Information-theoretic policy search methods for learning versatile, reusable skills

Gerhard Neumann

University of Lincoln / University of Darmstadt
Some geography…

Lincoln?
Learning Complex Motor Skills

Case Study: Table-Tennis

Humans

Robots

More Autonomy in Learning:
• Efficient skill exploration
• Reactive skills
• Simultaneous learning of multiple skills
• Switching between skills

State-of-the art:
• Learn single skill (forehand)
• Predefined hitting plane
• Heavily depends on demonstrations
• Engineered skill activations
• Limited policy representations

Mülling + Peters
Reinforcement Learning (RL)
Reinforcement Learning (RL)

Learning: Adapting the policy $\pi(u|x)$ of the agent

Objective: Find policy that maximizes expected long term reward

$$J_\pi = \mathbb{E} \left[ \sum_{t=0}^{T} r_t \mid u_t \sim \pi(\cdot|x_t) \right]$$
Movement Primitives

Parametrized trajectories:

$$\tau^* = q_{1:T}^* = f(w)$$

- Desired mean movement
- Followed by trajectory tracking controllers

Examples:

- Dynamic Movement Primitives (DMPs) [Ijspeert 2002]
- Probabilistic Movement Primitives (ProMPs) [Paraschos 2013]

Reinforcement Learning by Direct Policy Search:

$$\pi^* = \arg \max_\pi \int \pi(w) R(w) dw$$

$$R(w) = \mathbb{E} \left[ \sum_{t=1}^{T} r_t \mid w \right]$$

- Find distribution $$\pi(w)$$ with maximum expected return $$R(w)$$
**Information-Theoretic Direct Policy Search**

**Search distribution / parameter policy:**  \( \pi(w) = \mathcal{N}(w|\mu, \Sigma) \)

- **Mean:** Estimate of the maximum
- **Covariance:** Direction to explore

**How to update?**

- **Too moderate**
- **Too greedy**
- **About right**

**Exploration-Exploitation tradeoff:** immediate vs. long-term performance
How can we find a good update?

Information-theoretic policy update: incorporate information from new samples

1. Maximize return
   \[ \arg \max_{\pi} \int \pi(w) R(w) dw \]

2. Bound information loss [Peters 2011]
   \[ \text{s.t.:} \quad \text{KL}(\pi || \pi_{\text{old}}) \leq \epsilon \]
   Controls step-size for mean and covariance

Kullback Leibler Divergence:

\[ \text{KL}(q || p) = \int q(w) \log \frac{q(w)}{p(w)} dw \]

- Information-theoretic similarity measure
- Non-negative
- Non-symmetric

Illustration: Distribution Update

Large initial exploration

Small initial exploration

26.05.2016 | Gerhard Neumann | EWRL 2016 | Barcelona | 9
**Information-Theoretic Policy Update**

**Information-theoretic policy update:** incorporate information from new samples

1. Maximize return
   \[
   \arg \max_{\pi} \int \pi(\omega) R(\omega) d\omega
   \]
   reduces variance too quickly

2. Bound information gain [Peters 2011]
   \[
   \text{s.t. } \text{KL}(\pi || \pi_{\text{old}}) \leq \epsilon
   \]

   \[
   H(\pi_{\text{old}}) - H(\pi) \leq \gamma
   \]
   loss in entropy

**Entropy:**

\[
H(p) = - \int p(\omega) \log p(\omega) d\omega
\]

- Measure for uncertainty

---

A. Abdolmaleki, ..., **G. Neumann**, *Model-Based Relative Entropy Stochastic Search*, NIPS 2015
Illustration: Distribution Update

No entropy loss bound

With bounded entropy loss
Solution for Search Distribution

Solution for unconstrained distribution: \( \pi(w) \propto \pi_{\text{old}}(w) \frac{\eta}{\eta + \omega} \exp \left( \frac{R(w)}{\eta + \omega} \right) \)

- \( \eta \) … Lagrangian multiplier for: \( \text{KL}(\pi \| \pi_{\text{old}}) \leq \epsilon \)
  \( \epsilon \to 0 \quad \Rightarrow \quad \eta \to \infty \quad \Rightarrow \quad \pi \to \pi_{\text{old}} \)
  \( \epsilon \to \infty \quad \Rightarrow \quad \eta \to 0 \quad \Rightarrow \quad \pi \to \text{greedy} \)

- \( \omega \) … Lagrangian multiplier for: \( H(\pi_{\text{old}}) - H(\pi) \leq \gamma \)
  \( \gamma \to 0 \quad \Rightarrow \quad \omega \gg 0 \quad \Rightarrow \quad \pi \to \text{more uniform} \)

Gaussianity needs to be „enforced“!

- Fit new policy on samples (REPS, [Daniel2012, Kupcsik2014, Neumann2014])
- Fit return function on samples (MORE, [Abdolmaleki2015])

A. Abdolmaleki, ..., G. Neumann, Model-Based Relative Entropy Stochastic Search, NIPS 2015

26.05.2016 | Gerhard Neumann | EWRL 2016 | Barcelona | 12
Fit Return Function

Use compatible function approximation:
- Gaussian distribution: $\mathcal{N}(w|\mu, \Sigma) \propto \exp \left( -\frac{1}{2} w^T \Sigma^{-1} w + w^T \Sigma^{-1} \mu - \frac{1}{2} \mu^T \Sigma^{-1} \mu \right)$
  - quadratic
  - linear
  - const
- Match functional form: $\tilde{R}(w) = w^T A w + a^T w + a_0 \approx R(w)$
- Quadratic in $w$, but linear in parameters: $\theta_R = \{A, a, a_0\}$
- $\theta_R$ obtained by linear regression on current set of samples

Model-Based Relative Entropy Stochastic Search (MORE) : [Abdolmaleki 2015]

1. Evaluation: Fit local surrogate $\tilde{R}(w) \approx w^T A w + a^T w + a_0$
2. Update: $\pi(w) \propto \pi_{\text{old}}(w) \frac{\eta}{\eta + \omega} \exp \left( \frac{\tilde{R}(w)}{\eta + \omega} \right) \Rightarrow \pi(w) = \mathcal{N}(w|\mu^*, \Sigma^*)$

A. Abdolmaleki, …, G. Neumann, Model-Based Relative Entropy Stochastic Search, NIPS 2015
Skill Improvement: Table Tennis

Setup:
- Single ball configuration
- 17 movement primitive parameters (DMPs)

More Autonomy in Learning:
- Efficient skill exploration
- Reactive skill representations
- Simultaneous learning of multiple skills
- Switching between skills
**Adaptation of Skills**

**Goal:** Adapt parameters $\mathbf{w}$ to different situations

- Different ball trajectories
- Different target locations

**Introduce context vector $\mathbf{s}$**

- Continuous valued vector
- Characterizes environment and objectives of agent
- Individual context per execution $\mathbf{s} \sim p(\mathbf{s})$

Learn contextual distribution $\pi(\mathbf{w}|\mathbf{s}) = \mathcal{N}(\mathbf{w}|M\phi(\mathbf{s}), \Sigma)$

---

Abdolmaleki, …, **Neumann**, *Model-Based Relative Entropy Stochastic Search*, NIPS 2015
Kupcsik, …, **Neumann**, *Data-Efficient Generalization of Robot Skills with Contextual Policy Search*, AAAI 2013
Adaptation of Skills

Contextual distribution update:

1. Maximize expected return
   \[ \arg \max_{\pi} \mathbb{E}_p(s) \left[ \int \pi(w|s)R(s, w)dw \right] \]
2. Bound expected information loss
   \[ \mathbb{E}_p(s) \left[ \text{KL}(\pi(\cdot|s)||\pi_{\text{old}}(\cdot|s)) \right] \leq \epsilon \]
3. Bound entropy loss
   \[ H(\pi_{\text{old}}) - H(\pi) \leq \gamma \]

Contextual MORE: [Abdolmaleki 2015]

1. Evaluation: Fit local surrogate \( \tilde{R}(s, w) \approx w^T A w + a^T B \phi(s) + a^T w + a_0 \)
2. Update: \( \pi(w|s) \propto \pi_{\text{old}}(w|s)^{\eta+\omega} \exp \left( \frac{\tilde{R}(s, w)}{\eta + \omega} \right) \)
   \[ \Rightarrow \pi(w|s) = \mathcal{N} \left( w|M^* \phi(s), \Sigma^* \right) \]

A. Abdolmaleki, …, G. Neumann, Model-Based Relative Entropy Stochastic Search, NIPS 2015
Adaptation of Skills: Table Tennis

Contextual Policy Search:

- Context: Initial ball velocity (in 3 dimensions)
- Successfully return 100% of the balls
Reactive Skills

**Goal:** React to unforeseen events
- **Adaptation during execution** of the movement
- **Add perceptual variables** to state representation $x$
- E.g.: ball position + velocity

**Approach:**
- Learn and explore in action space
- **No linearizations of dynamics**

**Example:** Perturbation at impact (spin)

Learn time-dependent feedback controllers

$$\pi_t(u|x) = \mathcal{N}(K_t x + k_t, \Sigma_t)$$
Policy Evaluation

Time-Dependent Local Value Function Approximation

- **V-Function:**
  Quality of state $s$ when following policy $\pi$

$$V_t^\pi(x) = \mathbb{E}_\pi \left[ \sum_{h=t}^{T} r_h(x_h, u_h) \middle| x_t = x \right] \approx x^T V_t x + x^T v_t + v_{0,t}$$

- **Q-Function:**
  Quality of state $s$ when taking action $a$ and following policy $\pi$ afterwards

$$Q_t^\pi(x, u) = \mathbb{E}_\pi \left[ \sum_{h=t}^{T} r_h(x_h, u_h) \middle| x_t = x, u_t = u \right] \approx a^T Q_t a + a^T B_t x + a^T q_t + q_{0,t} + f_t(x)$$

- Estimated by linear regression
Policy Improvement

Policy Improvement per Time-Step:

1. Maximize Q-Function
   \[
   \arg \max_{\pi_t} \mathbb{E}_{p_t(x)} \left[ \int \pi_t(u|x) Q_{t}^{\pi_{old}}(x, u) du \right]
   \]

2. Bound expected information loss
   s.t.: \[
   \mathbb{E}_{p_t(x)} \left[ \text{KL}(\pi_t(\cdot|x) || \pi_{t,old}(\cdot|x)) \right] \leq \epsilon
   \]

3. Bound entropy loss
   \[
   H(\pi_{t,old}) - H(\pi_t) \leq \gamma
   \]

Model-free Trajectory Optimization (MOTO): [Akour 2016]

1. Evaluation: Fit local Q-Function
   \[
   \tilde{Q}^{\pi_{old,t}}(x, u) \approx a^T Q_t a + a^T B_t x + a^T q_t + q_{0,t} + f_t(x)
   \]

2. Update:
   \[
   \pi_t(u|x) \propto \pi_{old,t}(u|x) \frac{\eta}{\eta + \omega} \exp \left( \frac{\tilde{Q}_{t}^{\pi_{old}}(x, u)}{\eta + \omega} \right)
   \]
   \[
   \Rightarrow \pi_t(u|x) = \mathcal{N}(u|K_t^* x + k_t^*, \Sigma_t^*)
   \]

Reactive Skills: Table Tennis

Reactive Skills:

- Returns ball 100% of the times
- Not possible with desired trajectories

More Autonomy in Learning:

- Efficient skill exploration
- Reactive skills
- Simultaneous learning of multiple skills
- Switching between skills
Skill Selection

**Goal:** Select skill given a context

- Forehand
- Backhand
- Smash...

**Hierarchical Policy:**

1. **Gating policy:** Select skill
   \[
   \pi(o|s)
   \]

2. **Single Skills:** Decide on controls
   \[
   \pi(w|s, o_1) \\
   \pi(w|s, o_2) \\
   \pi(w|s, o_3)
   \]
Hierarchical RL for Skill Selection

- Independent RL problems
- Estimate reward models using importance sampling
  \[ \arg\min_{\tilde{R}} \int p(x, \theta | o) \left( \tilde{R}(x, \theta) - R(x, \theta) \right)^2 \]
  \[ \approx \arg\min_{\tilde{R}} \sum_i p(x_i, \theta_i | o) \frac{p(x_i, \theta_i)}{p(x_i, \theta)} \left( \tilde{R}(x_i, \theta_i) - R_i \right)^2 \]
- Shared experience
- Similar update strategy can be used for gating layer

Learned Skills

More Autonomy in Learning:
- Efficient skill exploration
- Reactive skills
- Simultaneous learning of multiple skills
- Switching between skills
Skill Activation

Full game:
• No fixed horizon
• When to *activate which stroke*?
• When to *terminate*?

Use Options / Option Discovery!

Ongoing work: concentrate on easier problem

Existing approaches [Simsek, McGovern, Konidaris]:
• Find „Bottlenecks“ in the state space
• Chaining of goal states
  ✓ Transfer Learning
  ✗ Often limited to discrete domains
  ✗ Need to know the MDP

Probabilistic Inference [Daniel2016]:
• Graphical model with latent variables
• Parametric policies at all levels
  ✓ Continuous domains
  ✓ Combine simple controllers
Hierarchical Policy [Sutton2000]:

- **Initiation policy:**
  \[ \pi(O_t | x_t) \theta_O \]

- **Termination policy:**
  \[ \pi(b_t | o_{t-1}, x_t, x; \theta_B^{[j]}) \]

- **Sub-policy:**
  \[ \pi(u_t | x_t, \theta_A^{[j]}) \]

Parameters: \( \Theta = \{ \theta_O, \theta_B^{[1:K]}, \theta_A^{[1:K]} \} \)

---

**C. Daniel, …, G. Neumann (2016):** Probabilistic Inference for Determining Options in Reinforcement Learning, Machine Learning, best student paper ECML (journal track)
Hierarchical Imitation Learning

Imitation Learning:
- Optimize marginal log-likelihood:
  \[ L(\Theta) = \sum_{o_{1:T}, b_{1:T}} \log p(x_{1:T}, u_{1:T}, o_{1:T}, b_{1:T} | \Theta) \]
- Latent variables: option index, termination event,
- Observed variables: states, actions

Expectation-Maximization
- **E-Step:** Compute responsibilities: \[ p_t(\theta_O, b_{t:t+1}, o_{t+1} | x_{1:T}, u_{1:T}, \Theta_{old}) \]
- **M-Step:** Optimize lower bound \[ \arg \max_{\Theta} \mathcal{L}(\Theta, q) \]
  - Initiation: \[ \mathcal{L}_O(\theta_O, q) = \sum_t \mathbb{E}_{q_t} \left[ \log \pi(o_t | x_t; \theta_O) \right] \]
  - Termination: \[ \mathcal{L}_B(\theta_B^{[j]}, q) = \sum_t \mathbb{E}_{q_t} \left[ \log \pi(b_t | x_t, o_{t-1}; \theta_B^{[j]}) \right] \quad o_{t-1} = j \]
  - Sub-Policies: \[ \mathcal{L}_U(\theta_U^{[j]}, q) = \sum_t \mathbb{E}_{q_t} \left[ \log \pi(u_t | x_t, o_t; \theta_U^{[j]}) \right] \quad o_t = j \]

Decoupled updates
Option Discovery for Reinforcement Learning

Probabilistic Reinforcement Learning:

- Compute weighting for each sample
  \[ p_i = \exp \left( \frac{Q(x_i, u_i) - V(x_i)}{\eta} \right) \]

- Use **weighted maximum likelihood** for policy update

**M-Step:** Introduce weighting of data-points

- **Initiation:**
  \[ \mathcal{L}_O(\theta_O, q) = \sum_t \mathbb{E}_{q_t} \left[ \log \pi \left( \frac{Q(x_t, u_t)}{b_t = 1} \right) \log p_t(x_t, o_t; \theta_O) \big| b_t = 1 \right] \]

- **Termination:**
  \[ \mathcal{L}_B(\theta_B^{[j]}, q) = \sum_t \mathbb{E}_{q_t} \left[ \log \pi \left( \frac{Q(x_t, u_t)}{b_t = 1} \right) \log p_t(x_t, o_t; \theta_B^{[j]}) \big| o_t-1 = j \right] \]

- **Sub-Policies:**
  \[ \mathcal{L}_U(\theta_U^{[j]}, q) = \sum_t \mathbb{E}_{q_t} \left[ \log \pi \left( \frac{Q(x_t, u_t)}{b_t = 1} \right) \log p_t(x_t, o_t; \theta_U^{[j]}) \big| o_t = j \right] \]

**Weighted maximum likelihood at each layer!**
Option Learning Results

Grid-World:

![Graph showing performance comparison across different algorithms in a corridor world scenario.](image)

**Pendulum Swing-Up:**

![Graph showing performance across episodes for different methods.](image)

More Autonomy in Learning:

- Efficient skill exploration
- Reactive skills
- Simultaneous learning of multiple skills
- Switching between skills
More than Table Tennis

Hierarchical Grasp Learning:

The learned policy was tested with 10 objects on a real robotic system.

T. Osa
Conclusion: Motor Skill Learning

Efficient Motor Skills Learning:

- Efficient local search with Gaussian distributions
- Information-theoretic policy updates
- Versatile skills using information-theoretic objectives
- Learn hierarchical policies for skill selection by hierarchical RL

Future work:

- Application on the real robot
- Integrate switching between skills in MOTO framework
- Deep Networks and Perception
- More Layers for Hierarchical RL
Swarm Robotics

Kilobots:

- Massively parallel low cost robot
- 3 legs, rotate and move with vibration
- **Limited sensing** and communication
  - Light sensors
  - Sense distance to other Kilobots (< 7cm)
  - Own location and location of other Kilobots unknown

- Future of **molecular robotics**?

Applications:

- Cooperative Assembly
- Forming structures (graphs)
- Localizing targets
Learning for Swarm Robotics

Challenges:

- Varying number of agents
- High dimensional states
- Only local limited observations
- Need to cooperate

Reinforcement Learning on such systems is often infeasible

Applications:

- Cooperative Assembly
- Forming structures (graphs)
- Localizing targets
Centralized Control: Cooperative Assembly

Assemble 2 objects:
- Pushing them together
- Specified orientation

Task Decomposition
- Plan desired path for objects
- Push objects along path with object pushing policy

Object pushing policy:
- Learned with reinforcement learning
- Different desired rotation and translation

G. Gebhardt, …, G. Neumann: Learning to assemble objects with robot swarms, submitted to AAMAS 2017
Centralized Control: Cooperative Assembly

Learning object pushing policy:

- **State:**
  Distribution of Kilobots relative to object
  Invariance to number of kilobots

- **Action:**
  Movement of torch-light

- **Reward:**
  Achieve desired translation and rotation of object

- Learned with Actor-Critic REPS [Wirth, AAAI 2016]
Decentralized Control: Deep Guided Learning

Guided Learning:

- During learning, we know full state \( x \) (e.g. cameras)
- Policies can only use histories \( h \) of observations \( o \)
- Learning infeasible otherwise

Actor(s)-Critic Architecture:

- **Critic**: Learns full-state Q-Function \( Q^\mu_i(x, u_1, \ldots, u_N) \)
- **Actors/Policies**: Choose action due to local history \( u_i = \mu(h_i) \)

Deep representations for actors and critic

- Trained with Deep Deterministic Policy Gradients [Silver 2014]

M. Hüttenrauch, …, G. Neumann: Deep Guided Reinforcement Learning for Robot Swarms, submitted to AAMAS 2017
Decentralized Control: Deep Guided Learning

Observations:
- Histogram of distances to nearby agents

Task 1: Building graph-structures
- Edge: distance between kilobots = d
- Reward: maximize number of edges
Decentralized Control: Deep Guided Learning

Observations:

- Histogram of distances to nearby agents
- Distance to target (if seen)
- Flag if target has been reached

Task 2: Localizing Target

- Maximize number of Agents that have seen target
Conclusion: Multi-Agent Learning

Centralized control:

- Learning of complex task such as object assembly
- Invariance to number of agents: state as distribution of Kilbots

Decentralized control:

- Guided learning with full state information
- Can learn decentralized control policies

Future work:

- Implementation on real robots
- More complex objects, more complex tasks
- Communication, simulate pheromons
- Memory
Many thanks to...

My team:

IAS @ TUDa

Collaborations: