?





Useful symbolic vocabulary should be learned

Neoclassical learning



Given experience, $\{\langle s_t, a_t, s_{t+1} \rangle\}$

Find rule set that optimizes

$$\text{score}(R) = \sum_t \log \Pr(s_{t+1} \mid s_t, a_t, R) - \alpha |R|$$

Start with one default rule: "stuff happens"

- Symbolic: add, delete rule; change rule conditions
  - Greedy: choose set of outcomes
    - Convex optimization: find maximum likelihood probabilities

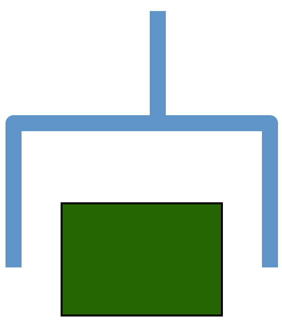
Concept invention

New concepts allow predictive theory to be expressed more compactly and learned from less data

`p1(X) :- ¬∃Y. on(X,Y)`	X is in the hand
`p2() :- ¬∃Z. p1(Z)`	nothing is in the hand
`p3(X) :- ¬∃Y. on(Y,X)`	X is clear
`p4(X,Y) :- on(X,Y)*`	X is above Y
`p5(X,Y) :- p3(X) ∧ p4(X,Y)`	X is on the top of the stack containing Y
`f6(X) :- #Y. p4(X,Y)`	the height of X

Rules learned from data

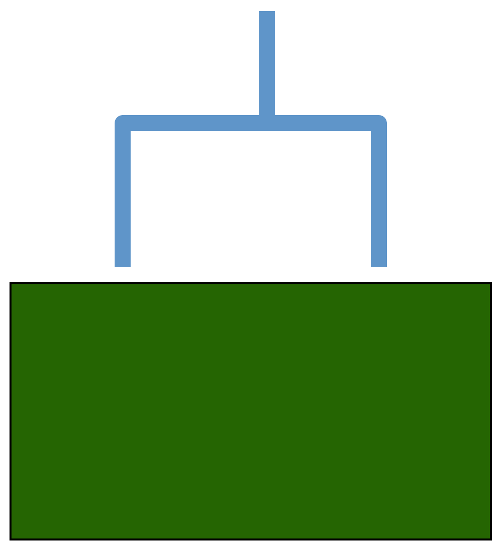
```
pickup(X): {Y: on(X,Y)}
    clear(X), inhand-nil, size(X)>2, size(X)<7→
      0.803 :¬on(X,Y)
      0.093 : no change
```



picking up middle-sized blocks usually works

Rules learned from data

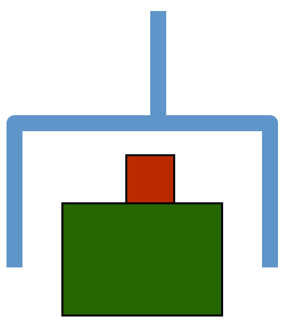
```
pickup(X):
    clear(X), inhand-nil, ¬size(X)<7 →
      0.906 : no change
```



it's impossible to pick up very big blocks

Rules learned from data

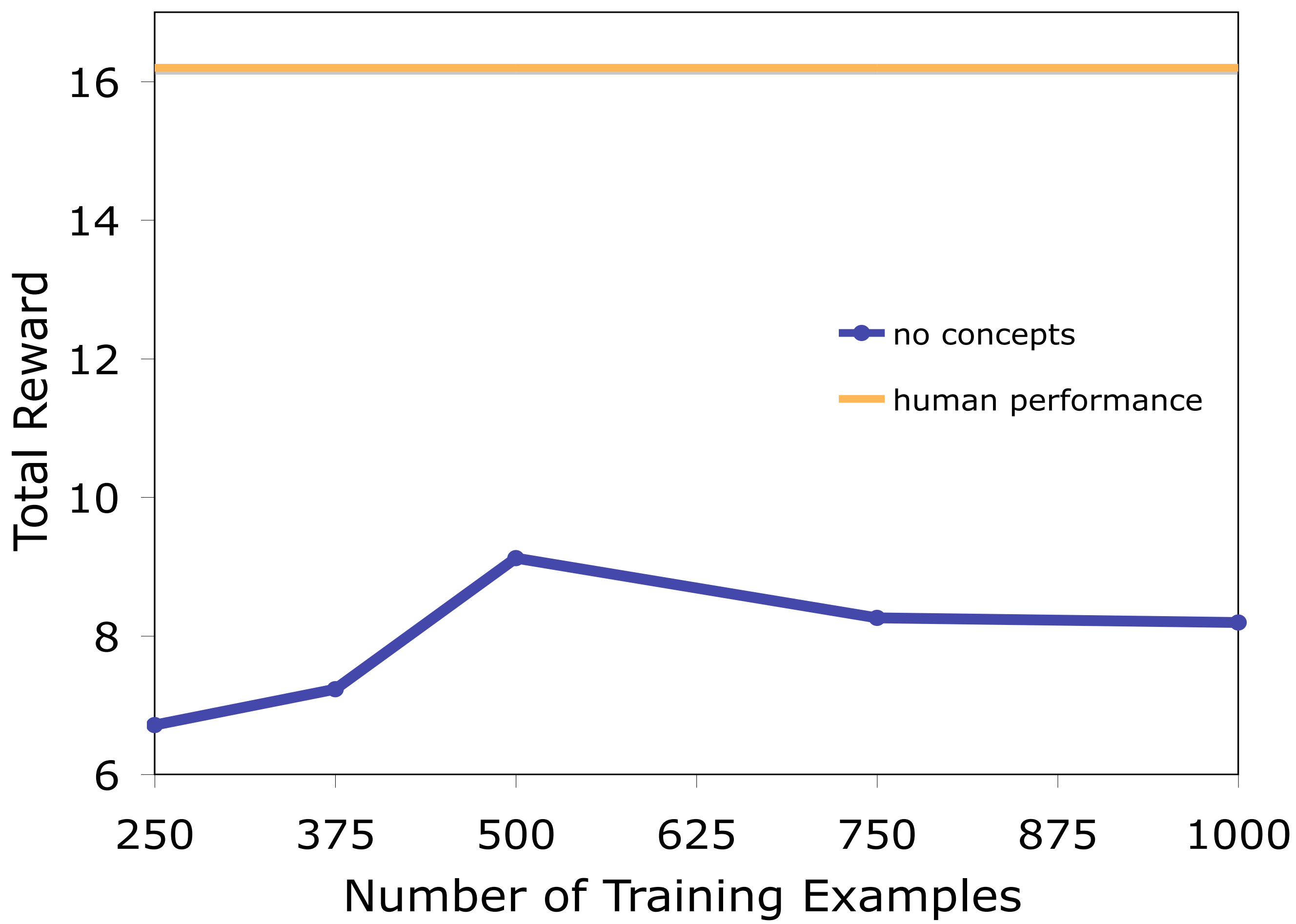
```
pickup(X): {T: table(T)}, {Y: on(X,Y), on(Y,T)}
   clear(X), inhand-nil, size(X)<2  →
      0.105 :¬on(X,Y)
      0.582 :¬on(Y,T)
      0.312 : no change
```



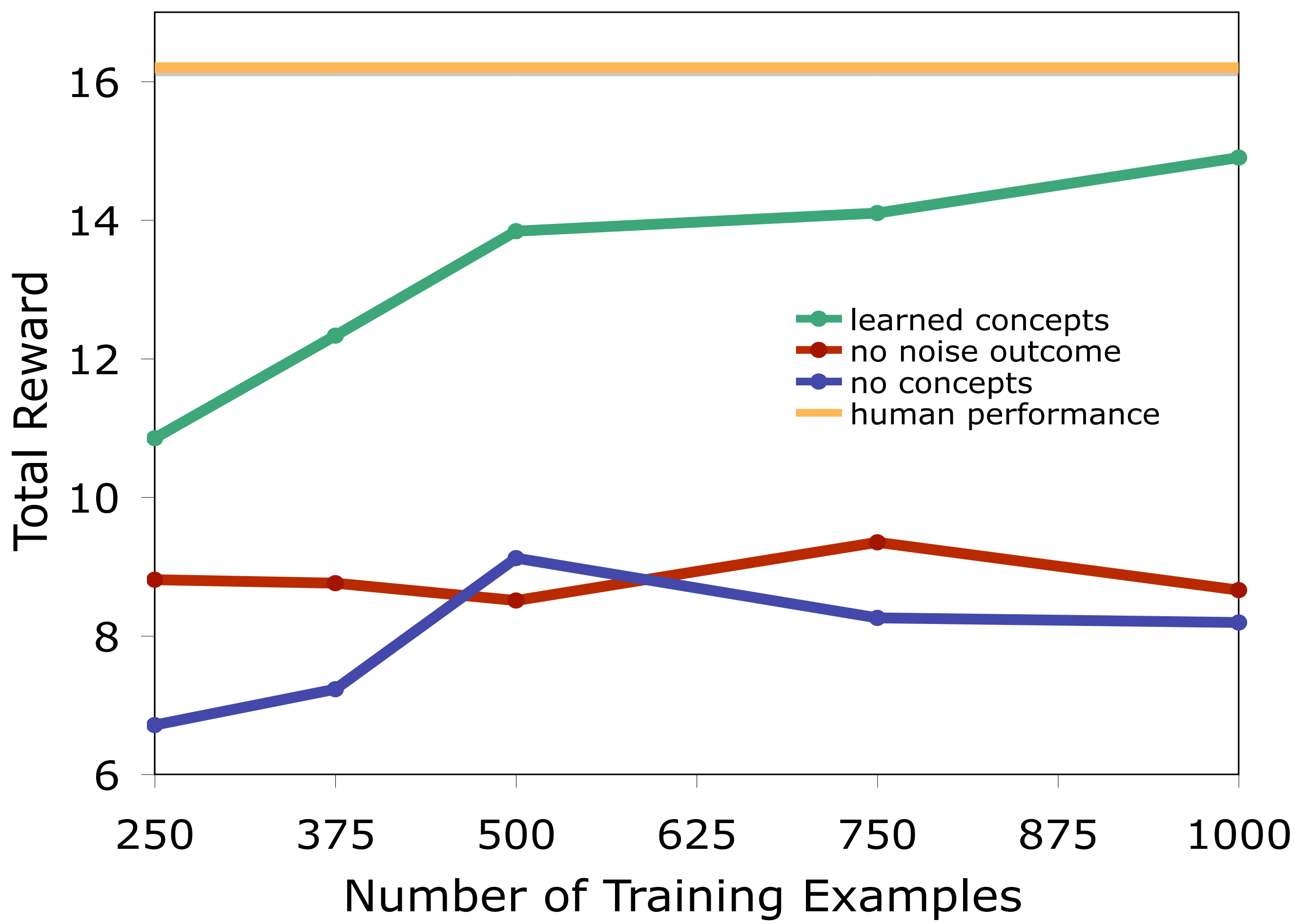
if a tiny block is on another block that is on the table, and we try to pick up the tiny block, we'll often pick up the other block as well, or fail

Planning with learned rules

Planning with learned rules



Planning with learned rules



Learning models of partially observable domains

We (the ML community) usually see this problem as a kind of HMM learning….**hard!**

Lots of opportunities for at least partial supervision:

- Learn dynamics first, from full observation; then learn observation model
  - learning object permanence
- Eventually get local, correct observation
  - see inside the cupboard
  - correlate visual observations with more reliable laser observations

- PSR learning is completely supervised (but compression is the issue)

Help us learn

World dynamics models

- at multiple levels of abstraction
- from semi-partially observable data

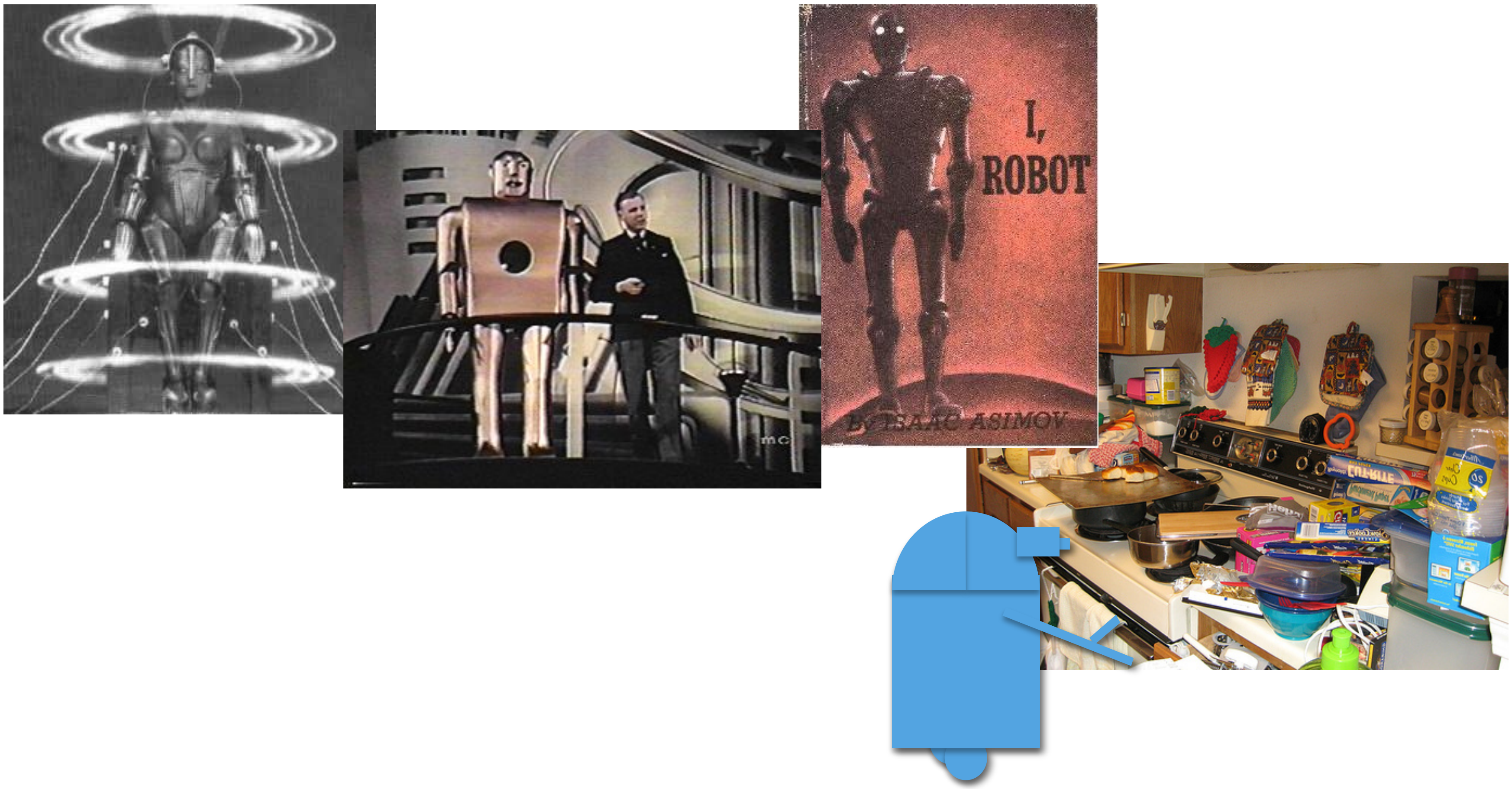
Meta-cognitive knowledge

- how to construct hierarchy effectively
- what aspects of the domain to filter more carefully
- what level of fidelity is needed in a model

Thanks!

Collaborators: Stan Rosenschein, Tom Dean, Tomas Lozano-Perez, Michael Littman, Tony Cassandra, Hagit Shatkay, Jim Kurien, Nicolas Meuleau, Milos Hauskrecht, Jak Kirman, Ann Nicholson, Bill Smart, Luis Ortiz, Leon Peshkin, Mike Ross, Kurt Steinkraus, Yu-Han Chang, Paulina Varshavskaya, Sarah Finney, Kaijen Hsiao, Luke Zettlemoyer, Han-Pang Chiu, Natalia Hernandez, James McLurkin, Emma Brunskill, Meg Aycinena Lippow, Tim Oates, Terran Lane, Georgios Theocharous, Kevin Murphy, Bruno Scherrer, Hanna Pasula, Brian Milch, Bhaskara Marthi, Kristian Kersting, Sam Davies, Dan Roy, Jenny Barry, Selim Temizer, Rob Platt, Russ Tedrake, David Silver, Frans Oliehoek

Funders: NSF, DARPA, AFOSR, ONR, NASA, Singapore-MIT Alliance, Gambit, Ford, Boeing, BAE Systems


I,
ROBOT
By ISAAC ASIMOV