236609 - AI and Robotics

Planning Under Partial Observability

Sarah Keren

The Taub Faculty of Computer Science
Technion - Israel Institute of Technology

Resources

- Caelan R. Garrett, Chris Paxton, Tomás Lozano-Pérez, Leslie P. Kaelbling, Dieter Fox. Online Replanning in Belief Space for Partially Observable Task and Motion Problems, IEEE International Conference on Robotics and Automation (ICRA), 2020.
  https://youtu.be/IOtrO29DFUg
  https://arxiv.org/abs/1911.04577
- ICAPS 2014: Leslie Kaelbling on "Integrated Task and Motion Planning in Belief Space"
  https://youtu.be/6ks24mIRj-Y
- ICAPS 2014: Tutorial by Siddharth Srivastava on "Task and Motion Planning for Robots in the real world"
  https://youtu.be/wRZ2yqRrPiY

Resources

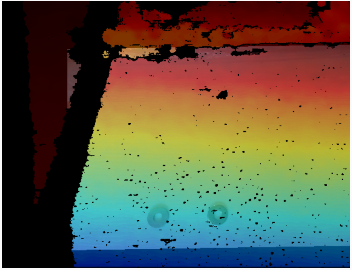
- 3rd ICAPS Summer School on Cognitive Robotics talk by Caelan Garrett. *https://sites.usc.edu/cognitive-robotics/* *http://web.mit.edu/caelan/www/presentations/CRSS19.pdf* *https://youtu.be/JNOk1rylDpU*
- ICAPS 2019: Tutorial on Integrated Task and Motion Planning by Malik Ghallab, Felix Ingrand, Rachid Alami, Thierry Simeon *https://youtu.be/5iNAjwoYMrQ*

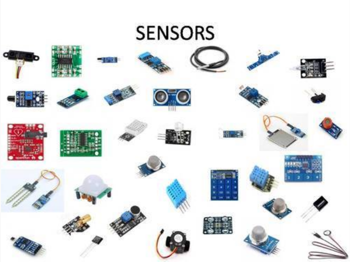
Example

Partial Observability

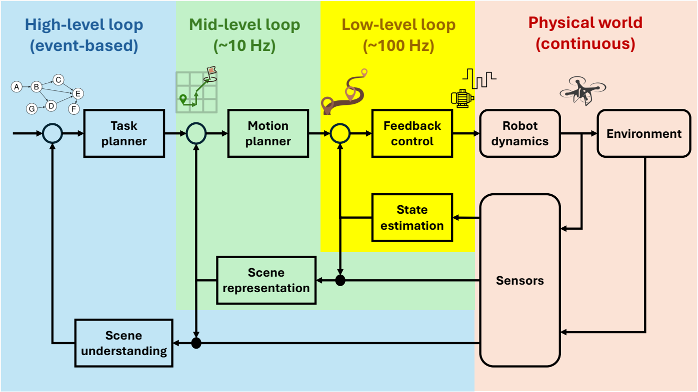
The agent does not know

- exactly where it is
- where relevant objects are
- where there are obstacles

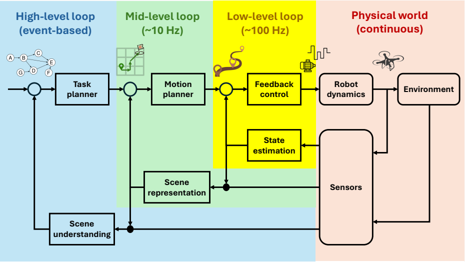




Layered Control Architecture (LCA)



Layered Control Architecture (LCA)



Making Decisions with Partial Information

Markov Decision Process

A **Markov Decision Process** (MDP) is a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$ where

- $\mathcal{S}$ is a finite set of states defined via a set of random variables $\mathcal{X} = X_1, \ldots, X_n$
- $\mathcal{A}$ is a finite set of actions
- $\mathcal{P}$ is a state transition probability matrix $\mathcal{P}^a_{s,s'} = \mathcal{P}[S_{t+1} = s' | S_t = s, A_t = a]$
- $\mathcal{R}$ is a reward function, $\mathcal{R}^a_s = \mathbb{E}[R_{t+1} | S_t = s, A_t = a]$, and
- **optional:** γ is a discount factor $\gamma \in [0, 1]$ that is used to favor immediate rewards over future rewards.

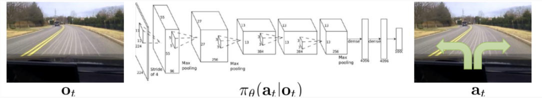
The Markov property: "The future is independent of the past given the present".

Extensions: Infinite and continuous MDPs, partially observable MDPs, undiscounted, average reward MDPs. etc.

Accounting for Partial Information

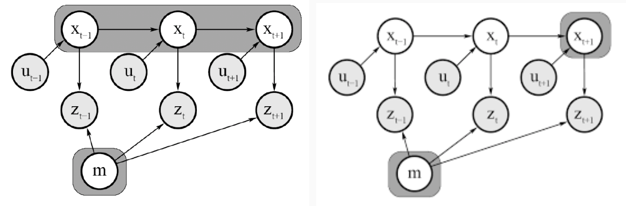
- Can we use a **Markov Decision Process**(MDP) $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$ to account for partially observable environments ?

Sometimes, yes.



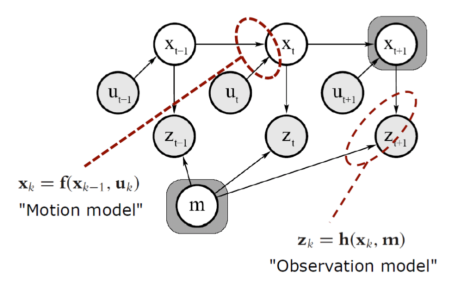
Sometimes, an MDP is not enough.

Probabilistic sensors and dynamics - graphical model



Circles represent variables. Arrows represent dependencies (influences).

Probabilistic sensors and dynamics - graphical model



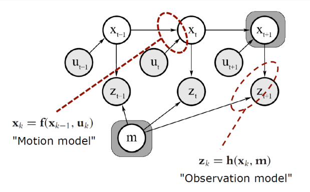
Motion model: how does the robot move ?

$$P(x_t | x_{t-1}, \mathbf{u}_t)$$

Observation model: how to interpret the observations ?

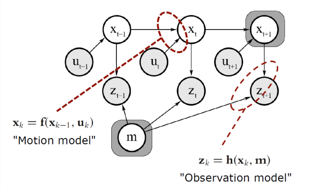
$$P(o_t | x_{t-1}, \mathbf{m})$$

where $\mathbf{m}$ is the representation of the environment, e.g., map.



Which models this reminds us of ?

Probabilistic sensors and dynamics - graphical model

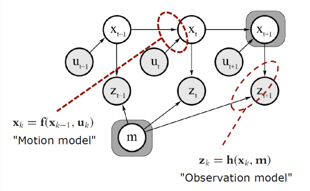


- Hidden Markov Models (HMM)
- Markov Decision Process (MDP)
- Partially observable Markov decision process (POMDP)

All based on the **Markov property** i.e., the future is independent of the past given the present.

Which model is most suitable?

Probabilistic sensors and dynamics - graphical model



- Hidden Markov Models (HMM)
- Markov Decision Process (MDP)
- Partially observable Markov decision process (POMDP)

All based on the **Markov property** i.e., the future is independent of the past given the present.

Which model is most suitable? It depends!

Algorithm Bayes_filter($bel(x_{t-1}), u_t, z_t$)**:**

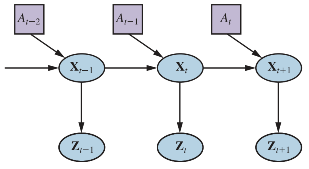
for all x_t do

$\overline{bel}(x_t) = \int p(x_t \mid u_t, x_{t-1})\, bel(x_{t-1})\, dx$

$bel(x_t) = \eta\, p(z_t \mid x_t)\, \overline{bel}(x_t)$

endfor

return $bel(x_t)$

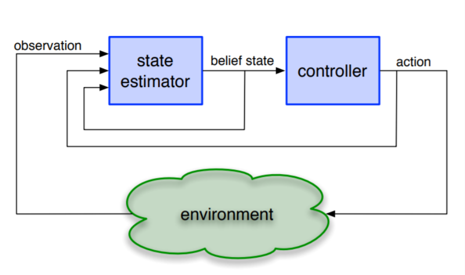


Where is the Markovian assumption used here?

*From Probabilistic Robotics by S. Thrun(2002)

Planning in Belief Space

- A **belief** is a probability distribution over the possible world states such that $\beta(s)$ stands for the probability that s is the true world state.
- In partially observable domains, we may have a **sensor model / state estimator** represented as a mapping function from what is observed to the actual world state.



From Kaelbling, L. P., and T. Lozano-Perez. "Integrated Task and Motion Planning in Belief Space" 2013 *https://dspace.mit.edu/bitstream/handle/1721.1/87038/Kaelbling_Integrated%20task.pdf?sequence=1&isAllowed=y*

Partially Observable Markov Decision Process (POMDP)

A **Partially Observable Markov Decision Process**(POMDP) is a tuple

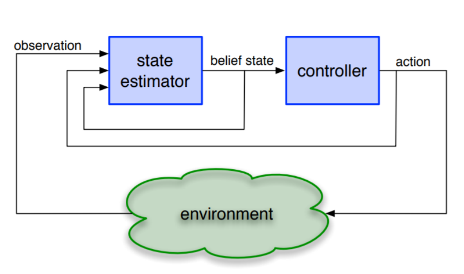
$\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma, \Omega, \mathcal{O}, \beta_0 \rangle$ where

- $\mathcal{S}$, $\mathcal{A}$, $\mathcal{P}$, $\mathcal{R}$ and γ are as for an MDP.
- Ω is a set of observations (observation tokens),
- $\mathcal{O}$ is a sensor function specifying the conditional observation probabilities $\mathcal{O}^o_{s,a} = \mathbb{P}[O_{t+1} = o | S_t = s, A_t = a]$ of receiving observation token $o \in \mathcal{O}$ in state s after applying a [1].
- β_0 the initial belief: a probability distribution over the states such that $belief_0(s)$ stands for the probability of s being the true initial state.

[1]alternatively: $\mathcal{O}^o_s = \mathbb{P}[O_t = o | S_t = s]$

Two key challenges when planning in belief space:

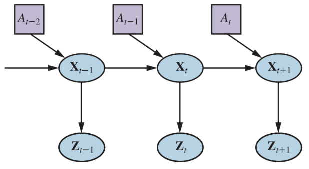
- Belief tracking - what is the state of the world ?
- Policy computation - what is the best action to perform ?



Pineau, Nicholas and Thrun. "A hierarchical approach to POMDP planning and execution." 2001. https://www.cs.mcgill.ca/ jpineau/files/jpineau-icml01.pdf

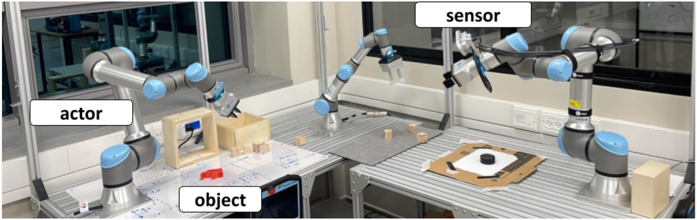
POMDPs and Bayes' Filter*


```
Algorithm Bayes_filter(bel(x_{t-1}), u_t, z_t):
    for all x_t do
        \overline{bel}(x_t) = ∫ p(x_t | u_t, x_{t-1}) bel(x_{t-1}) dx
        bel(x_t) = η p(z_t | x_t) \overline{bel}(x_t)
    endfor
    return bel(x_t)
```



How is this related to a **Partially Observable Markov Decision Process**(POMDP) $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma, \Omega, \mathcal{O}, \beta_0 \rangle$?

CLAIR lab Example

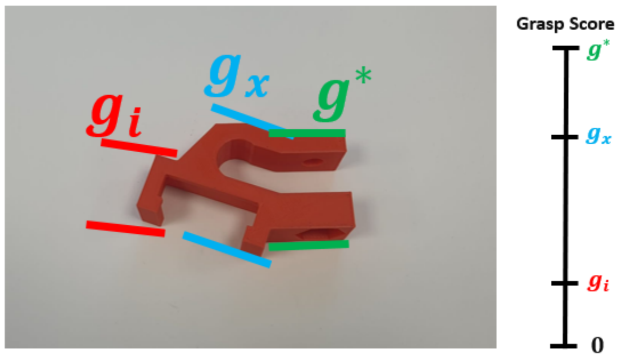


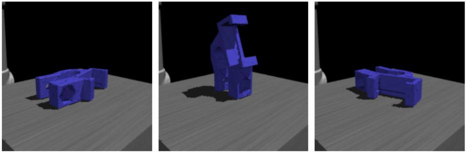
Value of Assistance for Grasping

Mohammad Masarwy, Yuval Goshen, David Dovrat, Sarah Keren

https://arxiv.org/abs/2310.14402

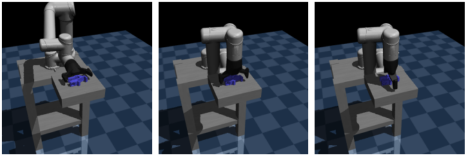
From Beliefs to Decisions





(a) (b) (c)

Fig. 2: Example stable poses.



(a) (b) (c)

Fig. 3: Example grasp configurations from which the actor can attempt to grasp the object - each configuration is associated with a score, i.e., probability of success.

From Beliefs to Decisions

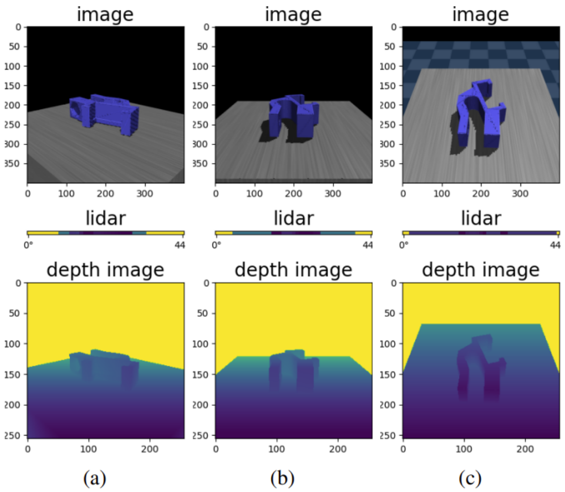
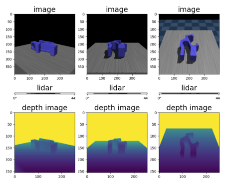


Fig. 4: Example sensor configurations and corresponding observations for a given stable pose. Each column represents the RGB image [top] lidar reading [middle] and depth image [bottom] for a sensor configuration-object pose pair.

Definition (Sensor Function)

Given object pose $p \in \mathcal{P}_o$ and sensor configuration $q \in \mathcal{Q}$, sensor function $\mathcal{O} : \mathcal{P}_o \times \mathcal{Q} \mapsto \Omega$ is a random function, such that if $o = \mathcal{O}(p, q)$ then $P\left(o|p, q\right)$ provides the conditional probability of obtaining the observation o when the sensor configuration is q and the object pose is p.



(a) (b) (c)

From Beliefs to Decisions

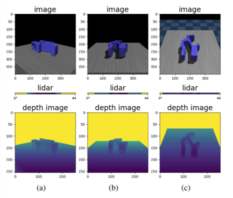
Making Decisions with Partial Information

CLAIR lab Example

Solution Approaches

Definition (Sensor Function)

Given object pose $p \in \mathcal{P}_o$ and sensor configuration $q \in \mathcal{Q}$, sensor function $\mathcal{O} : \mathcal{P}_o \times \mathcal{Q} \mapsto \Omega$ is a random function, such that if $o = \mathcal{O}(p, q)$ then $P\left(o|p, q\right)$ provides the conditional probability of obtaining the observation o when the sensor configuration is q and the object pose is p.



Typically, the actual distribution is not known, and we use a *predicted sensor function* $\tilde{\alpha}$ and a *predicted observation probability* $\hat{P}$, which may be incorrect or inaccurate.

From Beliefs to Decisions

We use a similarity score, $\omega : \Omega \times \Omega \mapsto [0, 1]$ to compare the predicted and received observations:

$$\hat{P}(o|p, q) = \frac{\omega(\tilde{\alpha}(p, q), o)}{\int_{p' \in \mathcal{P}_o} \omega(\tilde{\alpha}(p', q), o) dp'} \tag{1}$$

The literature is rich of various definitions for ω, which may vary between applications and sensor types.

From Beliefs to Decisions

For **belief update** we use a Bayesian filter such that for any observation $o \in \Omega$ taken from sensor configuration $q \in \mathcal{Q}$, the updated pose belief $\beta^{o,q}(p)$ for pose $p \in \mathcal{P}_o$ is given as

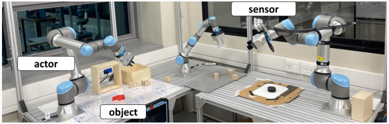
$$\beta^{o,q}(p) = \frac{\hat{P}(o|p,q)\,\beta(p)}{\int_{p' \in \mathcal{P}_o} \hat{P}(o|p',q)\,\beta(p')\,dp'} \tag{2}$$

where $\beta(p)$ is the estimated probability that p is the object pose prior to considering the new observation o.

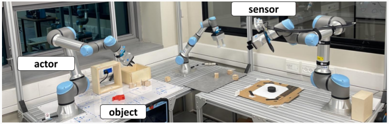
Value of Assistance (VOA) for Grasping

Given the actor's belief $\beta_a \in \mathcal{B}$, the belief of the helping agent $\beta_h \in \mathcal{B}$, the predicated observation probability $\hat{P}$, sensor configuration $q \in \mathcal{Q}$, and the actor's belief update function τ_a,

$$U_{\alpha}^{VOA}(\beta_h, \beta_{ac}) \stackrel{\text{def}}{=\!=} \mathbb{E}_{p\sim\beta_h}\left[\mathbb{E}_{o\sim\hat{P}(o|p,q)}\left[\gamma(q_g^*(\beta_a^{o,q}), p)\right] - \gamma(q_g^*(\beta_{ac}), p))\right]. \tag{3}$$



From Beliefs to Decisions



(a) P1 (b) P2 (c) P3 (d) P4 (e) P5 (f) P6

Is this a good observation?

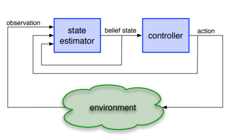
What about sequential decision-making?

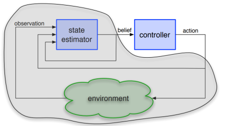
Solution Approaches

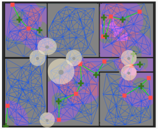
Planning in Belief Space: Solution Approaches

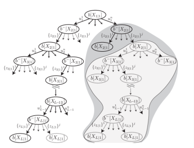
Combinations of different approaches:

- Planning in an MDP with beliefs as states
- Sampling / discretization
- Approximations / relaxations









See work by Vadim Indelman from the Technion, e.g.,

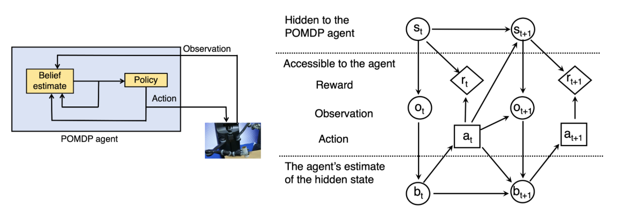
https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8793548

POMDPs and Robotics

Many ideas. We will focus on two.

- SARSOP: Efficient Point-Based POMDP Planning by Approximating Optimally Reachable Belief Spaces. Kurniawati et al. 2008 *https://bigbird.comp.nus.edu.sg/m2ap/wordpress/wp-content/uploads/2016/01/rss08.pdf*
- Efficient point-based POMDP planning by approximating optimally reachable belief spaces Kurniawati (2021): *https://arxiv.org/pdf/2107.07599.pdf*

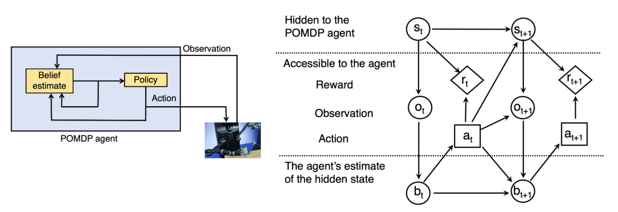
POMDPs and Robotics



POMDPs and Robotics

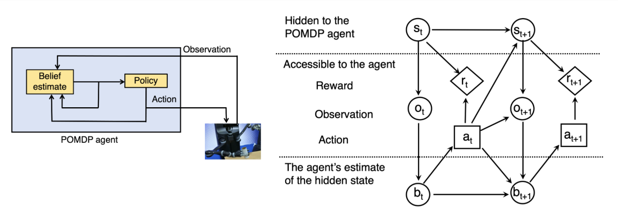
When a POMDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma, \Omega, \mathcal{O}, \beta_0 \rangle$ is used to represent a robot's task

- the transition function is typically represented as a noisy dynamics function $s' = f(s, a, \eta)$, where $s, s' \in \mathcal{S}$ and $\eta \sim N$ is a noise vector sampled from noise distribution N, while f denotes the system's dynamics.
- Similarly, $\mathcal{O}$ denotes the sensor/ observation function, representing errors and noise in measurement and perception.



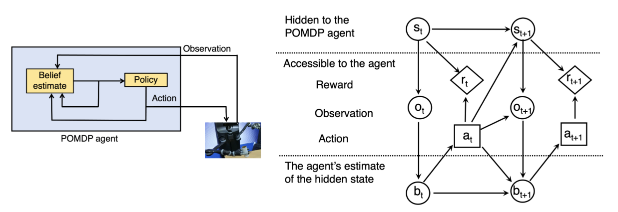
POMDPs and Robotics

- POMDP is powerful in its quantification of the non-deterministic effects of actions and partial observability due to errors in sensor measurements and in perception
- The computed policy will balance information gathering and goal attainment.



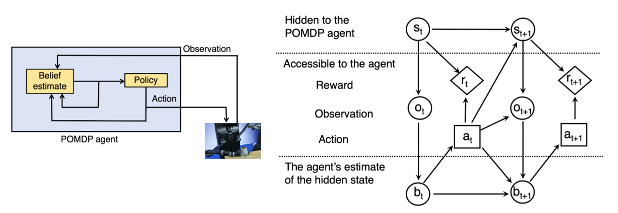
POMDPs and Robotics

- Precisely because of its expressive power, POMDP is notorious for its high computational complexity and deemed impractical for robotics.
- Until recently, most benchmark problems for POMDPs had less than 30 states and the best algorithms that could solve them took hours.

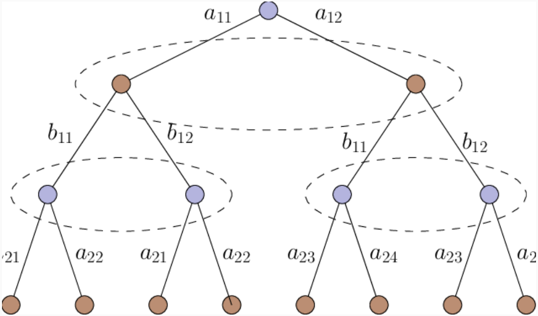


POMDPs and Robotics

- In the past 2 decades, POMDPs solving capabilities have advanced tremendously, thanks to **sampling-based approximate solvers**.
- Although optimality is compromised, robustness and computational efficiency are improved: practical for many realistic robotics problems.



Key idea: sample a set of representative beliefs and compute optimal policy only for them, thus substantially reducing complexity.



POMDPs and Robotics

Algorithm 1 A typical program skeleton for sampling-based POMDP solvers

```
Initialize policy π and a set of sampled beliefs B
{Generally, B is initialised to contain only a single belief (e.g., the initial belief b_0)}
repeat
    Sample a (set of) beliefs {Some methods sample histories (a history is a sequence of action–observation tuples) rather than beliefs. In POMDPs, beliefs provide sufficient statistics of the entire history [25], and therefore the two provide equivalent information}
    Estimate the values of the sampled beliefs
    {Generally, via a combination of heuristics and update / backup operation}
    Update π {In most methods, this step is a byproduct of the previous step}
until Stopping criteria is satisfied
```

- Which set would be sufficiently representative?

Algorithm 1 A typical program skeleton for sampling-based POMDP solvers

1: Initialize policy π and a set of sampled beliefs B
{Generally, B is initialised to contain only a single belief (e.g., the initial belief b_0)}
2: **repeat**
3: Sample a (set of) beliefs {Some methods sample histories (a history is a sequence of action–observation tuples) rather than beliefs. In POMDPs, beliefs provide sufficient statistics of the entire history [25], and therefore the two provide equivalent information}
4: Estimate the values of the sampled beliefs
{Generally, via a combination of heuristics and update / backup operation}
5: Update π {In most methods, this step is a byproduct of the previous step}
6: **until** Stopping criteria is satisfied

- Which set would be sufficiently representative?
  - A variety of sampling strategies have been proposed to select the sample set and to estimate the values of the sampled beliefs.
  - Most sampling-based approximate POMDP solvers are **anytime**
  - Some methods compute upper and lower bound estimates of the value functions

Ideas?

SARSOP

SARSOP: Efficient Point-Based POMDP Planning by Approximating Optimally Reachable Belief Spaces. Kurniawati et al. 2008
https://bigbird.comp.nus.edu.sg/m2ap/wordpress/wp-content/uploads/2016/01/rss08.pdf

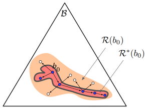


Fig. 1. Belief space $\mathcal{B}$, reachable space $\mathcal{R}(b_0)$, and optimally reachable space $\mathcal{R}^*(b_0)$. Note that $\mathcal{R}^*(b_0) \subseteq \mathcal{R}(b_0) \subseteq \mathcal{B}$.

SARSOP

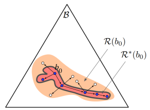


Fig. 1. Belief space $\mathcal{B}$, reachable space $\mathcal{R}(b_0)$, and optimally reachable space $\mathcal{R}^*(b_0)$. Note that $\mathcal{R}^*(b_0) \subseteq \mathcal{R}(b_0) \subseteq \mathcal{B}$.

- Some early POMDP algorithms sample the entire belief space B, using a uniform sampling distribution, such as a grid.
- More recent point-based algorithms sample only $\mathcal{R}(\beta_0)$, the subset of belief points reachable from a given initial point $\beta_0 \in \mathcal{B}$ under arbitrary sequences of actions.

SARSOP

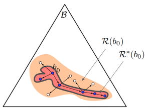


Fig. 1. Belief space $\mathcal{B}$, reachable space $\mathcal{R}(b_0)$, and optimally reachable space $\mathcal{R}^*(b_0)$. Note that $\mathcal{R}^*(b_0) \subseteq \mathcal{R}(b_0) \subseteq \mathcal{B}$.

- SASOP pushes this direction further, by sampling near $\mathcal{R}^*(\beta_0)$, a subset of belief points reachable from β_0 under **optimal sequences of actions**
- $\mathcal{R}^*.(\beta_0)$ is usually much smaller than $\mathcal{R}(\beta_0)$.
- Optimality not achievable, so approximations of $\mathcal{R}^*(\beta_0)$ are used.
    - Use successive approximations of $\mathcal{R}^*(\beta_0)$ and converge to it iteratively.
    - The algorithm relies on heuristic exploration to sample $\mathcal{R}(\beta_0)$ and improves sampling over time through a simple online learning technique.
    - Bounding techniques are used to avoid sampling in regions that are unlikely to be optimal
    - This leads to substantial gain in computational efficiency

We explored approaches to decision-making under uncertainty, when the model is given.

What if we don't have full access to the model of the environment ?
(more on this later in the course)