Learning Dynamic Manipulation Skills under Unknown Dynamics with Guided Policy Search

Sergey Levine  
UC Berkeley

Pieter Abbeel  
UC Berkeley
Team TROOPER: Lockheed Martin, University of Pennsylvania, Rensselaer Polytechnic Institute

Philipp Krahenbuhl, Stanford University

2D Insertion
unobserved slot positions
[neural network]
general-purpose neural network controller

\[ \theta = \arg \min_\theta J(\theta) \]

\[ J(\theta) = E_{\pi_\theta} \left[ \sum_{t=1}^{T} c(x_t, u_t) \right] \]

\[ \pi_\theta(u_t|x_t) - \text{control policy} \]

\[ \mathcal{N}(\mu_\theta(x), \Sigma_\theta) \]
policy search (RL)  complex dynamics  complex policy  HARD

supervised learning  complex dynamics  complex policy  EASY

trajectory optimization  complex dynamics  complex policy  EASY

trajectory optimization  supervised learning
Trajectory Optimization

guided policy search
Trajectory Optimization

\[ \min_{q(\tau)} E_q[c(\tau)] - \mathcal{H}(q) \]

approximate solution using iterative LQR
(similar to extended Kalman filter)

- locally linear dynamics
- locally quadratic cost
- Gaussian distribution

\[ q(u_t|x_t) = \mathcal{N}(k_t + K_t x_t, \Sigma_t) \]
1. Run time-varying policy $q(u_t|x_t)$ on robot $N$ times
2. Collect dataset $D = \{\tau_i\}$ where $\tau_i = \{x_{1i}, u_{1i}, \ldots, x_{Ti}, u_{Ti}\}$
3. For each $t \in \{0, \ldots, T - 1\}$, fit linear Gaussian $p(x_{t+1}|x_t, u_t)$
4. Solve control problem to get new $q(u_t|x_t)$
Trajectory Optimization

\[
\min_{q(\tau)} E_q[c(\tau)] \quad \text{s.t.} \quad D_{KL}(q(\tau)\|\bar{q}(\tau)) \leq \epsilon
\]

\[
\frac{1}{\eta} \mathcal{L}(q, \eta) = E_q \left[ \frac{1}{\eta} c(\tau) - \log \bar{q}(\tau) \right] - \mathcal{H}(q) - \epsilon
\]

\[
\min_{q(\tau)} E_q[c(\tau)] - \mathcal{H}(q) \quad \tilde{c}(\tau) = \frac{1}{\eta} c(\tau) - \log \bar{q}(\tau)
\]
run \( q(\mathbf{u}_t | \mathbf{x}_t) \) on robot
collect \( \mathcal{D} = \{\tau_i\} \)

fit dynamics \( p(\mathbf{x}_{t+1} | \mathbf{x}, \mathbf{u}) \)

DP solve for \( q(\mathbf{u}_t | \mathbf{x}_t) \)

update \( \eta \)
1. Run time-varying policy $q(u_t|x_t)$ on robot $N$ times
2. Collect dataset $\mathcal{D} = \{\tau_i\}$ where $\tau_i = \{x_{1i}, u_{1i}, \ldots, x_{Ti}, u_{Ti}\}$
3. For each $t \in \{0, \ldots, T-1\}$, fit linear Gaussian $p(x_{t+1}|x_t, u_t)$
4. Solve control problem to get new $q(u_t|x_t)$
run $q(u_t|x_t)$ on robot
collect $\mathcal{D} = \{\tau_i\}$

$\{x_j, u_j, x'_j\}$  $\{\tau_i\}$

fit GMM

fit dynamics $p(x_{t+1}|x, u)$

DP solve for $q(u_t|x_t)$

update $\eta$
Trajectory Optimization

2D Insertion
- optimized trajectory
- linear Gaussian control

Swimming
- optimized trajectory
- [linear Gaussian]
Trajectory Optimization
Guided Policy Search

see Levine & Koltun, ICML 2014
run $q(u_t | x_t)$ on robot
collect $D = \{\tau_i\}$

$\{x_j, u_j, x'_j\}$

fit GMM

fit dynamics $p(x_{t+1} | x, u)$

DP solve for $q(u_t | x_t)$

next iteration

prior

data

update $\eta$

train $\pi_\theta(u_t | x_t)$
2D Insertion
unobserved slot positions
[neural network]

Swimming
learned policy
[neural network]
Concluding Comments

• simple linear dynamics model
• fast, simple, standard LQR solver
• can handle contacts despite linear model
• fit very complex policies with guided policy search