Bayesian Reinforcement Learning in Continuous POMDPs

Stéphane Ross\textsuperscript{1}, Brahim Chaib-draa\textsuperscript{2} and Joelle Pineau\textsuperscript{1}

\textsuperscript{1}School of Computer Science, McGill University, Canada
\textsuperscript{2}Department of Computer Science, Laval University, Canada

May 23\textsuperscript{rd}, 2008
Motivation

Robots have to make decisions under:
- Imperfect Actuators
- Noisy Sensors
- Poor/Approximate Model

How to maximize long-term rewards?

[Rottmann]
Continuous POMDP

- **States**: $S \subseteq \mathbb{R}^m$
- **Actions**: $A \subseteq \mathbb{R}^n$
- **Observations**: $Z \subseteq \mathbb{R}^p$
- **Rewards**: $R(s, a) \in \mathbb{R}$

Gaussian model for Transition/Observation function:

- $s_t = g_T(s_{t-1}, a_{t-1}, X_t)$, $X_t \sim N(\mu_X, \Sigma_X)$
- $z_t = g_O(s_t, a_{t-1}, Y_t)$, $Y_t \sim N(\mu_Y, \Sigma_Y)$
Simple Robot Navigation Task:

\[
\begin{bmatrix}
  x' \\
  y'
\end{bmatrix} =
\begin{bmatrix}
  x \\
  y
\end{bmatrix} + v
\begin{bmatrix}
  \cos \theta & -\sin \theta \\
  \sin \theta & \cos \theta
\end{bmatrix}
\begin{bmatrix}
  X_1 \\
  X_2
\end{bmatrix}
\]

\[
\begin{bmatrix}
  z_x \\
  z_y
\end{bmatrix} =
\begin{bmatrix}
  x \\
  y
\end{bmatrix} +
\begin{bmatrix}
  Y_1 \\
  Y_2
\end{bmatrix}
\]

+1 reward when \( ||s - s_{GOAL}||_2 < d \)
In practice: $\mu_X, \Sigma_X, \mu_Y, \Sigma_Y$ unknown.

Need to trade-off between:
- Learning the model
- Identifying the state
- Gathering rewards
Bayesian Reinforcement Learning

Current Prior/Posterior

New Posterior

Observation

Action

Environment

Planner
Planning problem representable as a new POMDP:

- **States**: \((s, \theta)\)
- **Actions**: \(a \in A\)
- **Observations**: \(z \in Z\)
- **Rewards**: \(R(s, \theta, a) = R(s, a)\)

Joint Transition-Observation Probabilities:
\[
\Pr(s', \theta', z|s, \theta, a) = \Pr(s', z|s, a, \theta)I_\theta(\theta')
\]
Belief State = Posterior

**Belief Update :**

\[ b^{az}(s', \theta) \propto \int_s b(s, \theta) \Pr(s', z|s, a, \theta)ds \]

**Optimal policy by solving :**

\[ V^*(b) = \max_{a \in A} \left[ \int_s R(s, a) \Pr(s|b)ds + \gamma \int_z \Pr(z|b, a)V^*(b^{az})dz \right] \]
Belief Update

Bayesian Learning of \((\mu, \Sigma)\):

- Normal-Wishart prior \(\Rightarrow\) Normal-Wishart posterior
- Parametrized by \((n, \hat{\mu}, \hat{\Sigma})\)

Start with prior: \((n_0, \hat{\mu}_0, \hat{\Sigma}_0)\)

Posterior Update (after observing \(X = x\)):

\[
\begin{align*}
n' &= n + 1 \\
\hat{\mu}' &= \frac{n\hat{\mu} + x}{n+1} \\
\hat{\Sigma}' &= \frac{n-1}{n} \hat{\Sigma} + \frac{1}{n+1} (x - \hat{\mu})(x - \hat{\mu})^T
\end{align*}
\]
Belief Update

But $X$ not directly observable:

$$\Pr(\mu, \Sigma | z) \propto \int \Pr(\mu, \Sigma | x) \Pr(z | x) \Pr(x) dx$$

Approximate infinite mixture by finite mixture

Particle filter:

- Use particles of the form $(s, \phi, \psi)$
- $\phi, \psi$: Normal-Wishart posterior parameters for $X, Y$
Particle Filter

\[ b_t \]

\[ \phi_t \]

\[ \mu_X, \Sigma_X \]

\[ \sim NW(\phi_t) \]

\[ \phi_{t+1} \]

\[ \sim NW(\phi_t) \]

\[ s_t \]

\[ \phi_{t+1} \]

\[ X_t \]

\[ \sim N(\mu_X, \Sigma_X) \]

\[ Y_t \]

\[ \sim N(\mu_Y, \Sigma_Y) \]

\[ \phi_{t+1} \]

\[ a_t \]

\[ g_T(s_t, a_t, X_t) \]

\[ s_{t+1} \]

\[ g_O(s_{t+1}, a_t, Y_t) \]

\[ z_{t+1} \]
Monte Carlo Online Planning (Receding Horizon Control):
Simple Robot Navigation Task

Average evolution of the return over time:
Simple Robot Navigation Task

Average accuracy of the model over time:

Model Accuracy is measured as follows:

$$WL1(b) = \sum_{(s, \phi, \psi)} b(s, \phi, \psi) \left[ ||\mu_\phi - \mu_X||_1 + ||\Sigma_\phi - \Sigma_X||_1 + ||\mu_\psi - \mu_Y||_1 + ||\Sigma_\psi - \Sigma_Y||_1 \right]$$
Conclusion

- Presented a framework for optimal control under model and state uncertainty.
- Monte Carlo approximations for efficient tracking and planning.
- Framework can easily be extended to unknown rewards and mixture of Gaussians model.
Future Work

- What if $g_T, g_O$ unknown?
- What if $(\mu, \Sigma)$ change over time?
- More efficient planning algorithms.
- Apply to a real robot.
Questions
?