Inference & Planning Lecture

Robot Learning Summer School, Lisbon Portugal, 21st July 2009

Marc Toussaint
Machine Learning & Robotics Group, TU Berlin
mtoussai@cs.tu-berlin.de

Part 1: Introduction to probabilistic inference & learning

Part 2: Planning by Inference

Outline

- **Part 1: Introduction to probabilistic inference & learning**

  ...

- **Part 2: Planning by Inference**
  - typical use of inference in robotics: sensor $\rightarrow$ state
  - general idea of inference by planning
  - Markov Decision Processes:

    likelihood vs. expected rewards, EM learning for optimal policies, policies in structured processes, in relational domains, model-free Reinforcement Learning based on EM

  - Stochastic Optimal Control:

    likelihood vs. expected costs, Approximate Inference Control

- **Summary & further reading**

  ...

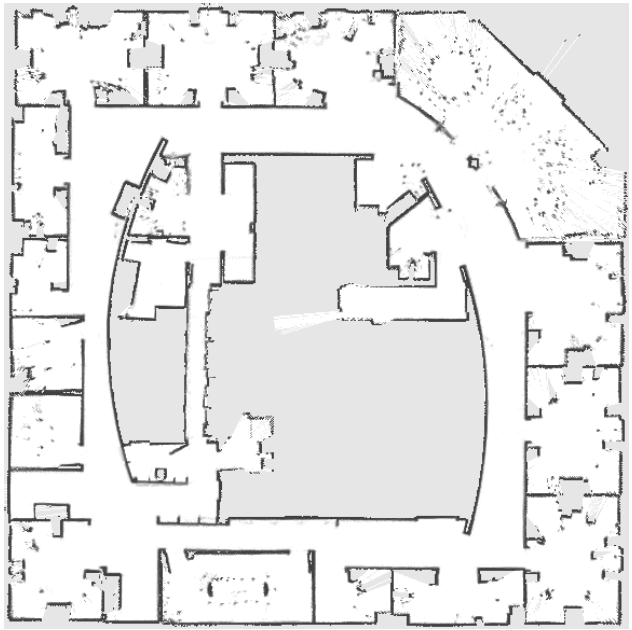
Typical use of inference in robotics

- robotics: many hard problems arise on the sensor side:
  e.g., robot localization, vision, scene understanding, object localization, even speech processing, cameras/lasers/haptics/microphone

- classical: inference for sensor processing:
  $$\text{sensors} \rightarrow P(\text{state}|\text{sensors})$$

- example: robot localization...

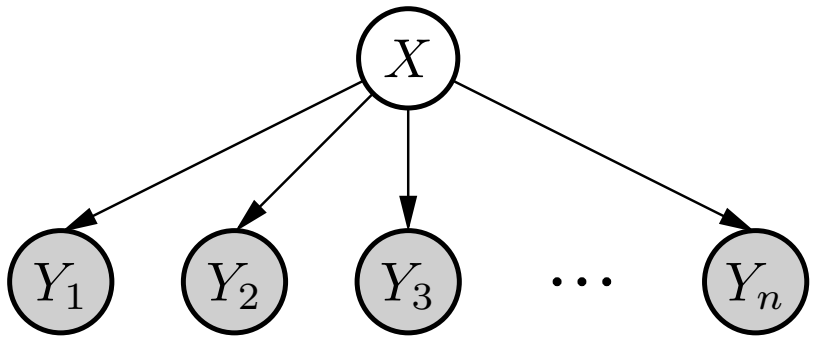
Example: robot localization



- mobile robot:
  - state = (position,orientation) = $X = (x_0, x_1, \theta) \in \mathbb{R}^3$
  - sensor readings = n laser ranges $Y_{1:n} \in \mathbb{R}$ for different angles $\alpha_{1:n}$

Example: robot localization

- probabilistic formulation:



this is

Naive Bayes!

– inference:

$$P(X|Y_{1:n}) \propto P(X) \prod_{i=1}^{n} P(Y_i|X)$$

(doing this computation is non-trivial: particles, discretizations, etc.)
Thrun, Burgard & Fox: *Probabilistic Robotics*, MIT Press 2005

Example: robot localization

- robot localization is very important (and big) field
  – “straight-forward” (but computationally non-trivial) application of inference for sensor interpretation
  – extension to temporal data and unknown environment (SLAM)

- lecture part 1: “inference is a generic means of information processing”

  – so far: reasoning about now:

  $$P(\text{states}|\text{sensors})$$

  – now: reasoning about the *future!*

  $$P(\text{actions}|\text{future goals})$$

Outline

- **Part 1: Introduction to probabilistic inference & learning**

  ...

- **Part 2: Planning by Inference**
  - typical use of inference in robotics: sensor $\rightarrow$ state
  - **general idea of inference by planning**
  - Markov Decision Processes:

    likelihood vs. expected rewards, EM learning for optimal policies, policies in structured processes, in relational domains, model-free Reinforcement Learning based on EM

  - Stochastic Optimal Control:

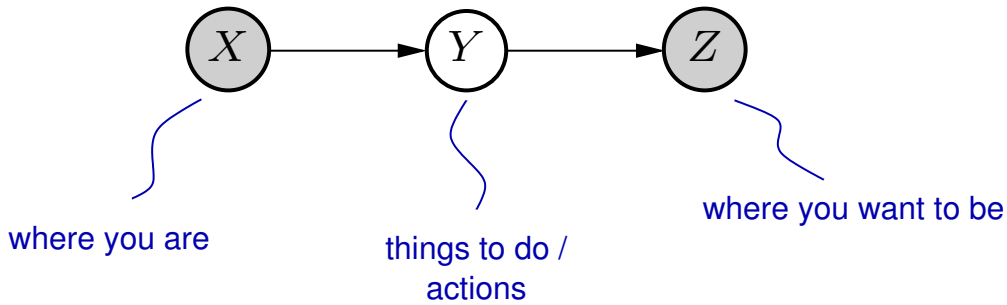
    likelihood vs. expected costs, Approximate Inference Control

- **Summary & further reading**

  ...

inference by planning – general idea

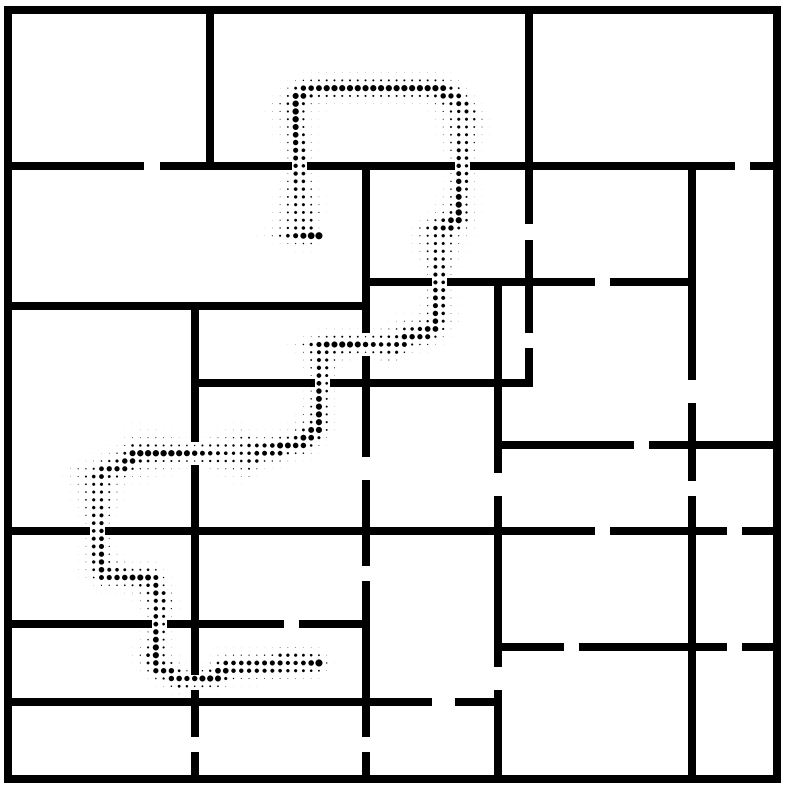
- Example 1:



  – you know where you are right now
  – you know what you *want to see in the future*
  – *What is the posterior over intermediate actions?*
    (assuming we have some model including prior over actions)

inference by planning – general idea

- Example 2:



– Luis is a random walker in a maze starting at location S

– You see (or imagine to see) Luis reaching state G

– *What is the posterior trajectory?*

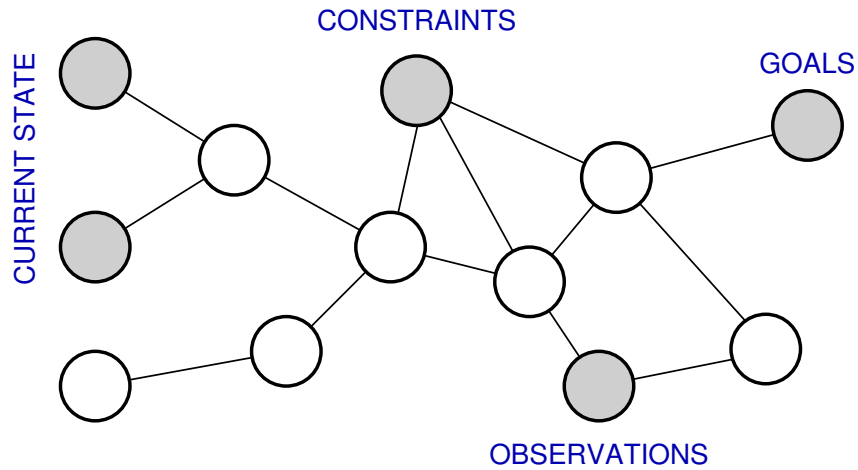
– *What is the posterior over actions?*

– Given a prior on Luis's finite life span, *What is the posterior over the time that Luis needed from S to G?*

- these are things we can compute!
  these Qs certainly are related to the problem of planning!

inference by planning – general idea

- *rephrase problem of planning as problem of inference!*
  – compute a posterior over actions, control signals, whole trajectories
  – conditioned on targets, goals, constraints
- works for arbitrary *networks* of goals, constraints, observations, etc.



→ planning on distributed representations!
  – on mixed discrete/continuous representations
  – contrasts classical notion of *state* as one big variable
    (value functions, spreading activation, RRTs, configuration space)
  – we know how to exploit structure with inference!

→ no distinction between sensor and motor, perception and action!

history: probabilistic inference & planning

– inference in influence diagrams: Cooper (1988), Shachter & Peot (1992)
  no sequential decisions, no discounting, finite horizon

– Dayan & Hinton (1997): neuroscience, immediate reward case only
– Attias (2003): "Planning by Probabilistic Inference"
  non-optimal policy, no arbitrary rewards, prefixed finite time
– Raiko (2005): optimistic inference control
– Bui et al. (2002): policy recognition/goal inference
– Verma & Rao (2006): maximal probable explanation (MPE)

– Toussaint & Storkey (ICML 2006): really solving MDPs
– de Freitas et al. (2007): MCMC methods for policy learning
– Toussaint & Goerick (IROS 2007): inference of robot trajectories
– Toussaint, Charlin, Poupart (UAI 2008): hierarchical POMDPs
– Vlassis & Toussaint (ICML 2009): model-free RL

mini survey: Toussaint (KI 09): *Probabilistic inference as a model of planned behavior*

Outline

- **Part 1: Introduction to probabilistic inference & learning**

  ...

- **Part 2: Planning by Inference**
  - typical use of inference in robotics: sensor $\rightarrow$ state
  - general idea of inference by planning
  - **Markov Decision Processes:**

    likelihood vs. expected rewards, EM learning for optimal policies, policies in structured processes, in relational domains, model-free Reinforcement Learning based on EM

  - Stochastic Optimal Control:

    likelihood vs. expected costs, Approximate Inference Control

- **Summary & further reading**

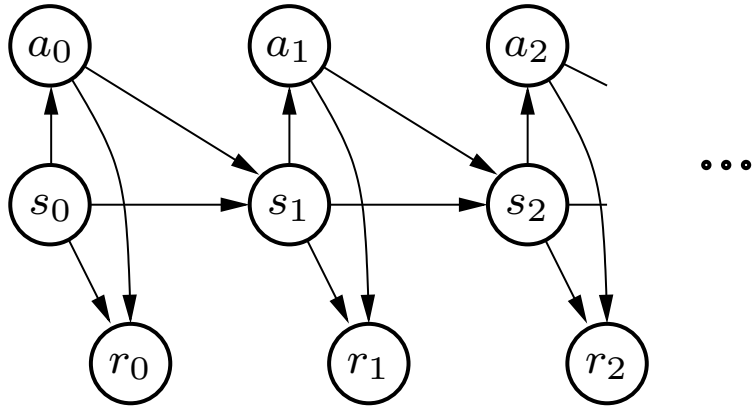
  ...

Markov Decision Processes revisited

- let's make the idea of planning by inference more explicit...

Markov Decision Processes

- Markov process on the random variables of states x_t, actions a_t, rewards r_t



$$P(s_{0:T}, a_{0:T}, r_{0:T}; \pi) =$$
$$P(s_0)P(a_0|s_0;\pi)P(r_0|a_0,s_0)\ \prod_{t=1}^{T} P(s_t|a_{t\text{-}1}, s_{t\text{-}1})P(a_t|s_t;\pi)P(r_t|a_t,s_t)$$

- the *world* defines: (stationarity: no explicit dependency on time)

  $P(s_0)$ initial state distribution

  $P(s_{t+1} \mid a_t, s_t)$ transition probabilties

  $P(r_t \mid a_t, s_t)$ reward probabilities

- the *agent* defines: (π is a *parameter* of the model)

  $P(a_t = a \mid s_t = x; \pi) \equiv \pi_{ax}$ action probabilities (policy)

MDP – definition of optimality

- there might be multiple ways to define what optimal behavior is!
  – standard consensus: (discounted) sum of expected rewards:
  given a policy (=model parameter) π, we define

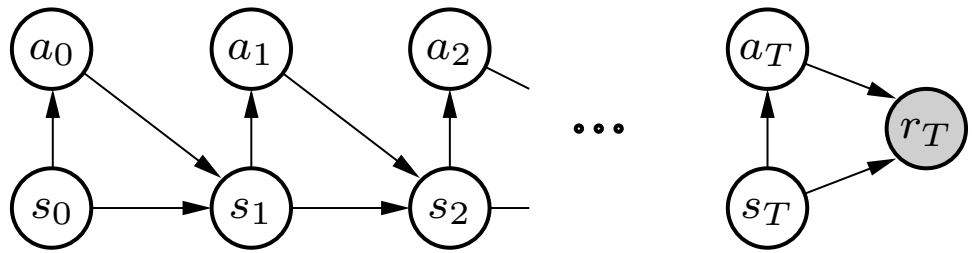
$$V^\pi = \mathrm{E}\{\sum_{t=0}^{\infty} \gamma^t R_t; \pi\}$$

  with discount factor $\gamma \in [0, 1]$

- note:
  – typical learning in graphical models:
  train parameters π to maximize the likelihood of data
  – Can expected reward be related to likelihood?
  – Can we use inference and learning in graphical models to find the optimal policy?

EM for planning I

- address a simplified case:
  – we care only for the reward r_T at finite time T
  – we assume binary rewards: dom$(r_T) = \{0, 1\}$



→ optimize the model parameters π to maximize the likelihood of observing reward $r_T = 1$:

- note: many more latent variables than observed variables:
  – observed variables (“data”): $r_T = 1$
  – latent (unobserved) variables: $s_{0:T}, a_{0:T}$

EM for planning II

- "observed data likelihood" (last lecture)

$$\exp \hat{L}(\pi) = P(r_T = 1; \pi)$$
$$= \sum_{a_{0:T}, s_{0:T}} P(r_T = 1, a_{0:T}, s_{0:T}; \pi) = \mathrm{E}\{r_T; \pi\}$$

- doing the summation analytically is intractable $\rightarrow$
- EM algorithm
  - E-step: *compute posterior over* $a_{0:T}, s_{0:T}$ *conditioned on "data"* $r_T = 1$

$$P(a_{0:T}, s_{0:T} \,|\, r_T = 1; \pi^{\text{old}})$$

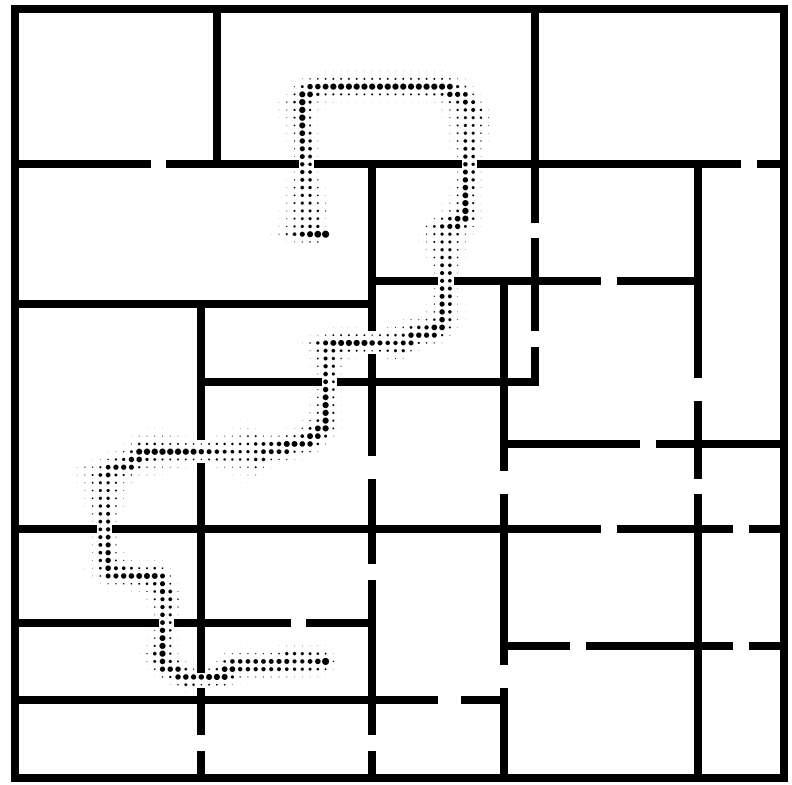
  - M-step: assign policy to new optimum of expected data log-likelihood

$$\pi^{\text{new}} = \operatorname*{argmax}_{\pi} Q(\pi, \pi^{\text{old}})$$

Example: MDP maze

- states: all locations
- actions: up, down, left, right
- transition probabilities:
  - $s \in$ wall: completely stuck
  - $s \notin$ wall: 90% make correct step,
    10% make random step

  $\rightarrow$ keep away from walls!



Interpretation of the E-step

- the E-step is really interesting:
  compute the posterior over states and actions conditioned on "data"
  $r_T = 1$

- we do as if $r_T = 1$ was data – although we (the agent) hasn't observed this data (yet)
  but we *imagine* we will observe it in the future

- internal simulation & mental imagery: we imagine to observe the event $r_T = 1$, and we "internally simulate" (compute the posterior over) the trajectory $a_{0:T}, s_{0:T}$ to get there

More general case

- what if the true rewards are not binary?
  – can rescale true rewards r_t such that $\mathrm{E}\{r_t \mid a_t, s_t\} \propto P(\hat{r}_t = 1 \mid a_t, s_t)$
  – caution: rescaling likelihoods changes convergence speed of EM!

- what if we care about all rewards $V^{\boldsymbol{\pi}} = \mathrm{E}\{\sum_{t=0}^{\infty} \gamma^t \, r_t; \pi\}$
  – we can introduce a “mixture model”

Value function as a mixture

- trivially:

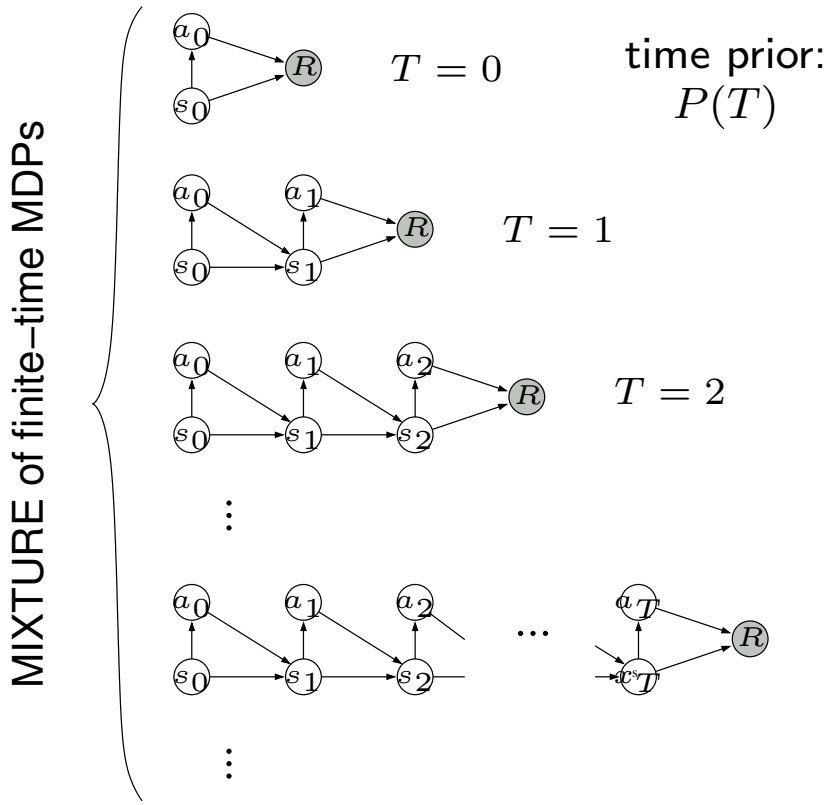
$$V^\pi = \mathrm{E}\{\sum_{t=0}^{\infty} \gamma^t r_t; \pi\}$$
$$= \sum_{T=0}^{\infty} \gamma^T \, \mathrm{E}\{r_T; \pi\}$$

- interpretations:
  – first line: one agent collects rewards in every time step
  – last line: mixture: agent only collects r_T, but at randomized time T with $P(T) \propto g^T$
    (heaven or hell interpretation...)

→ we may interpret T as a random variable with prior $P(T) \propto g^T$

Mixture of finite-time MDPs

- illustration of that mixture:



Mixture of finite-time MDPs

- joint distribution:

$$P(R{=}1, a_{0:T}, s_{0:T}, T; \pi) = P(T)\ P(R{=}1, a_{0:T}, s_{0:T} | T; \pi)$$

- if we choose $P(T)$ geometric, $P(T) = (1-\gamma)\ \gamma^T$,

$$\begin{aligned}
\exp \hat{L}(\pi) &= P(R{=}1; \pi) \\
&= \sum_{a_{0:T}, s_{0:T}, T} P(T)\ P(R{=}1, a_{0:T}, s_{0:T} | T; \pi) \\
&= \sum_T P(T)\ \mathrm{E}\{r_T; \pi\} = (1-\gamma) \sum_T \gamma^T\ \mathrm{E}\{r_T; \pi\} = (1-\gamma) V^{\boldsymbol{\pi}}
\end{aligned}$$

$\Rightarrow$ maximization of likelihood $R{=}1$

$\Longleftrightarrow$ maximization of expected discounted future return

$\Rightarrow$ we can compute optimal (in the traditional definition) policies using Expectation Maximization in $P(R{=}1, a_{0:T}, s_{0:T}, T; \pi)$

Expectation Maximization for policy optimization

- *policy optimization is framed as maximizing likelihood* $P(R=1)$

- EM algorithm has two stages:

  E-step: inference:
  compute posterior over actions condition on "you will see what you want to see" (rewards)

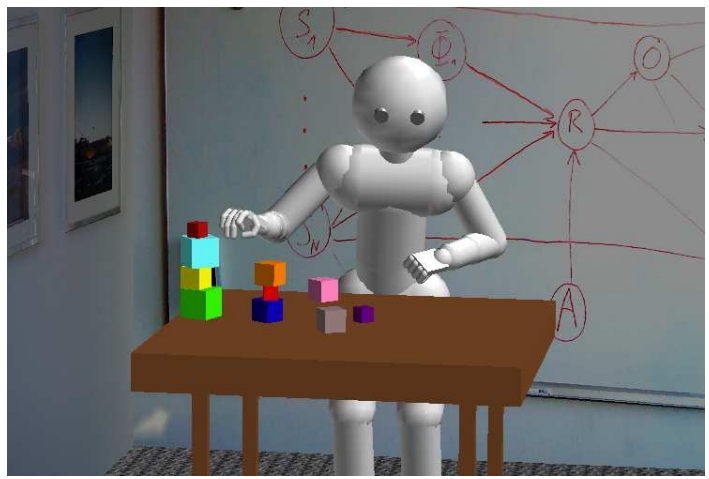
  M-step: behavior update:
  update your policy accordingly

Extensions & further reading

- discrete and continuous MDPs
  Toussaint, Storkey (ICML 2006): *Probabilistic inference for solving discrete and continuous state Markov Decision Processes*
- partial observability (hierarchical POMDPs)
  Toussaint, Charlin & Poupart: *Hierarchical POMDP Controller Optimization by Likelihood Maximization*
- *model-free* Reinforcement Learning
  Vlassis, Toussaint (ICML 2009): *Model-Free Reinforcement Learning as Mixture Learning*
  Vlassis et al (Autonomous Robots): *Robot Reinforcement Learning by a Monte-Carlo EM algorithm*
- Nando de Freitas & Matt Hoffman (UBC): Monte Carlo (sampling methods for the EM)
- David Wingate & Josh Tennenbaum (MIT): non-parameteric controllers
- Matthew Botvinick at al. (Princeton): *Goal-directed decision making in prefrontal cortex: A computational framework* (NIPS 2008)

Reasoning in complex uncertain environments

- real world is *composed of objects* (...exponential)
  – classical AI has efficient representations, but deterministic
  $\rightarrow$ marriage between logic and probabilistic representations



- the model (rules) is *learnt from random experiments*
  T Lang, M Toussaint (ICML 2009): *Approximate Inference for Planning in Stochastic Relational Worlds*
  T Lang, M Toussaint (ECML 2009): *Iterated Relevance Grounding for Planning in Relational Domains*

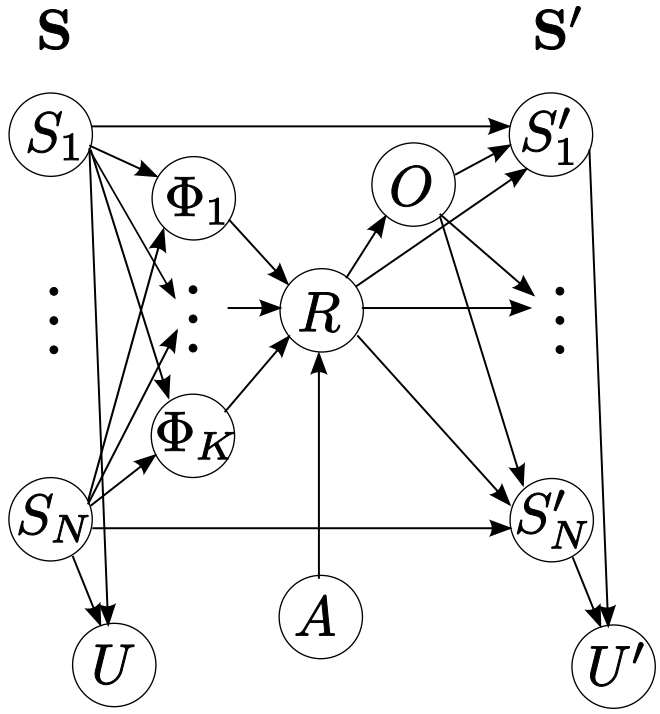
Reasoning in complex uncertain environments

- **Learning:**
  compact indeterministic rules (Zettlemoyer, Pasula, & Kaelbling 2005)

$$grab(X): \quad on(X,Y),\ block(Y),\ table(Z)$$
$$\rightarrow \begin{cases} 0.7 & : \quad inhand(X),\ \neg on(X,Y) \\ 0.2 & : \quad on(X,Z),\ \neg on(X,Y) \\ 0.1 & : \quad \text{noise} \end{cases}$$

- **Planning:**
  – so far: sparse sampling trees (horribly inefficient)
  – here: approximate inference & relevence grounding

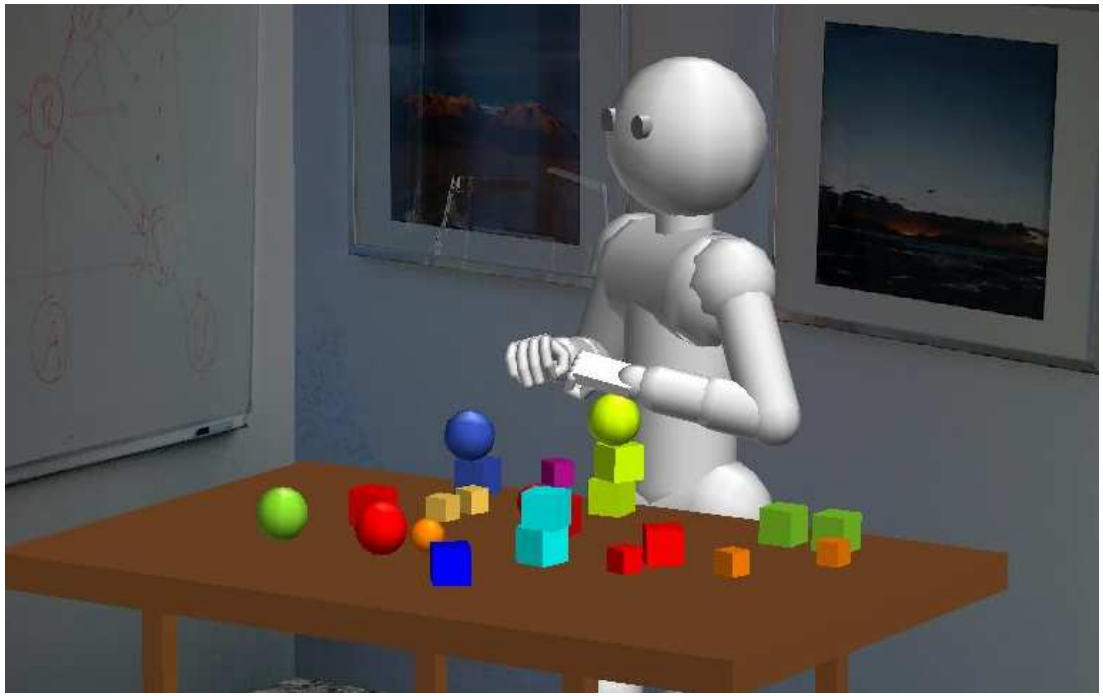
Translation to a Dynamic Bayesian Network



- we call this *grounding*
- ground the model *only for objects that seem relevant for the task*
  – form of working memory / working model
- theory exactly as before:

  planning $\leftrightarrow$ likelihood maximization in this DBN

results



Desktop Clearance task: Robot has to clear up the desktop by piling objects of the same size and color

Obj.	Config	Reward	Time
30	random $N_r = 1$	3.09 $\pm$ 1.09	**16.06** $\pm$ **0.41**
	random $N_r = 5$	0.52 $\pm$ 0.26	80.11 $\pm$ 1.47
	random $N_r = 10$	0.02 $\pm$ 0.02	162.17 $\pm$ 6.34
	heuristic $N_r = 1$	42.31 $\pm$ 3.06	16.47 $\pm$ 0.43
	heuristic $N_r = 5$	**59.79** $\pm$ **3.33**	81.34 $\pm$ 3.87
	heuristic $N_r = 10$	53.29 $\pm$ 3.50	159.01 $\pm$ 3.39
	full-grounding	22.42 $\pm$ 2.14	5893.81 $\pm$ 1006.56

Outline

- **Part 1: Introduction to probabilistic inference & learning**

  ...

- **Part 2: Planning by Inference**
  - typical use of inference in robotics: sensor $\rightarrow$ state
  - general idea of inference by planning
  - Markov Decision Processes:

    likelihood vs. expected rewards, EM learning for optimal policies, policies in structured processes, in relational domains, model-free Reinforcement Learning based on EM

  - **Stochastic Optimal Control:**

    likelihood vs. expected costs, Approximate Inference Control

- **Summary & further reading**

  ...

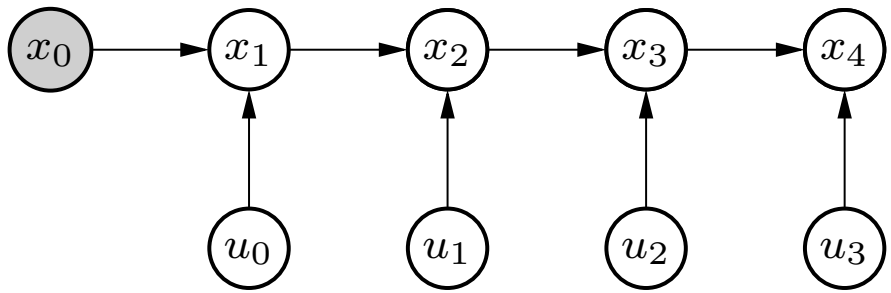
Motivation

that’s how robots should move...

Stochastic Optimal Control

- discrete time stochastic controlled process:

$$x_{t+1} = f_t(x_t, u_t) + \xi \ , \qquad \xi \sim \mathcal{N}(0, Q_t)$$



x_t	state at time t
u_t	control signal at time t
f	system dynamics
ξ	Gaussian noise

SOC – definition of optimality

- again, there might be alternative ways to define optimal control...
  – standard consensus:
- costs for a given state-control sequence $x_{0:T}, u_{0:T}$:

$$C(x_{0:T}, u_{0:T}) = \sum_{t=0}^{T} c_t(x_t, u_t)$$

- problem: find a control policy $\pi_t^* : x_t \mapsto u_t$ that minimizes expected cost
- optimal value function $J_t(x) = \min_{u_{t:T}} \mathrm{E}_{x_{t:T}|u_{t:T}, x_t=x}\{\sum_{k=t}^{T} c_k(x_k, u_k)\}$
- Bellman equation $J_t(x) = \min_u \left[c_t(x, u) + \int_{x'} P(x' \mid u, x)\ J_{t+1}(y)\right]$

Classical SOC methods

- general problem very hard

- classical variational (perturbative) solution:
  1. find optimal trajectory of *noise-free* system
  2. solve approximate stochastic system around this optimal trajectory

- 1. stage:
  – e.g., gradient on spline encoding
  – e.g., Sequential Quadratic Programming

- 2. stage:
  – what approximation should we assume around the optimal trajectory?
  $\rightarrow$ LQG: Linear Quadratic Gaussian

Classical SOC methods – LQG approximation

- assume locally: Linear dynamics, Quadratic costs, Gaussian noise

$$P(x_{t+1} \,|\, x_t, u_t) = \mathcal{N}(x_{t+1} \,|\, A_t x_t + a_t + B_t u_t, Q_t) \ ,$$
$$c_t(x_t, u_t) = x_t^\top R_t x_t - 2 r_t^\top x_t + u_t^\top H_t u_t \ .$$

$\Rightarrow$ things become really simple:

– all random variables are joint Gaussian, with pair-wise correlations

– the optimal value function is always quadratic

$$J_{t+1}(x) = x^\top V_{t+1} x - 2 v_{t+1}^\top x$$

– Ricatti equation gives exact solution of Bellman equation:

$$V_t = R + A^\top V_{t+1} A - K V_{t+1} A$$
$$v_t = r + A^\top (v_{t+1} - V_{t+1} a) - K (v_{t+1} - V_{t+1} a)$$
$$K := A^\top V_{t+1}^\top (V_{t+1} + B^{-\top} H B^{-1})^{-1}$$

– perfectly analogous to Kalman filtering, but bwd *(Kalman duality)*

Classical SOC methods

- classical variational solution:
  1. find optimal trajectory of *noise-free* system
  2. solve approximate stochastic system around this optimal trajectory

- 1. stage:
  – Sequential Quadratic Programming
  $\rightarrow$ each step of SQP can be solved exactly by Ricatti equation

- 2. stage:
  – LQG approximate around the optimal trajectory
  $\rightarrow$ Ricatti equation gives optimal (local) controller

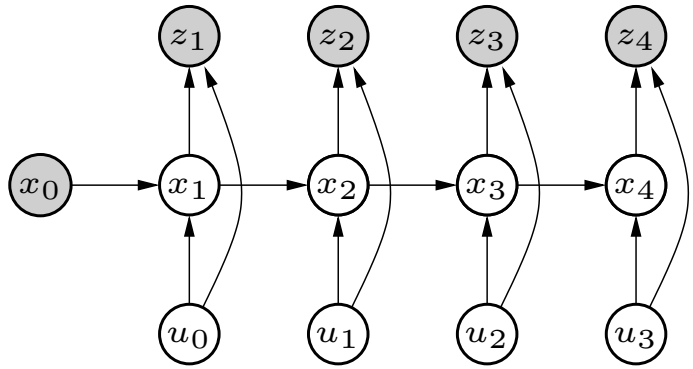
  (also called “iLQG” iterative LQG)

Probabilistic inference approach

- introduce an extra RV z_t to represent costs

$$P(x_{t+1} \,|\, u_t, x_t) = \mathcal{N}(x_{t+1} \,|\, f_t(x_t, u_t), Q_t)$$

$$P(z_t = 1 \,|\, x_t, u_t) = \exp\{-c_t(x_t, u_t)\}$$



- note:
  – in MDPs we introduced $P(R = 1|...)$ proportional to rewards
  – now we introduce $P(z_t = 1|...)$ in *in neg-log space!*

details:
Toussaint (ICML 2009): *Approximate Inference for Robot Trajectory Optimization*

Probabilistic inference approach – properties

- trivially: for a *specific* trajectory

$$\log P(z_{0:T}=1 \mid x_{0:T}, u_{0:T}) = \log \prod_{t=0}^{T} P(z_t=1 \mid x_t, u_t)$$

$$= -\sum_{t=0}^{T} c_t(x_t, u_t) = -C(x_{0:T}, u_{0:T})$$

  maximum-likelihood trajectory = optimal trajectory

- but:

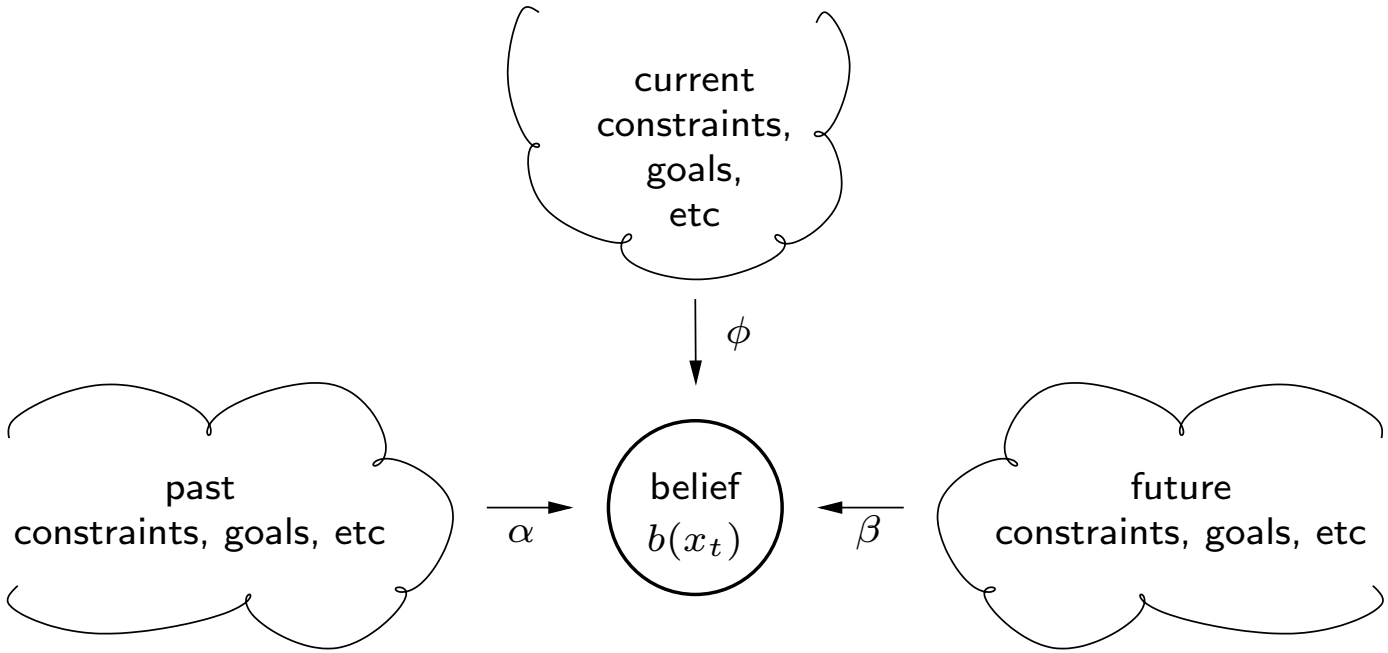
$$\log P(z_{0:T}=1) \;\geq\; -\mathrm{E}_{u_{0:T}, x_{0:T}}\{C(u_{0:T}, x_{0:T})\}$$

  likelihood maximization $\neq$ expected cost minimization

- nevertheless:
  approximate inference (Gaussian beliefs) has same fixed point as classical variational solution (e.g., iLQG)

How to do inference

- as we learnt for time series models:



loop back-and-forth over t {

update messages μ_S, μ_V, μ_R until $b(x_t)$ converges

}

How to do inference

- LQG case:

$$P(z_t = 1 \,|\, x_t) \propto \mathcal{N}[x_t \,|\, r_t, R_t]$$
$$P(u_t) = \mathcal{N}[u_t \,|\, 0, H]$$
$$P(x_{t+1} \,|\, x_t, u_t) = \mathcal{N}(x_{t+1} \,|\, Ax_t + a + Bu_t, Q)$$

  $\rightarrow$ *all RVs are joint Gaussian with pair-wise correlations*

$\Rightarrow$ all messages can be computed exactly as Gaussians
  – converges after single fwd-bwd
  – backward messages correspond directly to Ricatti equation

- in LQG case $\rightarrow$ same solution as classical Ricatti equation

$\rightarrow$ proof: local fixed point = optimal w.r.t. local LQG approximation

- core open question: *how generalize to non-LQG systems?*

How to do inference

- non-LQG case:
  – *use approximate inference methods!*

- here:
  – compute Gaussian approximate messages
    (using local linearizations, extended Kalman filter)

  – locally update these messages until belief converges
  $\rightarrow$ implicit focus on "difficult parts" (exact update equations $\rightarrow$ paper)

Approximate Inference COntrol – summary

- We formulated a probabilistic model such that:
  1. maximum-likelihood trajectory = optimal trajectory
  2. in the LQG case:
     – everything is joint-Gaussian $\longrightarrow$ exact inference
     – reproduces the classical stochastic optimal control solution
     (bwd messages $\Longleftrightarrow$ Ricatti eqn)
  3. non-LQG systems:
     – approximate inference
     – alternative to approximate SOC solvers

- using Gaussian messages representations:
  – if it converges, then to local LQG optimal controller

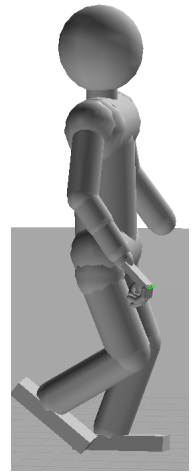
- message passing algorithms are *local*:
  – implicitly focus on "difficult parts"

Approximate Inference COntrol – readings

- more details in:
  Toussaint (ICML 2009): *Approximate Inference for Robot Trajectory Optimization*
- further reading: relation between inference and SOC: Kalman duality
  Todorov (2008): *"General duality between optimal control and estimation"*
  Kappen, Gomez, Opper (2009): *"Optimal control as graphical model inference problem"*

  (generalize the Kalman duality to special case non-LQG systems)

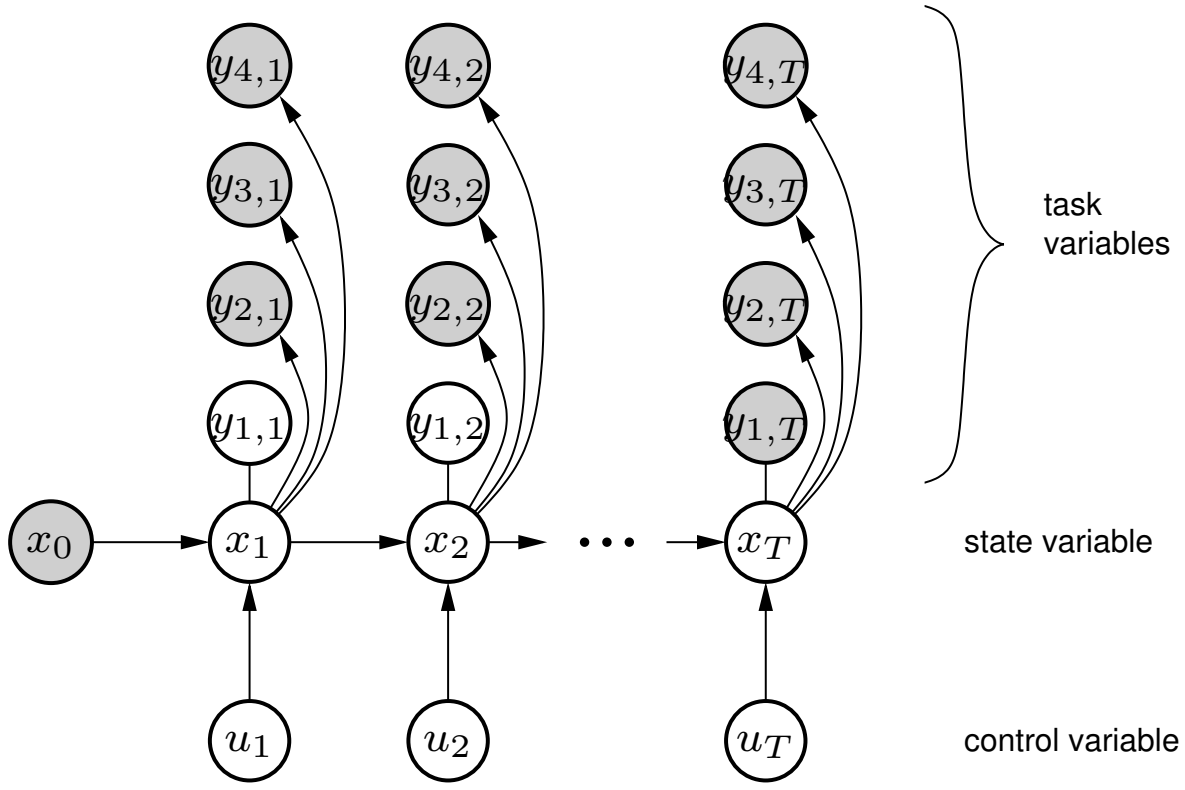
Example



- in typical robotics scenarios we have
  multiple *task - variables*
- assume we have a humanoid
  – x_t = posture of humanoid at time t
  – 1. task variable: $y_1 \in \mathbb{R}^3$ is the robot's finger tip position
  – 2. task variable: $y_2 \in \mathbb{R}^2$ is the robot's balance (horizontal offset)
  – 3. task variable: $y_3 \in \mathbb{R}$ measures collision/proximity
- for each task variable, we have
  – the kinematic function $\phi_i : x \mapsto y_i$, its Jacobian $J_i(x)$
  – desired values $y^*_{i,t}$ and variances $C_{i,t}$ for each time step t
  (corresponds to quadratic cost terms – could be more general..)

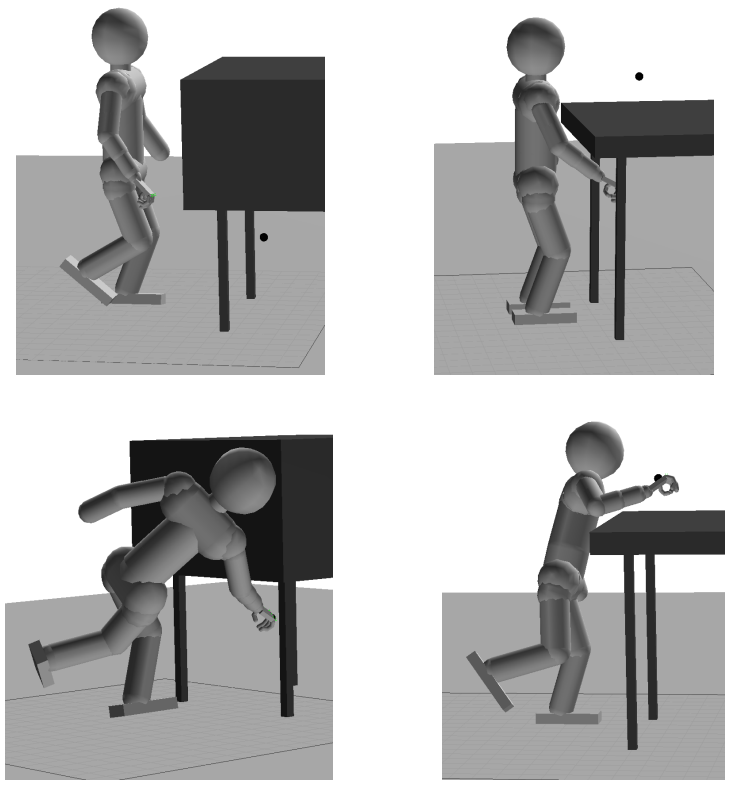
Example

- this corresponds to the model



$$\forall_i : \; P(y_{i,t} \mid x_t) = \mathcal{N}(y_{i,t} \mid \phi_i(x_t), C_{i,t})$$

Example

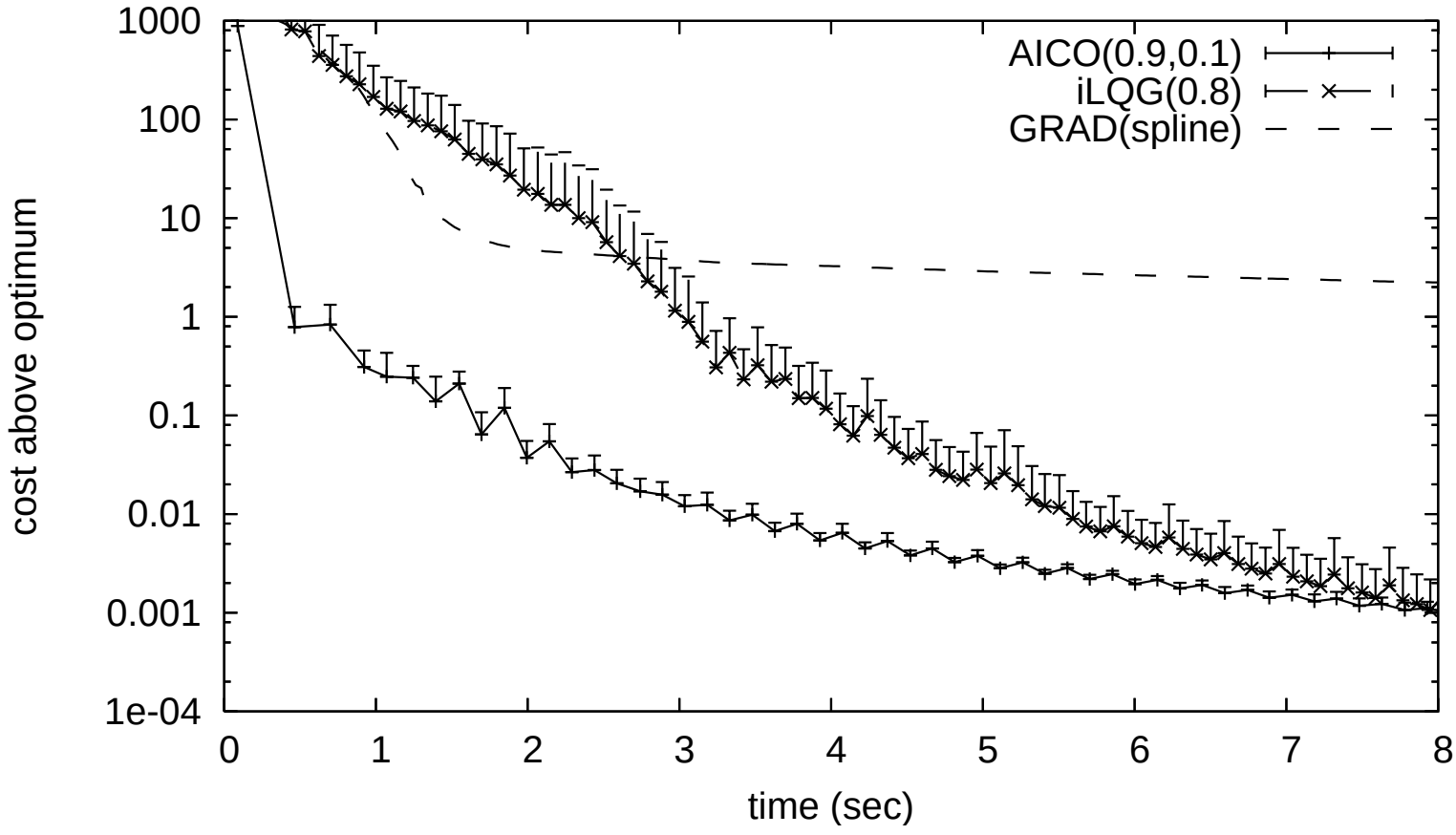


- $\sim$30 DoF robot, task variables:
  - $y_1 \in \mathbb{R}^3$ is the robot's finger tip position
  - $y_2 \in \mathbb{R}^2$ is the robot's balance (horizontal offset to support)
  - $y_3 \in \mathbb{R}$ measures collision/proximity

- cost parameters:
  (a) $C_{1,T} = 10^{-5}$, $C_{1,0:T\text{-}1} = 10^4$, $C_{2,0:T} = C_{3,0:T} = 10^{-5}$
  (b) $C_{1,T} = 10^{-2}$

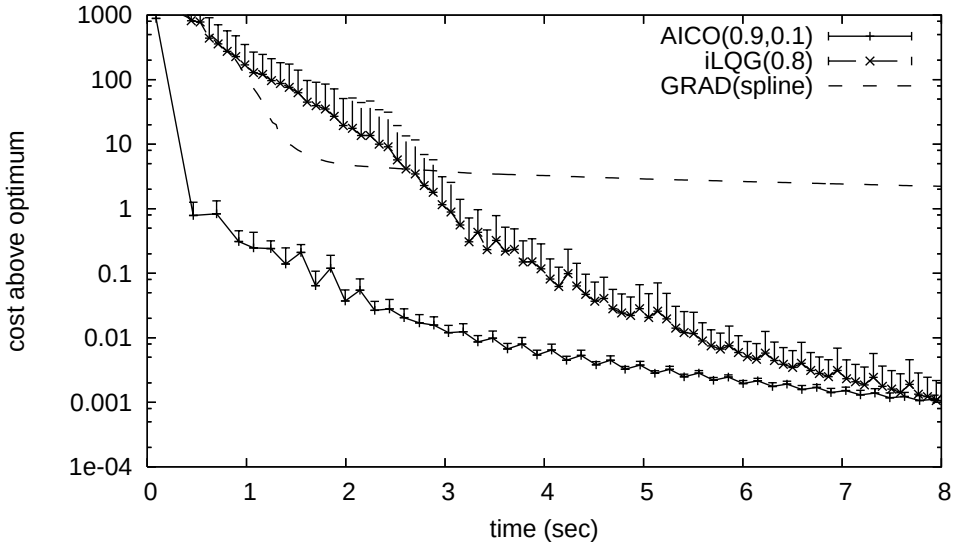
Example

1(a)

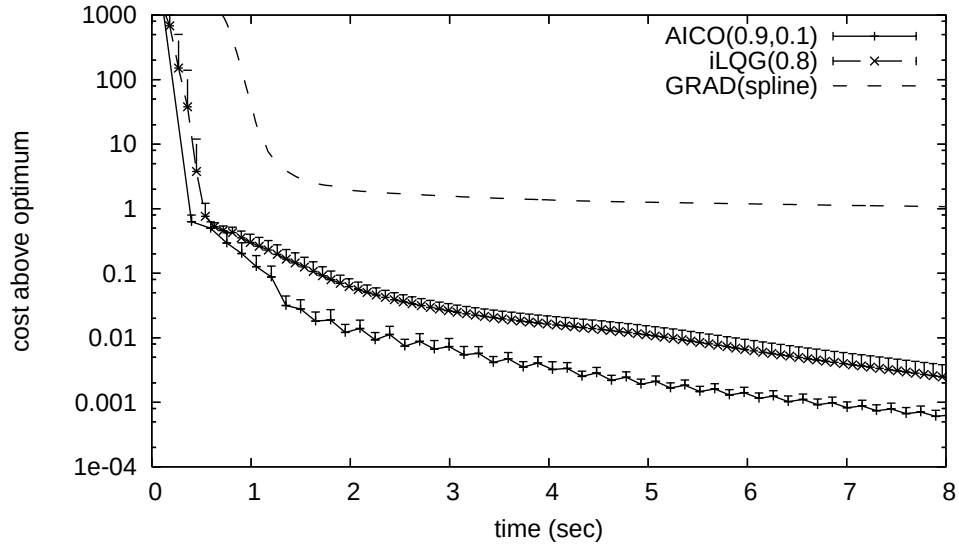

AICO(0.9,0.1)
iLQG(0.8)
GRAD(spline)
cost above optimum
time (sec)
1000
100
10
1
0.1
0.01
0.001
1e-04
0
1
2
3
4
5
6
7
8

Example

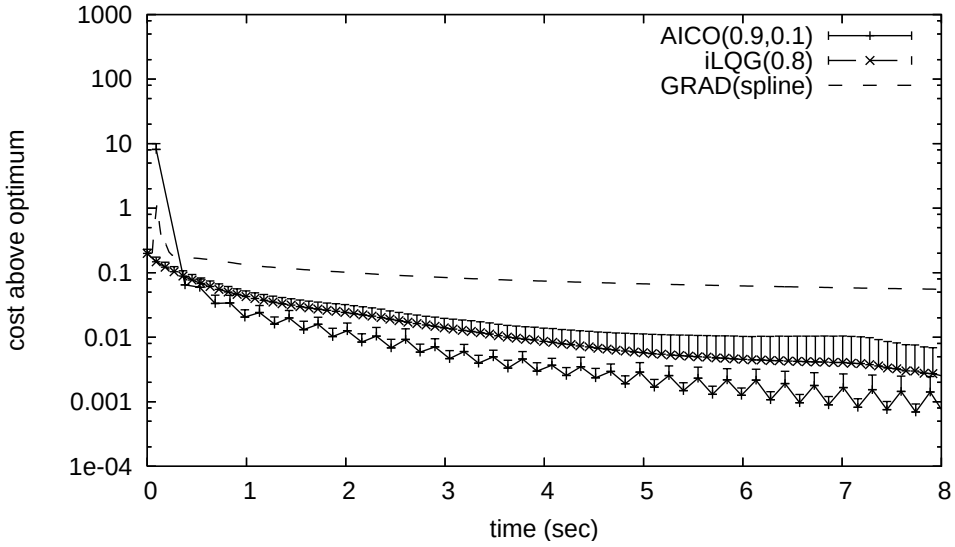
1(a) (scenario 1 – strict constraints)



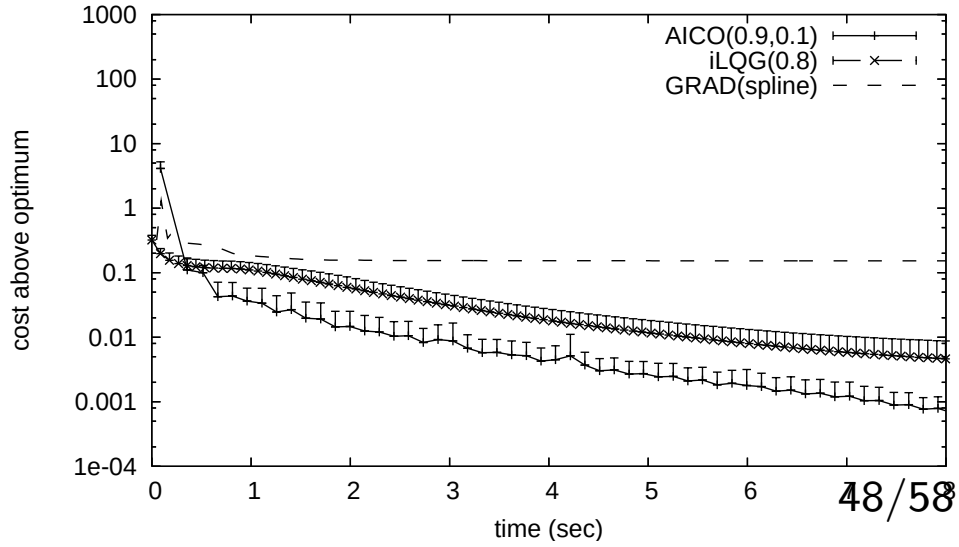
2(a) (scenario 2 – strict constraints)



1(b) (scenario 1 – very soft constraints)



2(b) (scenario 2 – very soft constraints)



Example 2

- Schunk LWA
- haptics (Schunk SDH)
- vision (Bumblebee stereo camera)
- depth (laser range finder)

- 14 joints, dynamic $\rightarrow x_t \in \mathbb{R}^{28}$

AICO – concluding remarks

- bottom line:

  motion generation as a problem of probabilistic inference

  – can also be applied on decomposed representations
  Toussaint & Goerick (IROS 2007): *Probabilistic inference for structured planning in robotics*

  – builds a bridge between ML methods and robotics

- much room for improvements:
  – proper Expectation Propagation updates?
  – truncated Gaussian messages?
  – other approximate inference methods
  – motion priors?

Outline

- **Part 1: Introduction to probabilistic inference & learning**

  ...

- **Part 2: Planning by Inference**

  ...

- **Summary & further reading**
  – food for thought
  – brief summary
  – further reading

Seymour Papert misses ”Big Ideas” of the good old days in AI.

Andrew Ng: “I didn’t come into AI just to do statistics.”

- disintegration of AI?
  - Information Theory, Statistical Learning Theory, Machine Learning
  - Computer vision
  - Speech analysis, Natural Language Processing
  - Reinforcement Learning, Decision Theory
  - Logic, symbolic reasoning
  - Robot control, planning, design, sensor processing

$\Rightarrow$ great progress in each subfields, but the integrated picture?

– e.g., *What are computational principles across domains?*

In this lecture...

- probabilistic inference as generic information processing tool
  – for sensor processing, learning, planning, and acting
  – on low-level (motion) as well a high-level (rules) representations
  – on distributed, structured, hierarchical representations
  – on rule-based, symbolic, and continuous representations

- build a bridge between ML and robotics:
  – representations in ML $\leftrightarrow$ representations for behavior & robotics
  – inference and learning methods from ML $\rightarrow$ RL & robotics

- in the following:
  – pointers to literature for broader perspective...

Further reading

- if you're interested in neuroscientific views on inference

  Bayesian Brain: Probabilistic Approaches to Neural Coding K. Doya, S. Ishii, A. Pouget, RPN. Rao (editors), MIT Press (2007)

  The Neurodynamics of Belief Propagation on Binary Markov Random Fields T. Ott, R. Stoop (NIPS 2006)

→ Bayesian information processing is a possible *abstraction* of neuronal functions

(not necessarily a model of *how* neurons work, but what their *function* from an information processing point of view is.)

Further reading

- if you're interested in cognitive science point of view

  Rick Grush (Behavioral and Brain Sciences, 2004): *The emulation theory of representation: motor control, imagery, and perception.*

  G. Hesslow (Trends in Cog Sciences, 2002):*Conscious thought as simulation of behaviour and perception.*

  – key ideas: internal simulation, mental imagery

→ could ML contribute more rigurous mathematical framework?

Further reading

- interested in neuroscience & planning by inference?:
- Matthew Botvinick at al. (Princeton): *Goal-directed decision making in prefrontal cortex: A computational framework* (NIPS 2008)

  > We take three empirically motivated points as founding premises: (1) Neurons in dorsolateral prefrontal cortex represent action policies, (2) Neurons in orbitofrontal cortex represent rewards, and (3) Neural computation, across domains, can be appropriately understood as performing structured probabilistic inference. [...]

- Johnson & Redish (J o Neuroscience, 2007): *Neural ensembles in CA3 transiently encode paths forward ...*

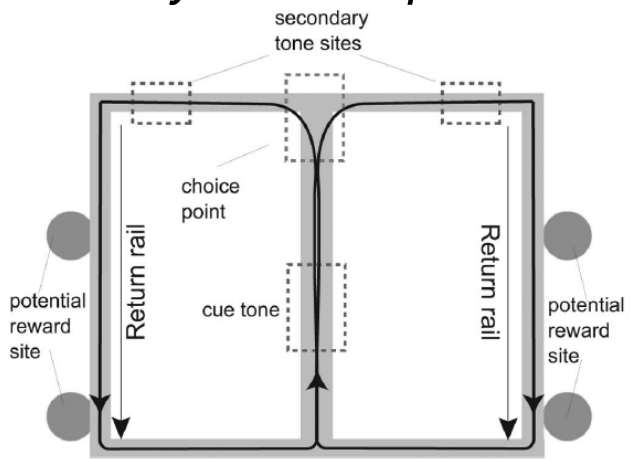
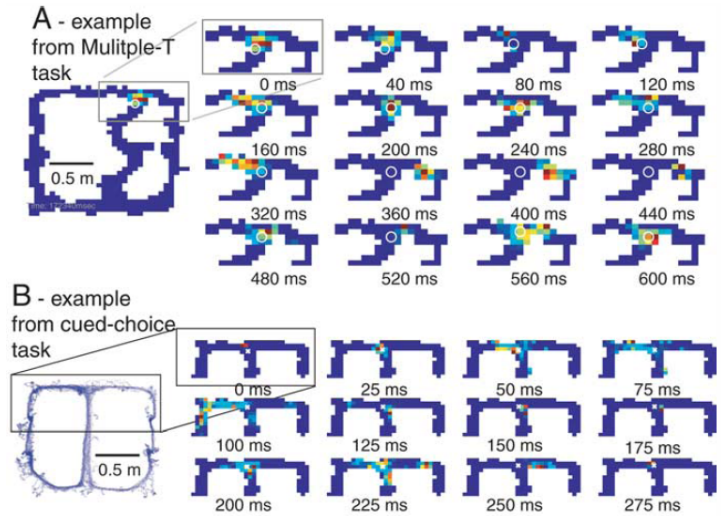


Figure 2. The cued-choice maze. The task consists of a single T turn with food reward available at two sites on each return rail. On each lap, a cue tone on the center stem (red dashed box) signaled whether the left or right arm would be rewarded. If and only if the rat made the correct turn, a matching tone would play at the T arm (blue dashed boxes).



Further reading

- limitations of main-stream ML methods and graphical models...

  for instance, graphical models often rely on hand-coded representations
  – autonomous learning of graph representations from data?
  – deep representations?
  – incremental learning of deeper and deeper hierarchies?
  – same for *behavior*?

- interesting readings:
  Thomas G. Dietterich et al.: *Structured machine learning: the next ten years*

  Bengio, Yoshua and LeCun, Yann: *Scaling learning algorithms towards AI*

  Rodney Douglas et al.: *Future Challenges for the Science and Engineering of Learning*

thanks for your attention!