Partially Observable Markov Decision Processes (POMDPs)

Sachin Patil
Guest Lecture: CS287 Advanced Robotics
Slides adapted from Pieter Abbeel, Alex Lee
Outline

- Introduction to POMDPs
- Locally Optimal Solutions for POMDPs
  - Trajectory Optimization in (Gaussian) Belief Space
  - Accounting for Discontinuities in Sensing Domains
- Separation Principle
Markov Decision Process \( (S, A, H, T, R) \)

Given

- **S**: set of states
- **A**: set of actions
- **H**: horizon over which the agent will act
- **T**: \( S \times A \times S \times \{0, 1, ..., H\} \rightarrow [0, 1] \), \( T_t(s, a, s') = P(s_{t+1} = s' | s_t = s, a_t = a) \)
- **R**: \( S \times A \times S \times \{0, 1, ..., H\} \rightarrow \mathbb{R} \), \( R_t(s, a, s') = \text{reward for } (s_{t+1} = s', s_t = s, a_t = a) \)

Goal:

- Find \( \pi \) \( : S \times \{0, 1\} \times H \rightarrow A \), that maximizes expected sum of rewards, i.e.,

\[
\pi^* = \arg \max_\pi E\left[ \sum_{t=0}^{H} R_t(S_t, A_t, S_{t+1}) | \pi \right]
\]
POMDP – Partially Observable MDP

= MDP

BUT
don’t get to observe the state itself, instead get sensory measurements

Now: what action to take given current probability distribution rather than given current state.
POMDPs: Tiger Example

S0
“tiger-left”
Pr(o=TL | S0, listen)=0.85
Pr(o=TR | S1, listen)=0.15

S1
“tiger-right”
Pr(o=TL | S0, listen)=0.15
Pr(o=TR | S1, listen)=0.85

Actions={$0$: listen,
        $1$: open-left,
        $2$: open-right$}$

Reward Function
- Penalty for wrong opening: -100
- Reward for correct opening: +10
- Cost for listening action: -1

Observations
- to hear the tiger on the left (TL)
- to hear the tiger on the right (TR)
Belief State

- Probability of $S_0$ vs $S_1$ being true underlying state
- Initial belief state: $p(S_0) = p(S_1) = 0.5$
- Upon listening, the belief state should change according to the Bayesian update (filtering)
Policy – Tiger Example

- Policy $\pi$ is a map from $[0,1] \rightarrow \{\text{listen, open-left, open-right}\}$
- What should the policy be?
  - Roughly: listen until sure, then open
- But where are the cutoffs?

Tiger example optimal policy for $t = 1$

- [0.00, 0.10]
- [0.10, 0.90]
- [0.90, 1.00]

Tiger example optimal policy for $t = 2$

- [0.00, 0.02]
- [0.02, 0.39]
- [0.39, 0.61]
- [0.61, 0.98]
- [0.98, 1.00]
Solving POMDPs

- Canonical solution method 1: Continuous state “belief MDP”
  - Run value iteration, but now the state space is the space of probability distributions
  - → value and optimal action for every possible probability distribution
  - → will automatically trade off information gathering actions versus actions that affect the underlying state

- Value iteration updates cannot be carried out because uncountable number of belief states – approximation
Solving POMDPs

- Canonical solution method 2:
  - Search over sequences of actions with limited look-ahead
  - Branching over actions and observations

Finite horizon:

\[ |A| \frac{|O|^H}{|O|-1} \] nodes

2 steps to go

1 step to go
Solving POMDPs

- Approximate solution: becoming tractable for $|S|$ in millions
  - $\alpha$-vector point-based techniques
  - Monte Carlo Tree Search
  - …Beyond scope of course…
Solving POMDPs

- Canonical solution method 3:
  - Plan in the MDP
  - Probabilistic inference (filtering) to track probability distribution
  - Choose optimal action for MDP for currently most likely state
Outline

- Introduction to POMDPs
- Locally Optimal Solutions for POMDPs
  - Trajectory Optimization in (Gaussian) Belief Space
  - Accounting for Discontinuities in Sensing Domains
- Separation Principle
Facilitate reliable operation of cost-effective robots that use:

- Imprecise actuation mechanisms – serial elastic actuators, cables
- Inaccurate encoders and sensors – gyros, accelerometers
Continuous state/action/observation spaces
Model Uncertainty As Gaussians

Uncertainty parameterized by mean and covariance
Dark-Light Domain

Problem Setup

State space plan

[Example from Platt, Tedrake, Kaelbling, Lozano-Perez, 2010]
Dark-Light Domain

- **Problem Setup**
  - start
  - goal

- **Belief space plan**

Tradeoff information gathering vs. actions
Problem Setup

- Stochastic motion and observation Model

\[ x_{t+1} = f[x_t, u_t, m_t], \quad m_t \sim \mathcal{N}[0, I], \]
\[ z_t = h[x_t, n_t], \quad n_t \sim \mathcal{N}[0, I], \]

- Non-linear

- User-defined objective / cost function

- Plan trajectory that minimizes expected cost
Locally Optimal Solutions

- Belief is Gaussian
  - \( b_t = (\hat{x}_t, \Sigma_t) \),

- Belief dynamics – Bayesian filter
  - [X] Kalman Filter
State Space – Trajectory Optimization

$$\min_{\theta_{1:T}} \sum_{t} \| \theta_{t+1} - \theta_t \|^2 + \text{other costs}$$

subject to

- no collisions
- joint limits
- other constraints
(Gaussian) Belief Space Planning

\[
\min_{\mu, \Sigma, u} \sum_{t=0}^{H} c(\mu_t, \Sigma_t, u_t)
\]

s.t. \((\mu_{t+1}, \Sigma_{t+1}) = xKF(\mu_t, \Sigma_t, u_t, w_t, v_t)\)
\[
\mu_H = \text{goal}
\]
\[
u \in \mathcal{U}
\]
(Gaussian) Belief Space Planning

\[ \min_{\mu, \Sigma, u} \sum_{t=0}^{H} c(\mu_t, \Sigma_t, u_t) \]

s.t. \((\mu_{t+1}, \Sigma_{t+1}) = xKF(\mu_t, \Sigma_t, u_t, 0, 0)\)

\[ \mu_H = \text{goal} \]

\[ u \in \mathcal{U} \]

= maximum likelihood assumption for observations
Can now be solved by Sequential Convex Programming

[Platt et al., 2010; also Roy et al.; van den Berg et al. 2011, 2012]
Dark-Light Domain

Problem Setup

Belief space plan

Tradeoff information gathering vs. actions
Collision Avoidance

- Prior work approximates robot geometry as points or spheres
  
  - Articulated robots cannot be approximated as points/spheres
    - Gaussian noise in joint space
    - Need probabilistic collision avoidance w.r.t robot links

Van den Berg et al.
Sigma Hulls

- **Definition:** Convex hull of a robot link transformed (in joint space) according to sigma points.

- Consider sigma points lying on the $\lambda$-standard deviation contour of uncertainty covariance (UKF).

$$\mathcal{X} = [\hat{x} \ldots \hat{x}] + \lambda[0 \ \sqrt{\Sigma} \ -\sqrt{\Sigma}]$$

Gaussian belief \rightarrow \sigma-points \rightarrow \Sigma Hulls (per link)
Collision Avoidance Constraint

Consider signed distance between obstacle and sigma hulls
Belief space planning using trajectory optimization

- Gaussian belief state in joint space: \( b_t = [\mu_t \Sigma_{\text{cov}}] \)
- Optimization problem:

Variables:

\[
\hat{\mathcal{B}} = [\hat{b}_0 \ldots \hat{b}_T]^T, \quad \hat{\mathcal{U}} = [\hat{u}_0 \ldots \hat{u}_{T-1}]^T
\]

\[
\begin{align*}
\min_{\hat{\mathcal{B}}, \hat{\mathcal{U}}} & \quad C(\hat{\mathcal{B}}, \hat{\mathcal{U}}) \\
\text{s. t.} & \quad \forall t \in T, \quad \hat{b}_{t+1} = g(\hat{b}_t, \hat{u}_t), \quad \Phi(\hat{\mathcal{B}}, \hat{\mathcal{U}}, \lambda) \geq 0, \\
& \quad \psi(\hat{x}_T) = \psi_{\text{target}}, \\
& \quad \hat{u}_t \in F_\mathcal{U},
\end{align*}
\]

Belief dynamics (UKF)  
Probabilistic collision avoidance  
Reach desired end-effector pose  
Control inputs are feasible
Collision avoidance constraint

- Robot trajectory should stay at least $d_{safe}$ distance from other objects

$$sd(A, O) \geq d_{safe} \quad \forall \ O \in O$$
Collision avoidance constraint

- Robot trajectory should stay at least $d_{\text{safe}}$ distance from other objects
  \[ \text{sd}(A, O) \geq d_{\text{safe}} \quad \forall \ O \in \mathcal{O} \]

- Linearize signed distance at current belief
  \[ \text{sd}_{AO}(\hat{b}_t) \approx \hat{n}(\bar{b}_t) \cdot (p_O - p_A(\hat{b}_t)) \]
  \[ \text{sd}_{AO}(\hat{b}_t) \approx \text{sd}_{AO}(\bar{b}_t) + S_t(\hat{b}_t - \bar{b}_t), \]
  \[ S_t = \frac{\partial \text{sd}_{AO}}{\partial \hat{b}}(\bar{b}_t) \approx -\hat{n}(\bar{b}_t)^T \frac{\partial p_A}{\partial \hat{b}}(\bar{b}_t). \]
Collision avoidance constraint

- Robot trajectory should stay at least $d_{\text{safe}}$ distance from other objects

$$\text{sd}(A, O) \geq d_{\text{safe}} \quad \forall \ O \in O$$

- Linearize signed distance at current belief

$$\text{sd}_{AO}(\hat{b}_t) \approx \hat{n}(\vec{b}_t) \cdot (p_O - p_A(\hat{b}_t))$$

$$\text{sd}_{AO}(\hat{b}_t) \approx \text{sd}_{AO}(\vec{b}_t) + S_t(\hat{b}_t - \vec{b}_t),$$

$$S_t = \frac{\partial \text{sd}_{AO}}{\partial \hat{b}}(\vec{b}_t) \approx -\hat{n}(\vec{b}_t)^T \frac{\partial p_A}{\partial \hat{b}}(\vec{b}_t).$$

- Consider the closest point $p_A(\hat{b}_t)$ lies on a face spanned by vertices $p_{A^i}, p_{A^j}, p_{A^k}$

$$\frac{\partial p_A}{\partial \hat{b}}(\vec{b}_t) = \sum_{l \in \{i,j,k\}} \alpha_l \frac{\partial p_{A^l}}{\partial \hat{b}}(\vec{b}_t)$$
Continuous Collision Avoidance Constraint

- Discrete collision avoidance can lead to trajectories that collide with obstacles in between time steps.

- Use convex hull of sigma hulls between consecutive time steps: 
  \[ \text{sd}(\text{convhull}(A_t, A_{t+1}), O) \geq d_{\text{safe}} \quad \forall O \in \mathcal{O} \]

- Advantages:
  - Solutions are collision-free in between time-steps.
  - Discretized trajectory can have less time-steps.

(a) Obstacle does not collide with discrete-time sigma hulls
(b) Obstacle overlaps with continuous-time sigma hulls
Model Predictive Control (MPC)

- During execution, update the belief state based on the actual observation
- Re-plan after every belief state update
- Effective feedback control, provided one can re-plan sufficiently fast
Example: 4-DOF planar robot
Example: 4-DOF planar robot

1-standard deviation belief space trajectory

$x_{\text{sensing}}$  Initial mean state $\hat{x}_0$

Final mean state $\hat{x}_T$

Narrow clearance from obstacles between consecutive time steps (sigma hull for last time step)
Example: 4-DOF planar robot

4-standard deviation belief space trajectory

\( x_{\text{sensing}} \)

Initial mean state \( \hat{x}_0 \)

Final mean state \( \hat{x}_T \)

Wider clearance from obstacles between consecutive time steps (sigma hull at last time step)
Experiments: 4-DOF planar robot

Probability of collision

- Red: Belief space plan (Open-loop execution)
- Green: Belief space plan (Feedback policy)
- Purple: Belief space re-planning

Collisions (%) vs. λ-parameter
Experiments: 4-DOF planar robot

Mean distance from target

- Red: Belief space plan (Open-loop execution)
- Green: Belief space plan (Feedback policy)
- Purple: Belief space re-planning

Distance from target vs. \( \lambda \)-parameter
Efficient trajectory optimization in Gaussian belief spaces to reduce task uncertainty

Prior work approximates robot geometry as a point or a single sphere

Pose collision constraints using signed distance between sigma hulls of robot links and obstacles

Sigma hulls never explicitly computed – fast convex collision checking and analytical gradients

Iterative re-planning in belief space (MPC)
Outline

- Introduction to POMDPs
- Locally Optimal Solutions for POMDPs
  - Trajectory Optimization in (Gaussian) Belief Space
  - Accounting for Discontinuities in Sensing Domains
- Separation Principle
Discontinuities in Sensing Domains

Zero gradient, hence local optimum

Patil et al., under review
Discontinuities in Sensing Domains

Increasing difficulty

Noise level determined by signed distance to sensing region
* homotopy iteration
Signed Distance to Sensing Discontinuity

Field of view (FOV) discontinuity

Occlusion discontinuity

(a) $sd(\hat{b}_t, \Pi^s) > 0$
Outside field of view

(b) $sd(\hat{b}_t, \Pi^s) < 0$
Inside field of view

(c) $sd(\hat{b}_t, \Pi^s) > 0$
Occluded view

(d) $sd(\hat{b}_t, \Pi^s) < 0$
Unoccluded view
$\delta_t^s$ vs. Signed distance

(a) $sd(\hat{b}_t, \Pi^s) > 0$
Outside field of view

(b) $sd(\hat{b}_t, \Pi^s) < 0$
Inside field of view

(c) $sd(\hat{b}_t, \Pi^s) > 0$
Occluded view

(d) $sd(\hat{b}_t, \Pi^s) < 0$
Unoccluded view

$\delta_t^s = \chi(sd(\hat{b}_t, \Pi^s))$
Modified Belief Dynamics

\[ x_{t+1} = f(x_t, u_t, q_t), \quad q_t \sim \mathcal{N}(0, I), \]
\[ z_t = h(x_t, r_t), \quad r_t \sim \mathcal{N}(0, I), \]

\[ \hat{b}_{t+1} = g(\hat{b}_t, \hat{u}_t) = \left[ \text{vec}\left( \frac{\hat{x}_{t+1}}{\sqrt{\Sigma_{t+1}^{-1} - K_t H_t \Sigma_{t+1}^{-1}}} \right) \right] \]

\[ \hat{x}_{t+1} = f(\hat{x}_t, \hat{u}_t, 0), \quad \Sigma_{t+1}^{-1} = A_t \sqrt{\Sigma_t} (A_t \sqrt{\Sigma_t})^T + Q_t Q_t^T, \]
\[ A_t = \frac{\partial f}{\partial x}(\hat{x}_t, \hat{u}_t, 0), \quad Q_t = \frac{\partial f}{\partial q}(\hat{x}_t, \hat{u}_t, 0), \]
\[ H_t = \frac{\partial h}{\partial x}(\hat{x}_{t+1}, 0), \quad R_t = \frac{\partial h}{\partial r}(\hat{x}_{t+1}, 0), \]
\[ K_t = \Sigma_{t+1}^{-1} H_t^T \Delta_{t+1} \left( \Delta_{t+1} H_t \Sigma_{t+1}^{-1} H_t^T \Delta_{t+1} + R_t R_t^T \right)^{-1} \Delta_{t+1}. \]

\[ \delta_i^s : \text{Binary variable } \{0, 1\} \]
\[ 0 \rightarrow \text{No measurement} \]
\[ 1 \rightarrow \text{Measurement} \]
Incorporating $\delta^s_t$ in SQP

- Binary non-convex program – difficult to solve
- Solve successively smooth approximations

\[ \delta^s_t(\alpha) = \tilde{\chi}(\text{sd}(\hat{b}_t, \Pi^s), \alpha) \]
\[ = 1 - \frac{1}{1 + \exp(-\alpha \cdot \text{sd}(\hat{b}_t, \Pi^s))} \]
Algorithm Overview

- While $\delta$ not within desired tolerance
  - Solve optimization problem with current value of $\alpha$
  - Increase $\alpha$
  - Re-integrate belief trajectory
  - Update $\delta$
Discontinuities in Sensing Domains

Increasing difficulty

Noise level determined by signed distance to sensing region

* homotopy iteration
“No measurement” Belief Update

Sensing region boundary

Truncate Gaussian Belief if no measurement obtained

Truncated belief state
Effect of Truncation

Without “No measurement” Belief Update

With “No measurement” Belief Update
Experiments
Car and Landmarks (Active Exploration)
Arm Occluding (Static) Camera

Initial belief  State space plan execution  (way-point)  (end) Belief space plan execution
Arm Occluding (Moving) Camera

Initial belief  State space plan execution  (way-point)  Belief space plan execution
Outline

- Introduction to POMDPs
- Locally Optimal Solutions for POMDPs
  - Trajectory Optimization in (Gaussian) Belief Space
  - Accounting for Discontinuities in Sensing Domains
- Separation Principle
**Separation Principle**

- **Assume:**
  \[
  x_{t+1} = Ax_t + Bu_t + w_t \quad w_t \sim \mathcal{N}(0, Q_t)
  \]
  \[
  z_t = Cx_t + v_t \quad v_t \sim \mathcal{N}(0, R_t)
  \]

- **Goal:**
  \[
  \text{minimize } E \left[ \sum_{t=0}^{H} u_t^\top U_t u_t + x_t^\top X_t x_t \right]
  \]

- **Then, optimal control policy consists of:**

  1. **Offline/Ahead of time:** Run LQR to find optimal control policy for fully observed case, which gives sequence of feedback matrices \( K_1, K_2, \ldots \)

  2. **Online:** Run Kalman filter to estimate state, and apply control
     \[
     u_t = K_t \mu_{t|0:t}
     \]
Extensions

- Current research directions
  - Fast! belief space planning
  - Multi-modal belief spaces
  - Physical experiments with the Raven surgical robot
Recap

- POMDP = MDP but sensory measurements instead of exact state knowledge
- Locally optimal solutions in Gaussian belief spaces
  - Augmented state vector (mean, covariance)
  - Trajectory optimization
- Sigma Hulls for probabilistic collision avoidance
- Homotopy methods for dealing with discontinuities in sensing domains