

BELIEFS: STATE UNCERTAINTY

AA228/CS238 DECISION MAKING UNDER UNCERTAINTY¹

ROBERT MOSS

STANFORD UNIVERSITY

mossr@cs.stanford.edu

OCTOBER 28, 2020

¹Mykel J. Kochenderfer, Tim A. Wheeler, and Kyle H. Wray. *Algorithms for Decision Making*. MIT Press, 2020.

POMDPs AND BELIEFS

- A POMDP² is an MDP with *state uncertainty*

$$\text{MDP: } \langle \mathcal{S}, \mathcal{A}, T, R, \gamma \rangle$$

$$\text{POMDP: } \langle \mathcal{S}, \mathcal{A}, \mathcal{O}, T, R, \mathcal{O}, \gamma \rangle$$

- The agent receives an *observation* of the current state rather than the true state (potentially imperfect observations)
- Using past observations, the agent builds a *belief* of their underlying state
 - Which can be represented by a probability distribution over true states
- Remember, a POMDP is a *problem formulation* and not an *algorithm*
 - A POMDP formulation enables the use of solution methods, i.e. algorithms.

²Partially observable Markov decision process. “Partially observable” is key in understanding beliefs.

OBSERVATION SPACE

- The agent receives an observation o , which belongs to some *observation space* \mathcal{O}
- The probability of observing o given action a and next state s' is: $O(o \mid a, s')$
 - If \mathcal{O} is continuous, then $O(o \mid a, s')$ is a probability density

DYNAMIC DECISION NETWORK FOR POMDPs

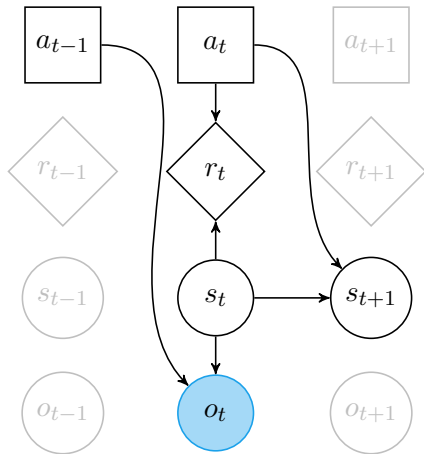


Figure: A dynamic decision network for the POMDP problem formulation.

BELIEF REPRESENTATION

Beliefs can be represented in different ways:

- **Parametric:** The belief distribution is represented by a set of parameters for a fixed distribution family
 - E.g., Categorical distribution³ or multivariate normal (Gaussian) distribution
- **Non-parametric:** The belief distribution is represented by particles (or points sampled from the state space)

Depending on the representation, different algorithms can be used to update beliefs.

³A probability mass is assigned to each discrete category.

ALGORITHMS FOR UPDATING BELIEFS

Various algorithms can update the current belief:

- If the state space is *discrete* (or certain linear Gaussian assumptions are met), then we can perform *exact belief updates*:⁴
 - Recursive Bayesian estimation
 - Kalman filter
- Otherwise, we can use approximations based on linearization or sampling:
 - Extended Kalman filter
 - Unscented Kalman filter
 - Particle filter
 - Particle filter with rejection
 - Injection particle filter
 - Adaptive injection particle filter

⁴Meaning we arrive at an analytical solution without approximations.

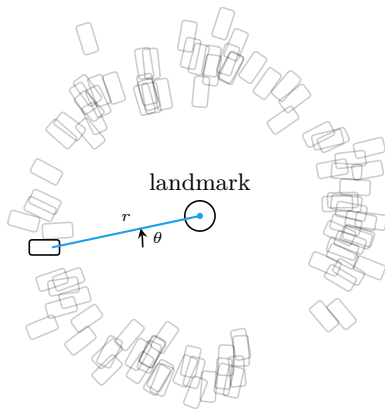
BELIEF INITIALIZATION

Before any actions or observations, we start with an initial belief distribution

- We can encode prior knowledge in the initial distribution
- Generally want to use diffuse (i.e. spread out) initial distributions to avoid over confidence in the absence of information
 - In non-parametric representations, a diffuse initial prior may cause difficulties
 - Thus, we may wait until an informative observation is made to initialize beliefs

EXAMPLE: LANDMARK BELIEF INITIALIZATION

Figure: Localization of an autonomous car using a landmark (Example 19.1).



Making a range r and bearing θ observation, we initialize our belief around the landmark.

BELIEF INFERENCE

- To infer the unknown belief distribution, we use *recursive Bayesian estimation*
 - Updates belief estimate recursively over time
 - Markov assumption: Only requires the current state, action, and observation
- Let $b(s)$ represent the probability⁵ assigned to state s
 - A particular belief b belongs to a *belief space* \mathcal{B} (containing all possible beliefs)
- For finite state and observation spaces, we can use a *discrete state filter* to perform exact inference

⁵or probability density for continuous state spaces

BELIEF VECTOR

- In the finite state case, we can represent beliefs using a categorical distribution⁶
 - Represented as a *belief vector* \mathbf{b} of length $|\mathcal{S}|$, therefore $\mathcal{B} \subset \mathbb{R}^{|\mathcal{S}|}$
 - Sometimes \mathcal{B} is referred to as a *probability simplex* or *belief simplex*⁷
- The belief vector \mathbf{b} must be strictly non-negative and sum to one:

$$b(s) \geq 0 \text{ for all } s \in \mathcal{S} \quad \sum_s b(s) = 1$$

- In vector notation:

$$\mathbf{b} \geq \mathbf{0} \quad \mathbf{1}^\top \mathbf{b} = 1$$

- In Julia syntax:

$$\text{all}(\mathbf{b} .\geq 0) \ \&\& \ \text{sum}(\mathbf{b}) \approx 1$$

⁶A probability mass is assigned to each discrete state.

⁷Simplex being the generalization of a triangle to arbitrary dimensions.

DISCRETE STATE FILTER: UPDATING BELIEFS

A *filter* is a process that remove noise from data.⁸

Due to the independence assumptions, if an agent with belief b takes an action a and receives an observation o , then the new belief b' becomes:⁹

$$\begin{aligned} b'(s') &= P(s' \mid b, a, o) \\ &\propto P(o \mid b, a, s')P(s' \mid b, a) && \text{(Bayes' rule)} \\ &\propto O(o \mid a, s')P(s' \mid b, a) && \text{(observation definition)} \\ &\propto O(o \mid a, s') \sum_s P(s' \mid b, a, s)P(s \mid b, a) && \text{(law of total probability)} \\ &\propto O(o \mid a, s') \sum_s T(s' \mid s, a)b(s) && \text{(state transition model)} \end{aligned}$$

⁸Often used in signal processing, effectively “filtering” out the noise.

⁹For finite/discrete state and observation spaces.

UPDATING BELIEFS: DERIVATION EXPLAINED

$$\begin{aligned} b'(s') &= P(s' \mid b, a, o) && \text{(probability of being in state } s') \\ &\propto P(o \mid b, a, s')P(s' \mid b, a) && \text{(Bayes' rule, dropping normalization)} \\ &\propto O(o \mid a, s')P(s' \mid b, a) && \text{(observation model def., } o \text{ independent of } b) \\ &\propto O(o \mid a, s') \sum_s P(s' \mid b, a, s)P(s \mid b, a) && \text{(law of total probability)} \\ &\propto O(o \mid a, s') \sum_s T(s' \mid s, a)b(s) && \text{(state transition model, belief def.)} \end{aligned}$$

Exact belief updating:¹⁰ $b'(s') \propto O(o \mid a, s') \sum_s T(s' \mid s, a)b(s)$

¹⁰Then normalize so beliefs sum to one.

EXAMPLE: CRYING BABY PROBLEM

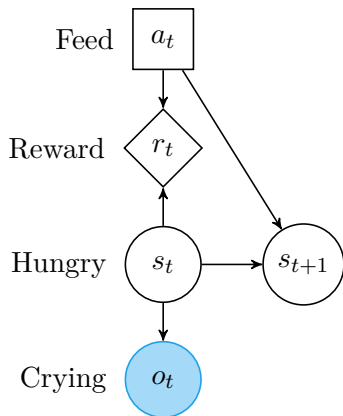


Figure: The crying baby POMDP.

- A simple POMDP with 2 states, 3 actions, and 2 observations:

$$\mathcal{S} = \{\text{hungry}, \text{sated}\}$$

$$\mathcal{A} = \{\text{feed}, \text{sing}, \text{ignore}\}$$

$$\mathcal{O} = \{\text{crying}, \text{quiet}\}$$

- See Pluto notebook:
 - crying_baby_problem.html

KALMAN FILTER

To update beliefs with *continuous* state spaces, we integrate instead of sum:

$$b'(s') \propto O(o \mid a, s') \int T(s' \mid s, a) b(s) ds$$

A *Kalman filter* assumes that T and O are linear-Gaussian and b is Gaussian:

$$T(\mathbf{s}' \mid \mathbf{s}, \mathbf{a}) = \mathcal{N}(\mathbf{s}' \mid \mathbf{T}_s \mathbf{s} + \mathbf{T}_a \mathbf{a}, \Sigma_s)$$

$$O(\mathbf{o} \mid \mathbf{s}') = \mathcal{N}(\mathbf{o} \mid \mathbf{O}_s \mathbf{s}', \Sigma_o)$$

$$b(\mathbf{s}) = \mathcal{N}(\mathbf{s} \mid \boldsymbol{\mu}_b, \Sigma_b)$$

See Pluto notebook: [StateEstimation.jl/kalman_filter.html](https://pluto.jpl.nasa.gov/StateEstimation.jl/kalman_filter.html)

PARTICLE FILTER

- *Particle filters* represent the belief state as a collection of states.
- Each state in the approximated belief is called a *particle*.
- Useful in problems with large discrete states spaces or continuous problems not well approximated by linear-Gaussian dynamics.

Algorithm 1 Particle filter algorithm.

```
function PARTICLEFILTER(b,  $T$ ,  $O$ ,  $a$ ,  $o$ )  
     $\mathbf{s}' \sim T(\mathbf{b}, a)$  ▷ next states  
     $\mathbf{w} \leftarrow O(o \mid \mathbf{s}', a)$  ▷ weights  
    particles  $\sim \text{SetCategorical}\left(\mathbf{s}', \frac{\mathbf{w}}{\sum_i w_i}\right)$  ▷ sample with normalized weights  
return particles
```

See Pluto notebook: [StateEstimation.jl/particle_filter.html](https://pluto.jl.org/StateEstimation.jl/particle_filter.html)

PARTICLE FILTER VARIANTS

Particle filter with rejection:

- Used in problems with discrete observations.
- Any sampled observation that does not equal the true observation is rejected.
- Problem of *particle deprivation*: lack of particles near the true state.¹¹

Injection particle filter:

- Inject random particles to protect against particle deprivation.

Adaptive injection particle filter:

- Inject particles adaptively based on a ratio of two exponentially moving averages of the mean particle weights (using *fast* and *slow* moving averages).

See Pluto notebook: [StateEstimation.jl/particle_filter.html](https://pluto.jl.org/StateEstimation.jl/particle_filter.html)

¹¹Due to low particle coverage given the stochastic nature of resampling.

EXACT BELIEF STATE PLANNING

INTRODUCTION: PLANNING

- The objective in a POMDP is to maximize accumulated reward while interacting with the environment.
- Several methods can compute an optimal policy which maps *beliefs* to *actions*:
 - One approach converts the POMDP into an MDP and applies dynamic programming.¹²
 - Another approach represents policies as conditional plans or piecewise linear value functions over the belief space.

¹²Introducing an algorithm that is analogous to MDP value iteration.

REFERENCES

Kochenderfer, Mykel J., Tim A. Wheeler, and Kyle H. Wray. *Algorithms for Decision Making*. MIT Press, 2020.