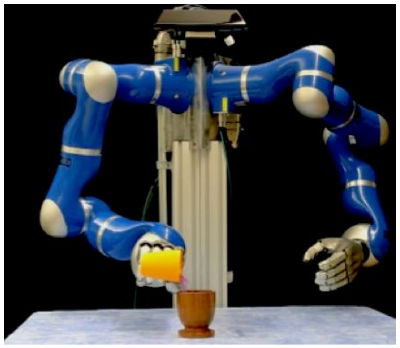
(REINFORCEMENT) LEARNING IN ROBOTICS

Gerhard Neumann

TECHNISCHE UNIVERSITÄT DARMSTADT
UNIVERSITY OF LINCOLN
EBERHARD KARLS UNIVERSITÄT TÜBINGEN
BOSCH

Long term vision

Future: Autonomous robots



Household

Health / Elderly Care

Assist as Co-Workers

Farming

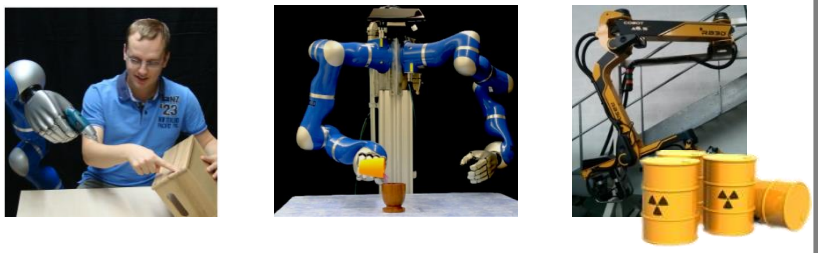
Hazardous Environments

Traffic

Challenges:

- × Complex skills
- × Data-efficient self improvement
- × Cooperation
- × Perception

How can we accomplish these challenges?



Machine Learning

Autonomous Robots

Data-Driven Algorithms

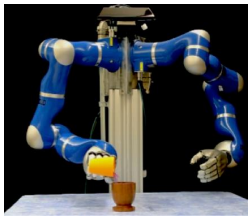
Domain Knowledge

Data-driven movement primitives	× Complex Skills
Reinforcement Learning	× Self-improvement
Representation Learning	× Cooperation
Multi-Agent Learning	× Perception

How can we accomplish these challenges?

Machine Learning

Data-Driven Algorithms


Autonomous Robots

Domain Knowledge

Data-driven movement primitives	× **Complex Skills**
Reinforcement Learning	× Self-improvement
Multi-Agent Learning	× Cooperation
Representation Learning	× Perception

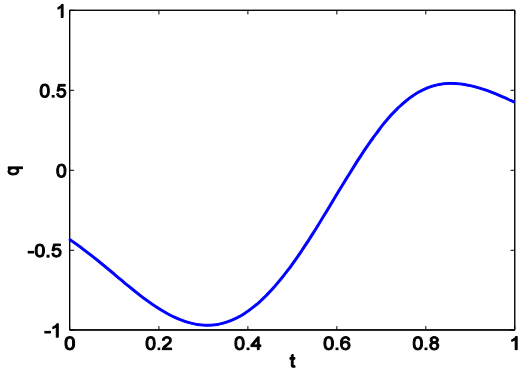
Data-Driven Movement Representations

Goal: data-driven, compact, flexible

Parametrized trajectories:

$$\boldsymbol{\tau}^* = \boldsymbol{q}_{1:T} = f(\boldsymbol{\theta})$$

- Mean movement
- Followed by trajectory tracking controllers

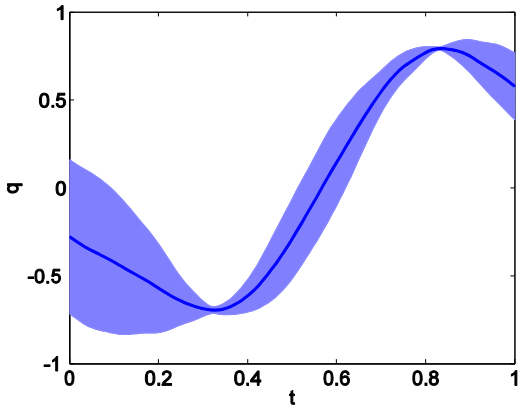


- Example: Dynamic Movement Primitives (DMPs) [Ijspeert 2002]

Parametrized trajectory distribution:

$$\boldsymbol{\tau} \sim p(\boldsymbol{\tau}) \Leftrightarrow \boldsymbol{\theta} \sim p(\boldsymbol{\theta})$$

- Family of movements
- **Gaussian:** Mean and Variance



- **Probabilistic Movement Primitives (ProMPs)** [Paraschos 2013]

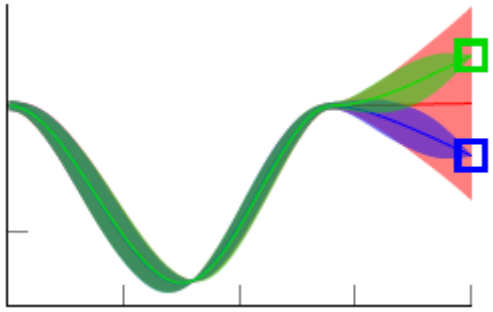
A. Paraschos, …, **G. Neumann,** *Probabilistic Movement Primitives*, NIPS, 2013

Adaptation and Variable Stiffness

Adapt final/intermediate position of the movement

- **Conditioning/Bayes Theorem**

$$p(\boldsymbol{\theta}|\boldsymbol{q}_t = \boldsymbol{q}_t^*) = \frac{p(\boldsymbol{q}_t^*|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\boldsymbol{q}_t^*)}$$

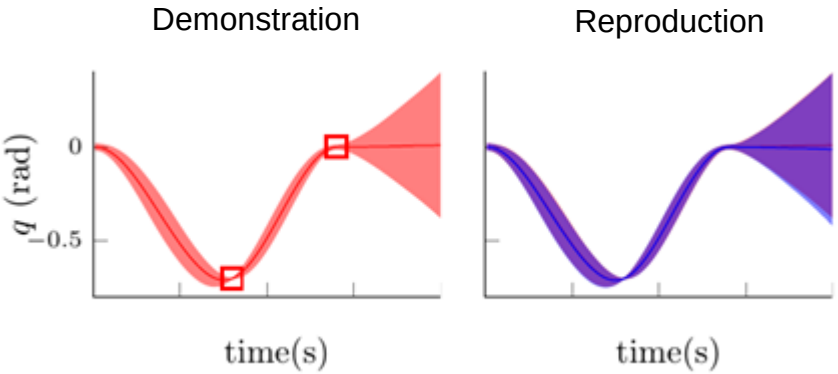


Variable stiffness

- Obtain variable stiffness controller

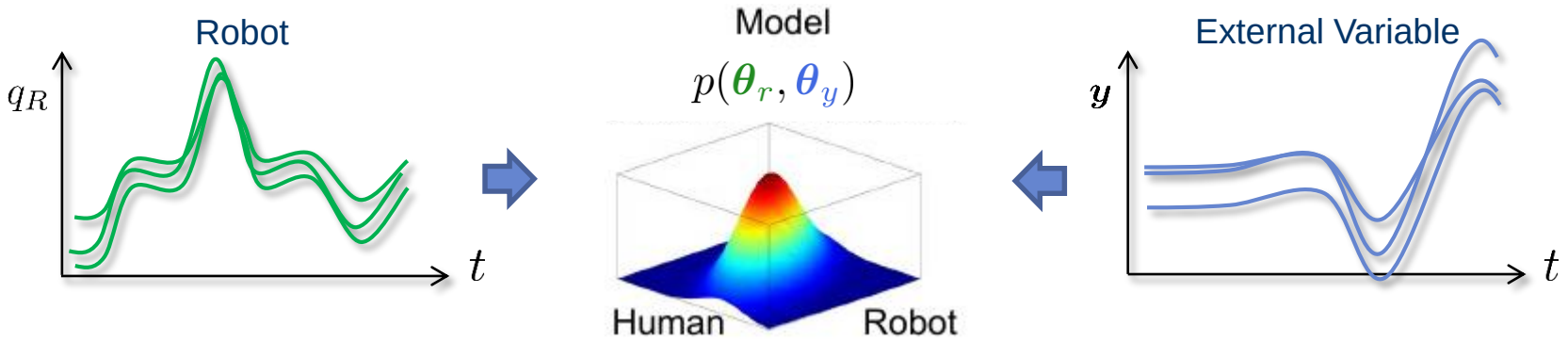
$$\boldsymbol{u}_t = \boldsymbol{K}_t \boldsymbol{x}_t + \boldsymbol{k}_t$$

- Matches moment of trajectory distribution
- High stiffness only if needed
- Requirement for safe interaction

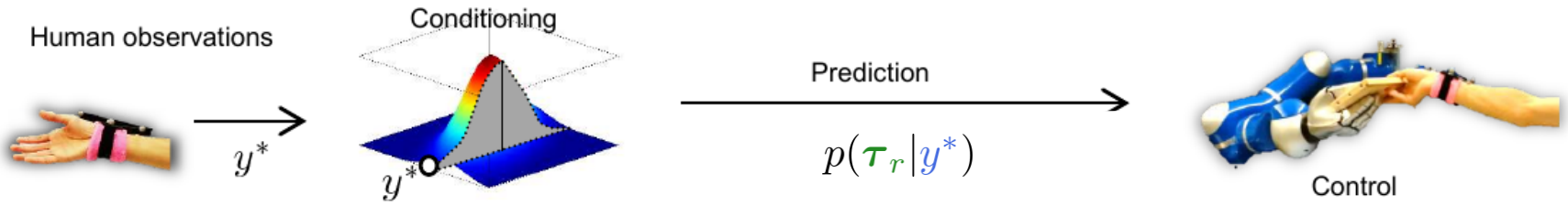


Movement coupling

- E.g. couple with movement of interaction partner y …
- Learn **joint distribution** of trajectory τ_y and robot trajectory τ_r from demonstrations



- **Condition** on observations of y:

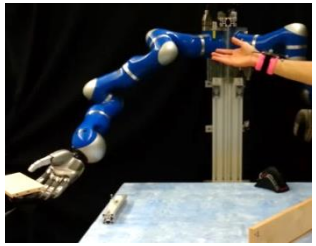


Robot Assistant

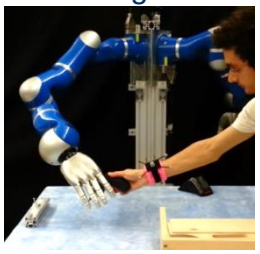
Box assembly

- **Learn interaction patterns** by kinesthetic teach in

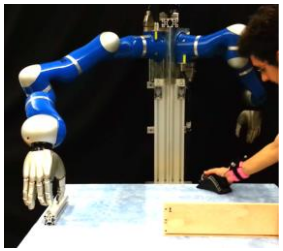
Plate handover



Holding tool



Screw handover



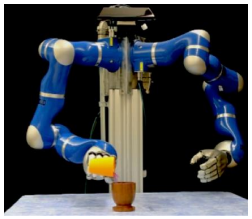
- **Couple robot movement** with human

Flexible and Complex Skills:

- **Variable stiffness** (for safe interaction)
- **Modulation**
- **Coupling**

Maeda, G.; **Neumann, G.**; et al. *Probabilistic Movement Primitives, for Coordination of Multiple Human-Robot Collaborative Tasks,* Autonomous Robots (AURO)

How can we accomplish these challenges?


Machine Learning

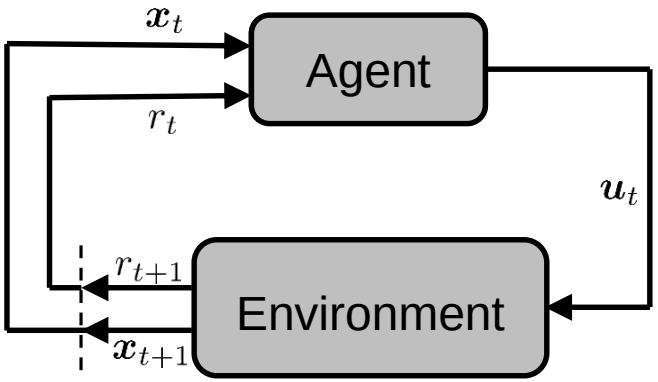
Data-Driven Algorithms

Autonomous Robots

Domain Knowledge

Data-driven movement primitives	× Flexible Skills
Reinforcement Learning	× **Self-improvement**
Multi-Agent Learning	× Cooperation
Representation Learning	× Perception

Reinforcement Learning (RL)



policy

$\pi(\boldsymbol{u}|\boldsymbol{x}),\ \boldsymbol{u} = \pi(\boldsymbol{x})$

Learning: **Adapting the policy $\pi(\boldsymbol{u}|\boldsymbol{x})$ of the agent**

Objective: Find policy that maximizes expected return

$$J_\pi = \mathbb{E}\left[\sum_{t=0}^{T} r_t \middle| \boldsymbol{u}_t \sim \pi(\cdot|\boldsymbol{x}_t)\right]$$

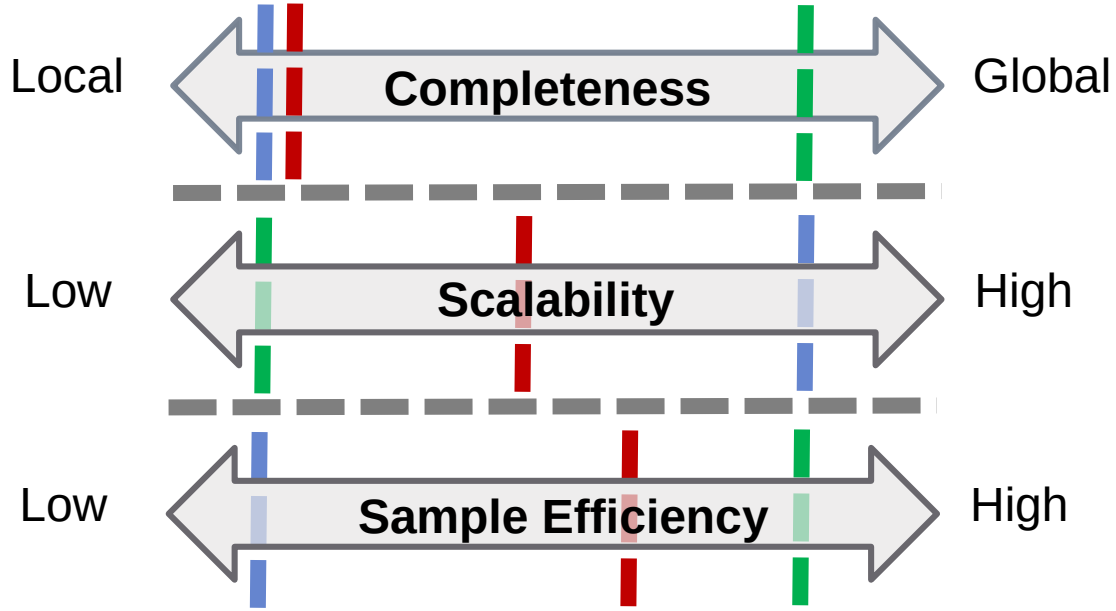
Learning with movement primitives

Objective: Find parameters $\boldsymbol{\theta}$ that maximize the expected return

$$R(\boldsymbol{\theta}) = \mathbb{E}\left[\sum_{t=0}^{T} r_t \middle| \boldsymbol{u}_t = \pi_{\boldsymbol{\theta}}(\boldsymbol{x}_t)\right]$$

Coarse Categorization of RL

- Deep Reinforcement Learning
- Bayesian Optimization
- Direct Policy Search / Stochastic Search



Direct Policy Search

Direct Policy Search by Stochastic Search:

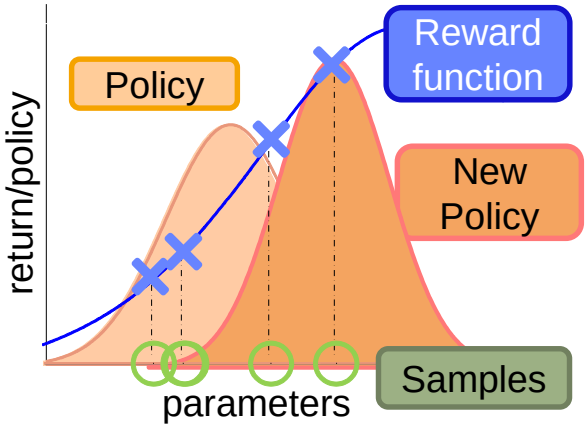
$$\pi^* = \arg\max_{\pi} \int \pi(\boldsymbol{\theta}) R(\boldsymbol{\theta}) d\boldsymbol{\theta}$$

Parameters of Movement Primitive

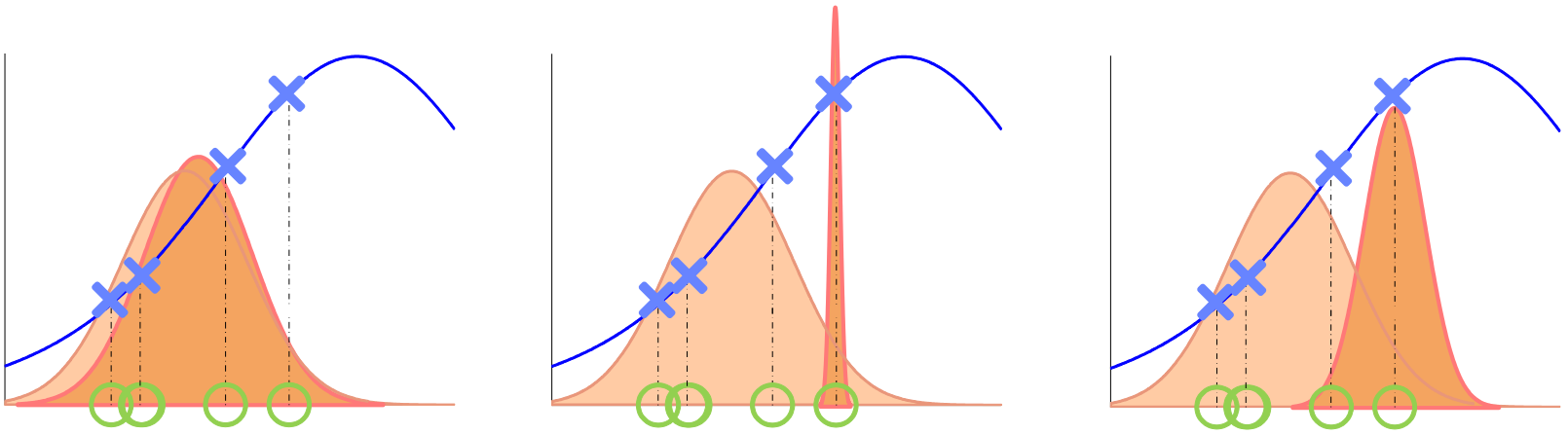
- Find distribution $\pi(\boldsymbol{\theta})$ with maximum expected return $R(\boldsymbol{\theta})$

Three basic steps:

- **Explore:** Generate trajectories $\boldsymbol{\tau}^{[i]}$ following the policy π_k
- **Evaluate:** Assess quality of trajectory or actions
- **Update:** Compute new policy π_{k+1}



How to find a good policy update?



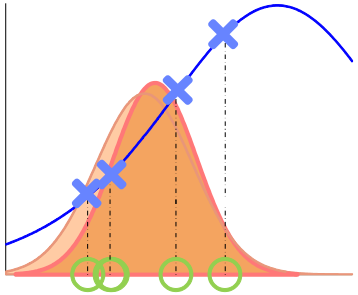
Too moderate **Too greedy** **About right**

Control exploration-exploitation tradeoff: immediate vs. long-term performance

Information-Theoretic Policy Update

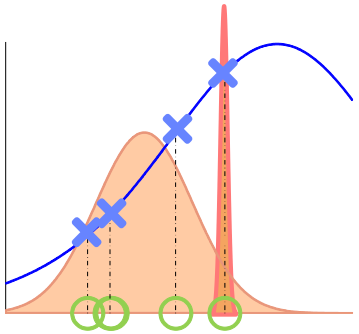
Information-theoretic policy update: incorporate information from new samples

1. Maximize return $\arg\max_{\pi} \int \pi(\boldsymbol{\theta}) R(\boldsymbol{\theta}) d\boldsymbol{\theta}$
2. Bound information gain [Peters 2011] $\text{s.t.} \quad \text{KL}\left(\pi || \pi_{\text{old}}\right) \leq \epsilon$ **Control Greediness**
3. Bound entropy loss [Abdolmaleki 2015] $\underbrace{H(\pi_{\text{old}}) - H(\pi)}_{\text{loss in entropy}} \leq \gamma$ **Control Entropy**



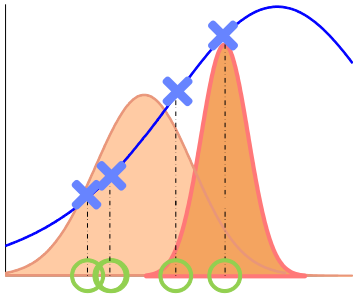
Too moderate

$\epsilon \ll 1$



Too greedy

$\epsilon \gg 1 \quad \gamma \gg 1$



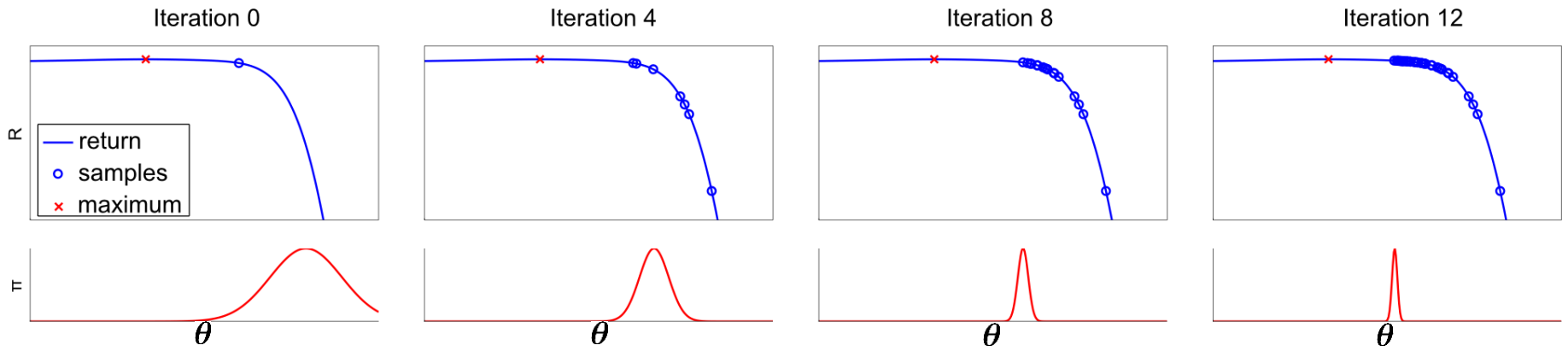
About right

$\epsilon \gg 1 \quad \gamma \ll 1$

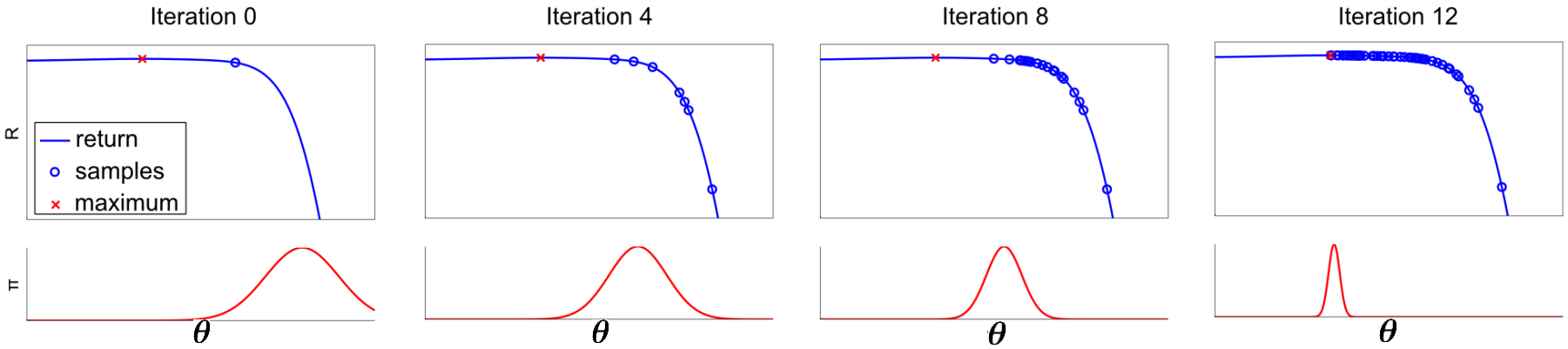
A. Abdolmaleki, …, **G. Neumann**, *Model-Based Relative Entropy Stochastic Search,* NIPS 2015

Illustration: Distribution Update

No entropy loss bound



With bounded entropy loss



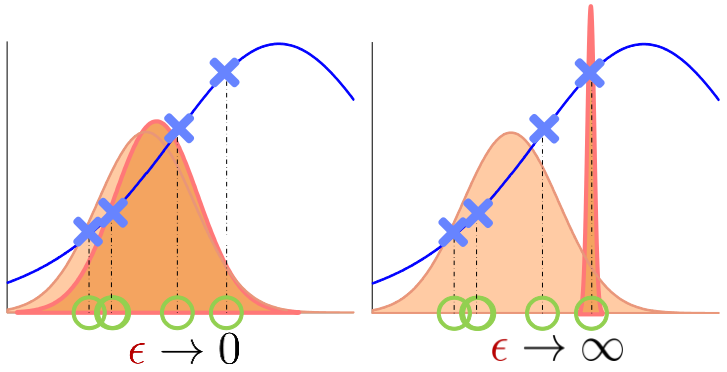
Solution for Search Distribution

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

- η … Lagrangian multiplier for: $\mathrm{KL}\left(\pi||\pi_{\text{old}}\right) \leq \epsilon$

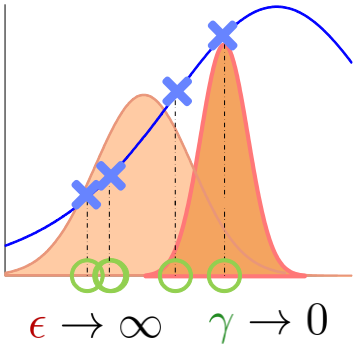
  $\epsilon \to 0$ ➡ $\eta \to \infty$ ➡ $\pi \to \pi_{\text{old}}$

  $\epsilon \to \infty$ ➡ $\eta \to 0$ ➡ $\pi \to$ greedy



- ω … Lagrangian multiplier for: $H\left(\pi_{\text{old}}\right) - H\left(\pi\right) \leq \gamma$

  $\gamma \to 0$ ➡ $\omega \gg 0$ ➡ $\pi \to$ more uniform



Gaussianity needs to be „enforced“ !

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

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

Fit Return Function

Use compatible function approximation:

- Gaussian distribution: $\mathcal{N}[\boldsymbol{\theta}|\boldsymbol{m}, \boldsymbol{\Lambda}] \propto \exp\left(-\frac{1}{2}\boldsymbol{\theta}^T\boldsymbol{\Lambda}\boldsymbol{\theta} + \boldsymbol{\theta}^T\boldsymbol{m} + \text{const}\right)$
- Gaussian in cannonical form (log linear)
- **Parameters:** Precision $\boldsymbol{\Lambda}$ and linear part $\boldsymbol{m}$

quadratic linear const

Match functional form: $\tilde{R}(\boldsymbol{\theta}) = \boldsymbol{\theta}^T\boldsymbol{A}\boldsymbol{\theta} + \boldsymbol{a}^T\boldsymbol{\theta} + a_0 \approx R(\boldsymbol{\theta})$

- Quadratic in $\boldsymbol{\theta}$, but linear in parameters: $\boldsymbol{w} = \{\boldsymbol{A}, \boldsymbol{a}, a_0\}$
- $\boldsymbol{w}$ obtained by **linear regression** on current set of samples
- Fits **local curvature** of the objective function

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

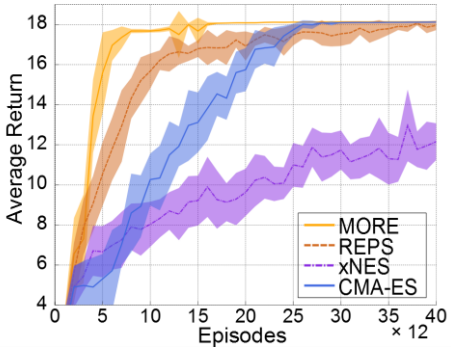
MORE Algorithm

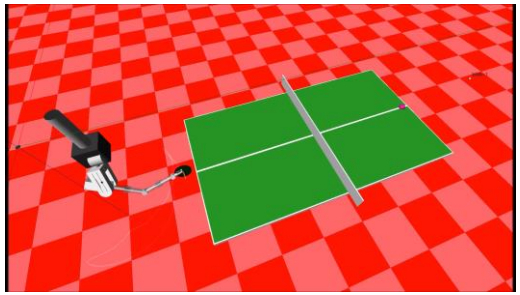
Model-Based Relative Entropy Stochastic Search (MORE) : [Abdolmaleki 2015]

1. **Evaluation:** Fit local surrogate $\tilde{R}(\boldsymbol{\theta}) \approx \boldsymbol{\theta}^T \boldsymbol{A}\boldsymbol{\theta} + \boldsymbol{a}^T\boldsymbol{\theta} + a_0$

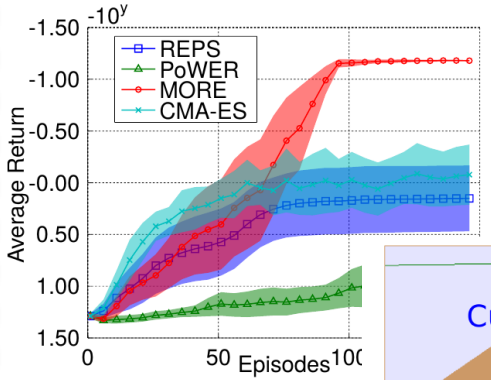
2. **Update:** $\pi(\boldsymbol{\theta}) \propto \pi_{\text{old}}(\boldsymbol{\theta})^{\frac{\eta}{\eta+\omega}} \exp\left(\frac{\tilde{R}(\boldsymbol{\theta})}{\eta+\omega}\right) \Rightarrow \pi(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*)$

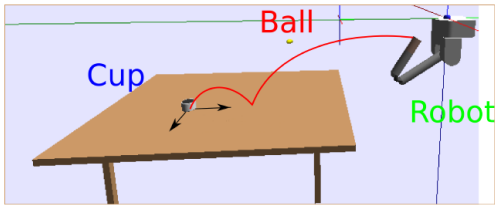
Table-Tennis





Beer-Pong





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

Adaptation of Skills

Adapt parameters to context s

- Continuous valued vector
- Characterizes environment / objectives

Learn contextual distribution:

$$\pi(\boldsymbol{\theta}|\boldsymbol{s}) = \mathcal{N}\left(\boldsymbol{\theta}|M\boldsymbol{\phi}(\boldsymbol{s}), \boldsymbol{\Sigma}\right)$$

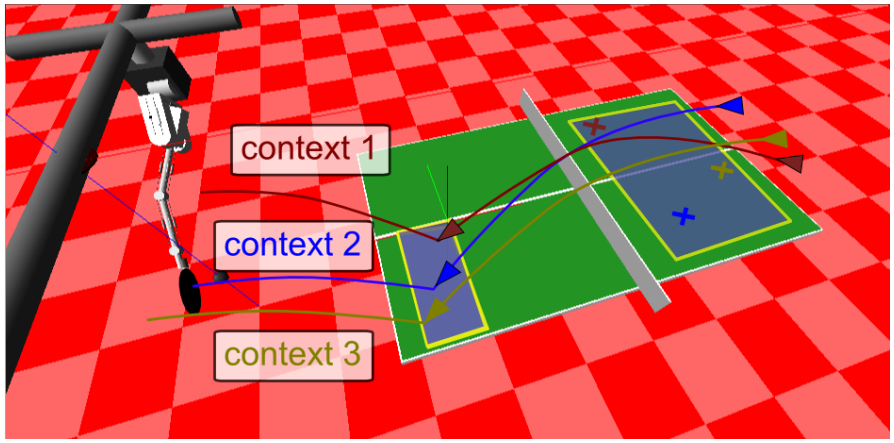
- Parameters are adapted to context

Leads to equivalent formulation:

$$\arg\max_{\pi} \mathbb{E}_{p(\boldsymbol{s})}\left[\int \pi(\boldsymbol{\theta}|\boldsymbol{s}) R(\boldsymbol{s},\boldsymbol{\theta}) d\boldsymbol{\theta}\right]$$

$$\text{s.t.:} \quad \mathbb{E}_{p(\boldsymbol{s})}\left[\mathrm{KL}\left(\pi(\cdot|\boldsymbol{s})||\pi_{\mathrm{old}}(\cdot|\boldsymbol{s})\right)\right] \leq \epsilon$$

$$H(\pi_{\mathrm{old}}) - H(\pi) \leq \gamma$$

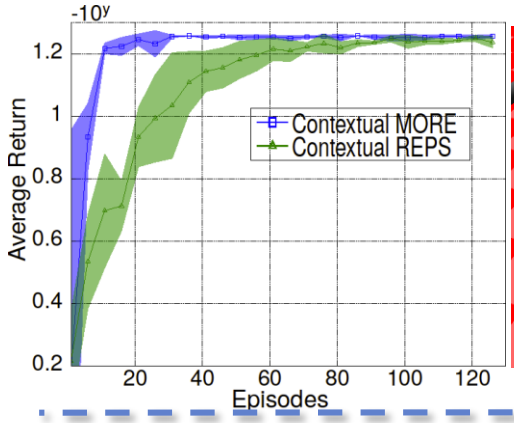


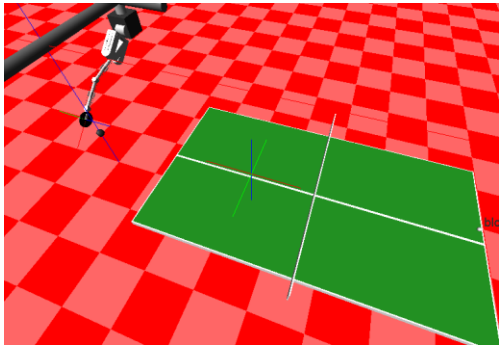
Abdolmaleki, …, **Neumann**, *Model-Based Relative Entropy Stochastic Search,* NIPS 2015

Adaptation of Skills

Table-Tennis:

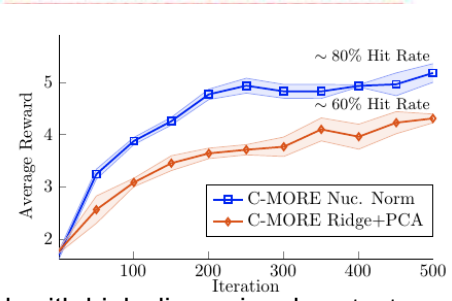
- Initial ball configuration
- Desired return impact point





Also works for high-D context:

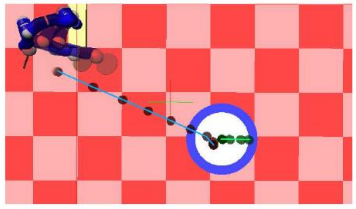
- Raw images of ball positions
- Has to reach shown ball position
- Regularization necessary

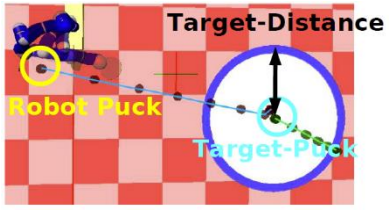


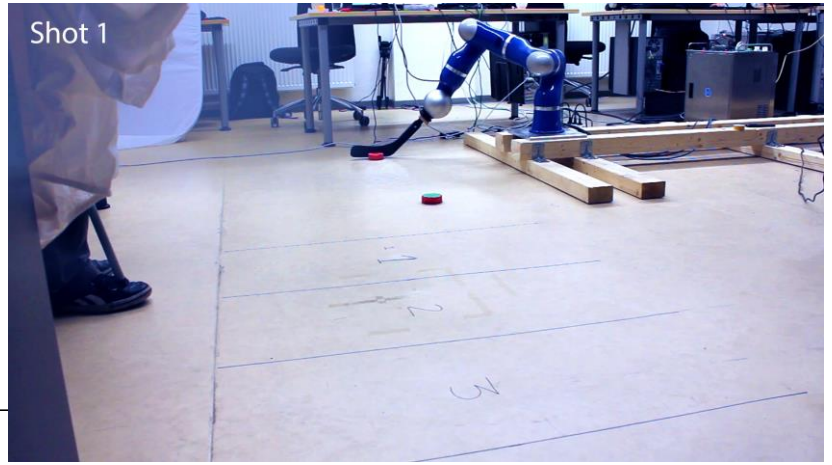
V. Tangkaratt et al., “Policy search with high-dimensional context variables”, in *AAAI Conference on Artificial Intelligence (AAAI)*, 2017.

Robot-Hockey:

- Position of target puck
- Desired displacement





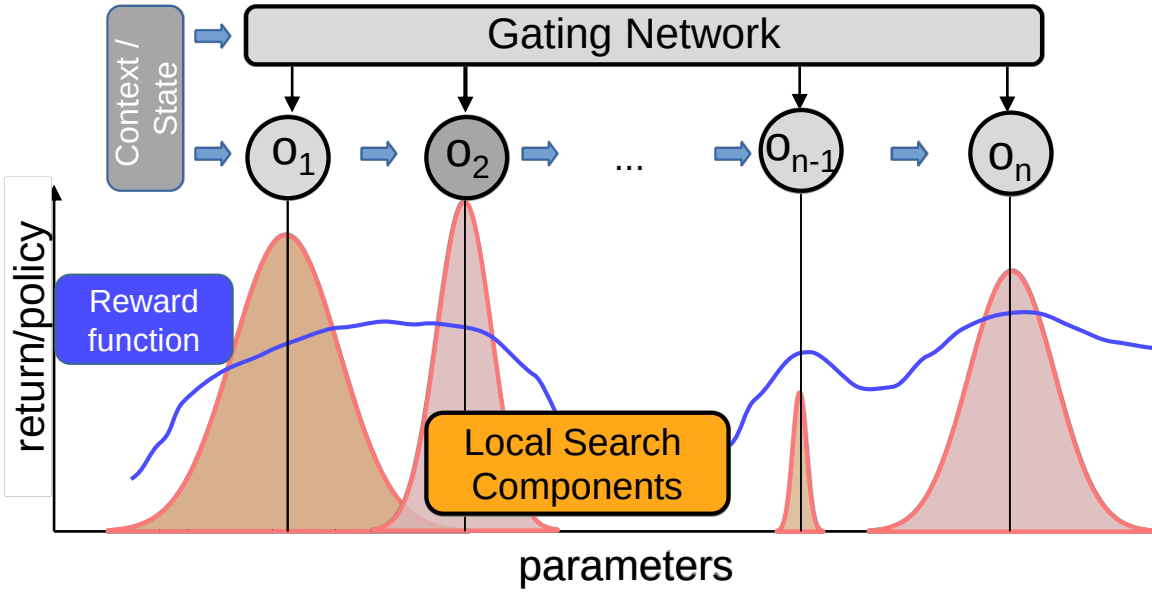


Versatile Policies for Multi-Modal Solutions

1. Hierarchical Policy:

- **Gating Policy:** $\pi(o|\boldsymbol{s})$
  - Select skill *o* given context
- **Local Sub-policies**: $\pi(\boldsymbol{\theta}|\boldsymbol{s}, o)$
  - Select parameters given context

$$\pi(\boldsymbol{\theta}|\boldsymbol{s}) = \sum_o \pi(o|\boldsymbol{s})\pi(\boldsymbol{\theta}|\boldsymbol{s}, o)$$



2. Maximum Entropy RL

[Harnoja et al, NIPS 2017]

$$J = \sum_o \pi(o) \int \pi(\boldsymbol{\theta}|o) R(\boldsymbol{\theta}) d\boldsymbol{\theta} \underbrace{- \sum_o \pi(o) \int \pi(\boldsymbol{\theta}|o) \log \pi(\boldsymbol{\theta})}_{H(\pi)}$$

➡ **intractable**

Variational Lower Bound

The objective J can be decomposed into:

Variational Distribution

$$J = \underbrace{U(\pi, \tilde{p}(o|\boldsymbol{\theta}))}_{\text{Lower Bound}} + \mathrm{KL}(p(o|\boldsymbol{\theta}) || \underbrace{\tilde{p}(o|\boldsymbol{\theta})}_{\text{Var. Dist.}})$$

- E-step: Tighten lower bound

$$\tilde{p}(o|\boldsymbol{\theta}) = p(o|\boldsymbol{\theta}) \quad \Rightarrow \quad \mathrm{KL}(p(o|\boldsymbol{\theta})||\tilde{p}(o|\boldsymbol{\theta})) = 0$$

- M-Step: Maximize lower bound

$$U(\pi, \tilde{p}(o|\boldsymbol{\theta})) = \sum_o \pi(o) \int \pi(\boldsymbol{\theta}|o)\big(R(\boldsymbol{\theta}) + \log \tilde{p}(o|\boldsymbol{\theta})\big) d\boldsymbol{\theta} + H(\pi(\boldsymbol{\theta}|o)) + H(\pi(o))$$

Yields decomposable max-ent RL problems with modified reward:

- **Gating:**

$$r(o) = \mathbb{E}_{\pi(\boldsymbol{\theta}|o)}\left[R(\boldsymbol{\theta}) + \log \tilde{p}(o|\boldsymbol{\theta})\right] + H(\pi(\boldsymbol{\theta}|o))$$

- **Parameter policy:**

$$r_o(\boldsymbol{\theta}) = R(\boldsymbol{\theta}) + \log \tilde{p}(o|\boldsymbol{\theta})$$

Application to Variational Inference

Test on "simple application" first: can we fit multi-modal target distributions?

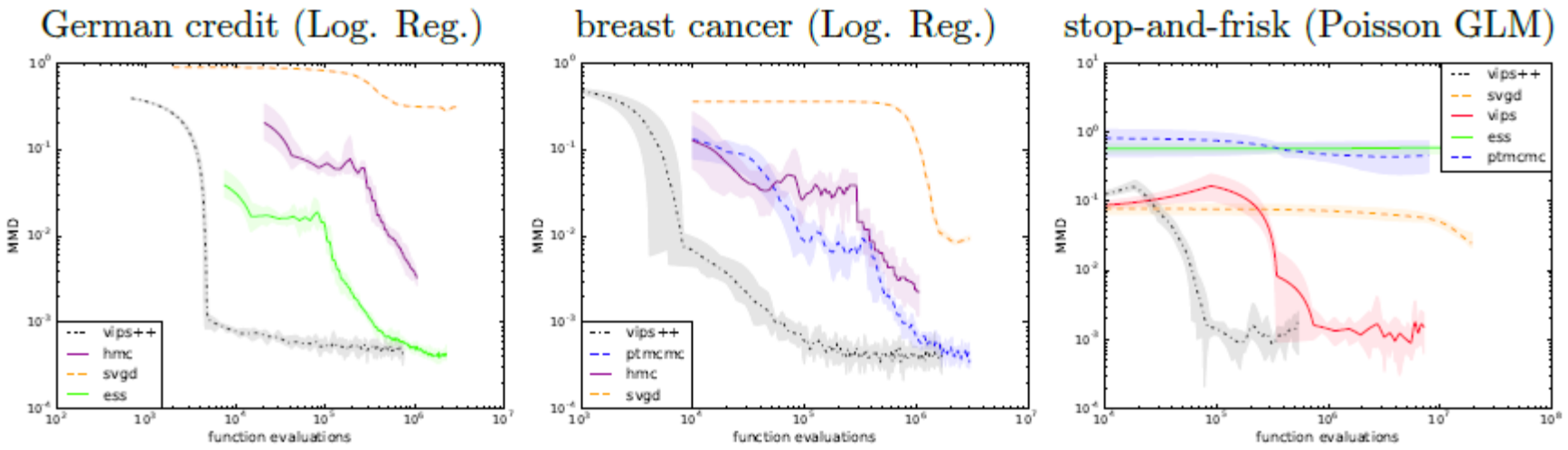
- Instead of maximizing the reward, we can minimize the I-projection

$$\arg\min_p \mathrm{KL}\left(p||p^*\right) = \int p(\boldsymbol{x}) \log p^*(\boldsymbol{x}) + H(p)$$

- Same information geometric-methods can be used [Arenz & Neumann, ICML 2018]

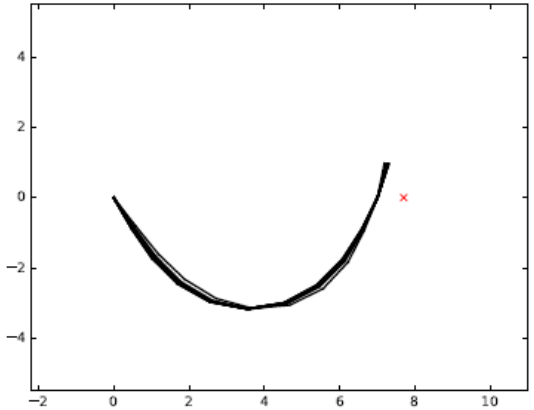
✓ Efficient

✓ Can find multiple modes

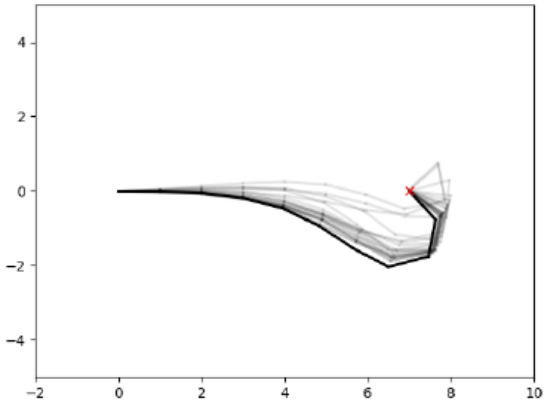


Robotic Inspired Tasks

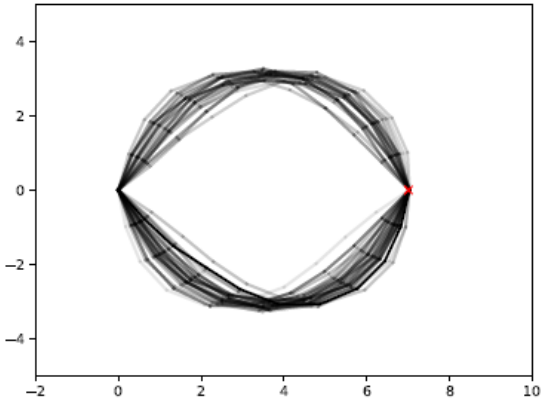
Planar 10-link robot arm:



NPVI

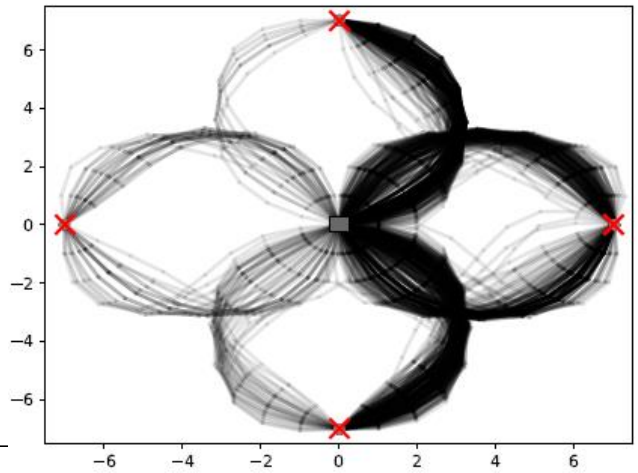


Varational Boosting



VIPS

4 Goals:



- ✓ Can find close to all solutions
- ✓ Policy search beats classical VI (faster, more sample efficient, much better approximations)

Learning multiple skills

Many tasks can be performed in multiple ways

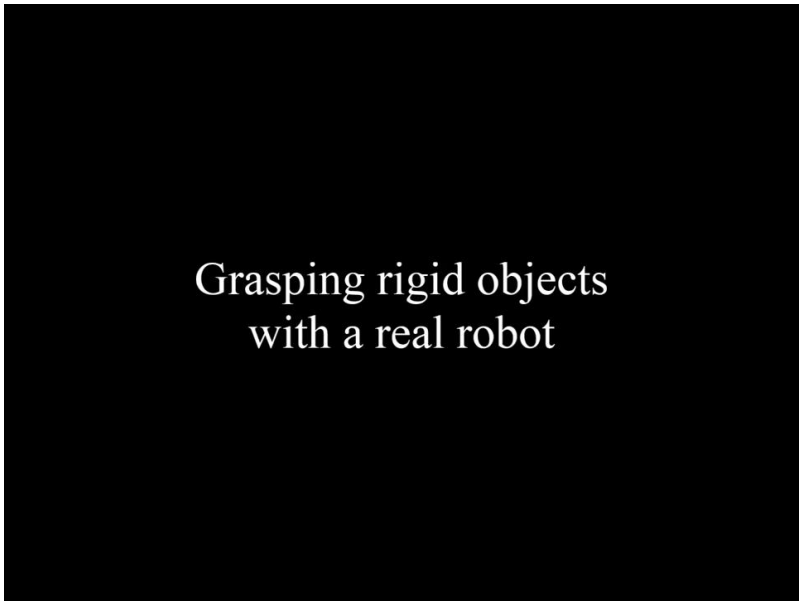
Table-Tennis:

- **Context:** Initial ball configuration
- **Skill Library:** Forehand and Backhand
- **Skill Parametrization:** Movement primitives

F. End, R. Akrour, **G. Neumann**, *Layered Policy Search for Learning Hierarchical Skills,* ICRA 2017

Multiple Grasp Types:

- **Context:** Local point-cloud features
- **Skill Library:** Grasp type
- **Skill Parametrization:** Grasp configuration



T,.Osa, J. Peters, **G. Neumann (2016)**: Experiments with Hierarchical Reinforcement Learning for Learning Multiple Grasp Types, ISER 2016

Policy Search for Grasping

Parameters encode:

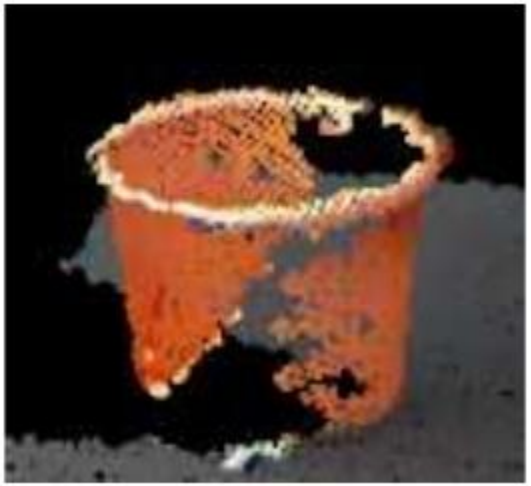
- Pre-Grasp configuration
- Grasp approach direction

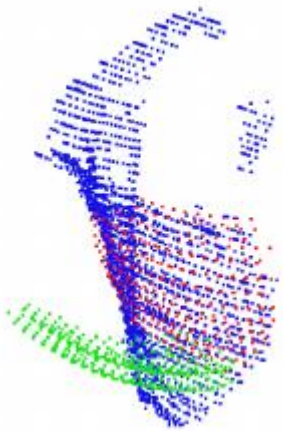
Context: 3D point cloud ?

- Very high dimensional input space
- Incomplete
- Unsuitable for context

However:

- Only **local features** at contact are important for a specific grasp
- **Grasp candidate:** contact region per finger
- Each grasp candidate has **individual context vector**





Context Generation

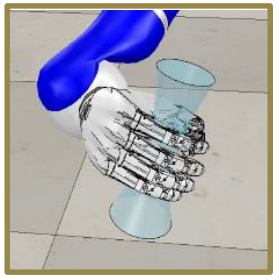
Grasp Library

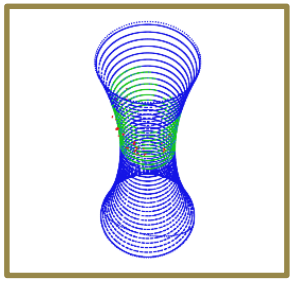
- Multiple Grasp Types
- Multiple Demonstrations

Contact Library

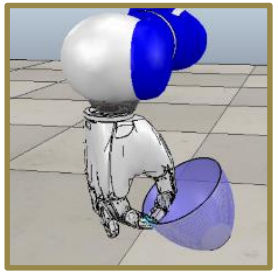
- Individual library per type *k*
- Store contact neighborhood

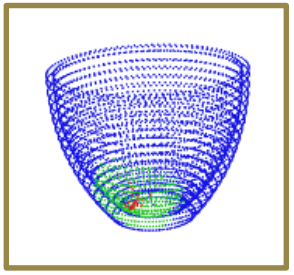
Power Grasp





Precision Grasp

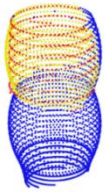


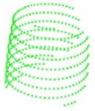


New Object

Find grasp candidates

- Search local points similar to contact
- Iterative Closest Point (ICP) Algorithm
- Multiple candidates *i* per type *k*





Extract Grasp Context

- Compute context $\boldsymbol{s}_i^{[k]}$ for each candidate i
- Characterizes grasp

For each finger:

- Contact centroid
- Contact normal

$$\boldsymbol{s}_i^{[k]} \in \mathbb{R}^{6\times \#\text{fingers}}$$

Set of **grasp contexts** per type $\mathcal{S}^{[k]} = \left\{\boldsymbol{s}_1^{[k]}, \ldots, \boldsymbol{s}_n^{[k]}\right\}$

Hierarchical Policy Search

Set of grasp canditates $\mathcal{S}^{[k]} = \left\{ \boldsymbol{s}_1^{[k]}, \ldots, \boldsymbol{s}_n^{[k]} \right\}$

Upper Level

Bayesian Optimization

Grasp Selection:

- Learn reward model per type: $R_k(\boldsymbol{s}) \approx \mathcal{GP}\big(0, k(\boldsymbol{s}, \boldsymbol{s}')\big)$
- Evaluate mean and variance of candidates: $[R_i^{[k]}, \sigma_i^{[k]}] = R_k\big(\boldsymbol{s}_i^{[k]}\big)$
- Select with Upper Confidence Bounds (UCB): $[k^*, i^*] = \arg\max_{[k,i]} R_i^{[k]} + \lambda \sigma_i^{[k]}$

Lower Level

Contextual Policy Search

Grasp Adaptation: Type 1

- Learn kernelized contextual policy
- Use 12 states as prototypes

$$\pi_1(\boldsymbol{\theta}|\boldsymbol{s}) = \mathcal{N}\left(\sum_j \boldsymbol{\alpha}_{1,j} k(\boldsymbol{s}, \boldsymbol{s}_j), \boldsymbol{\Sigma}_1 \right)$$

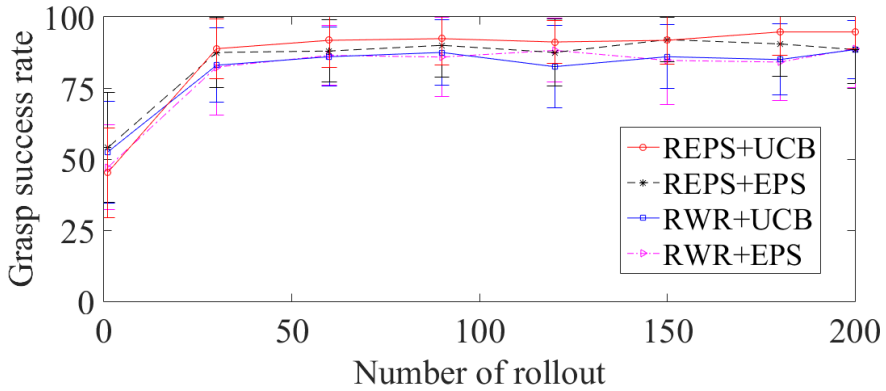
...

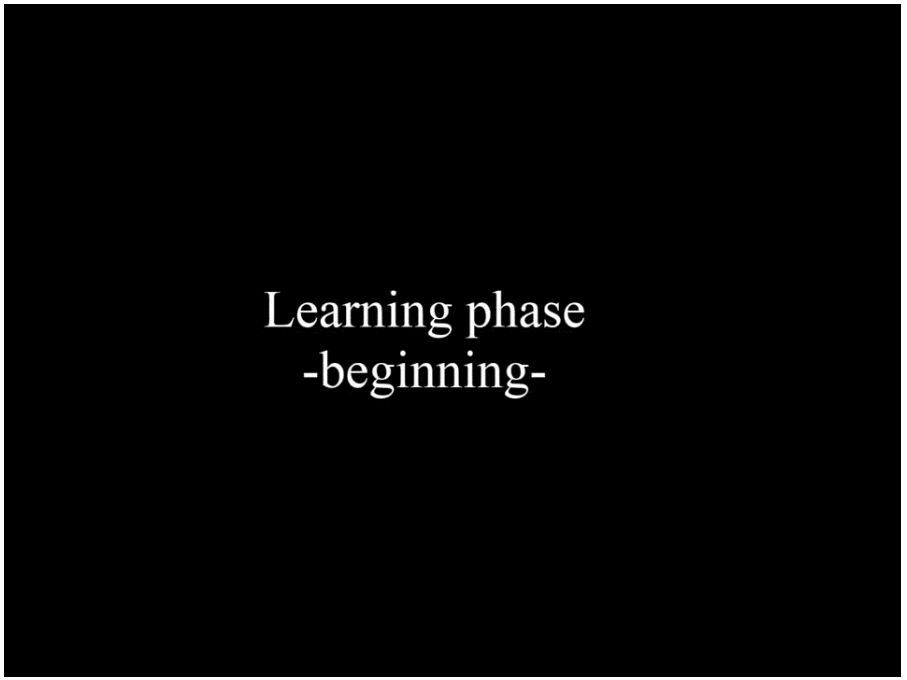
Grasp Adaptation: Type k

- Learn kernelized contextual policy

$$\pi_k(\boldsymbol{\theta}|\boldsymbol{s}) = \mathcal{N}\left(\sum_j \boldsymbol{\alpha}_{k,j} k(\boldsymbol{s}, \boldsymbol{s}_j), \boldsymbol{\Sigma}_k \right)$$

Performance in simulation

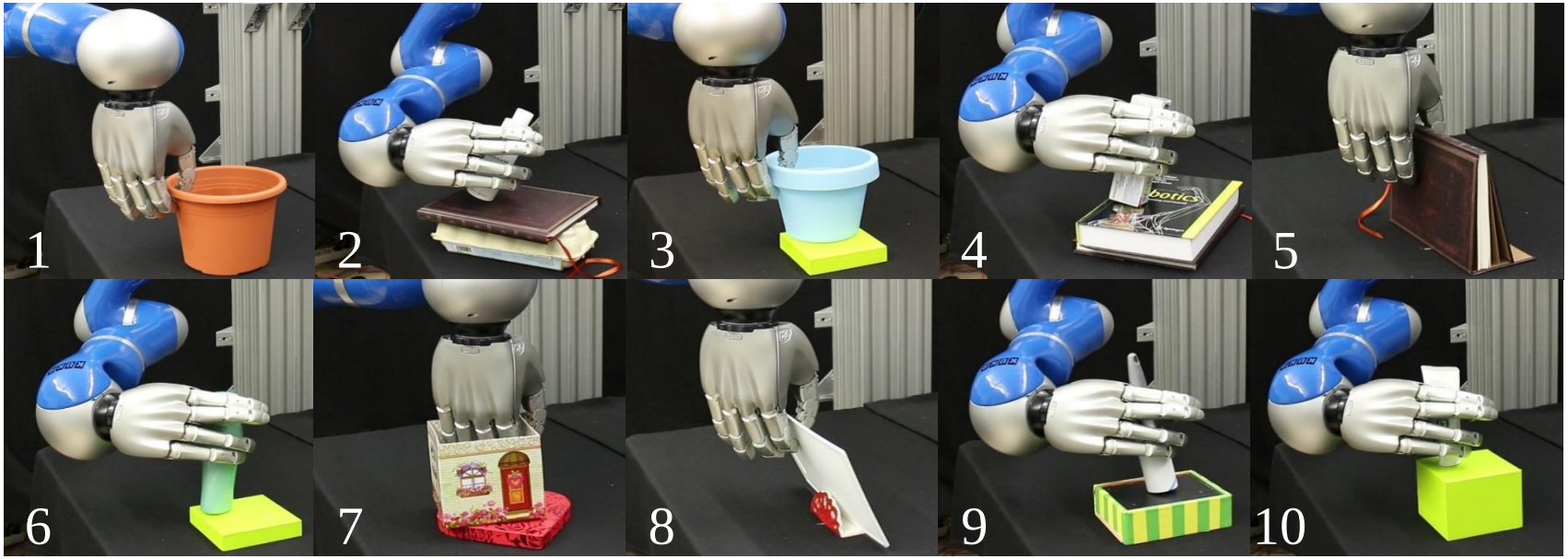




- **Reward:**
  - Force closure
  - Success of lifting
- 12 initial demonstrations per grasp type
- 50% successrate before learning
- 92.5 % successrate after learning

$$s = [c_{\text{thumb}}, n_{\text{thumb}}, c_{\text{index}}, n_{\text{index}}]$$

Testing on a real robot system

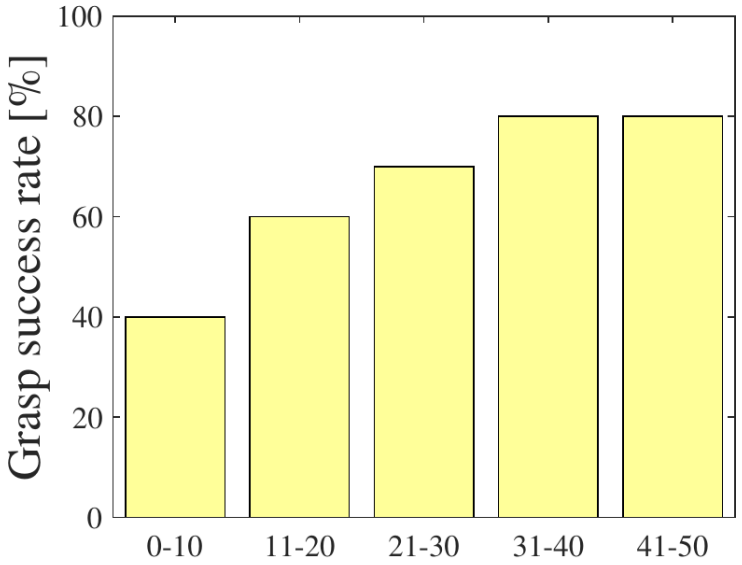


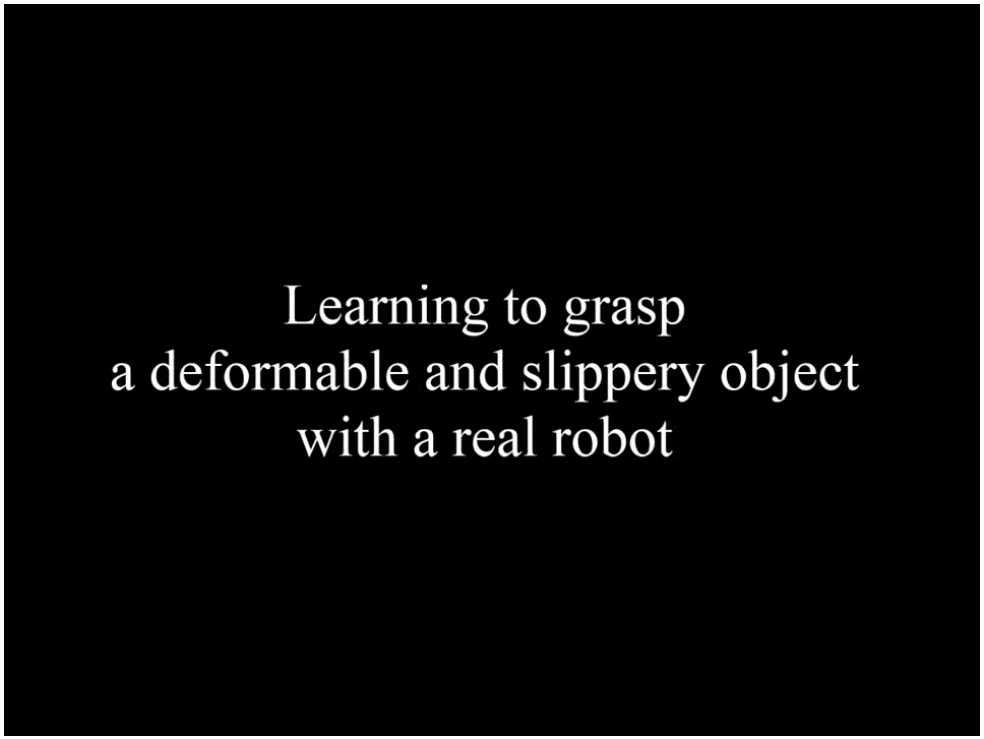
Object No.	1	2	3	4	5	6	7	8	9	10	Total
Success rate	5/5	5/5	4/5	3/5	5/5	5/5	4/5	5/5	5/5	5/5	92.0%

Learning on a real robot system

Many objects are deformable/slippery

- Can we also learn grasp policies in such case?





Application to Deep RL

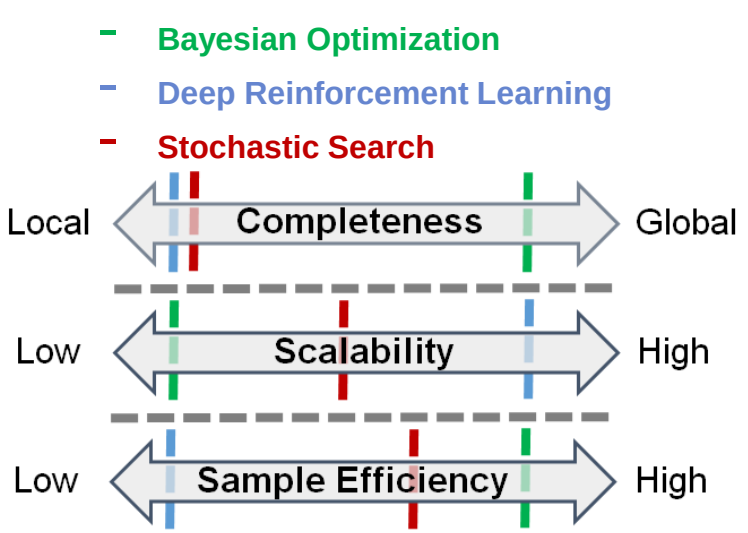
Deep Policies: $\pi(\boldsymbol{a}|\boldsymbol{s}) \propto \exp(\boldsymbol{\phi}_{\boldsymbol{\beta}}(\boldsymbol{s},\boldsymbol{a})^T\boldsymbol{\theta})$

- $\boldsymbol{\phi}_{\boldsymbol{\beta}}(\boldsymbol{s},\boldsymbol{a})$ … neural network with params $\boldsymbol{\beta}$
- $\boldsymbol{\theta}$ … log-linear parameters

Objective:

$$\pi^*_{\text{new}} = \text{argmax}_\pi \mathbb{E}_{\mu_{\pi_{\text{old}}}(\boldsymbol{s})}\left[\int \pi(\boldsymbol{a}|\boldsymbol{s}) A^{\pi_{\text{old}}}(\boldsymbol{s},\boldsymbol{a}) d\boldsymbol{a}\right]$$

- where $A^{\pi_{\text{old}}}(\boldsymbol{s},\boldsymbol{a})$ is the advantage function of the old policy



Some SOTA algorithms use similar trust regions (but use approximations to solve it)

- TRPO[Schulman2015], PPO[Schulman2016], MPO[Abdolmaleki2018]

Some SOTA algorithms use similar entropy regularization (not a bound)

- SAC[Harnoja2017], AC3

Can we unify this using our framework?

Compatible Policy Search

Compatible policy search (COPOS):

$$\pi^*_{\text{new}} = \text{argmax}_\pi \mathbb{E}_{\mu_{\pi_{\text{old}}}(\boldsymbol{s})} \left[\int \pi(\boldsymbol{a}|\boldsymbol{s}) A^{\pi_{\text{old}}}(\boldsymbol{s}, \boldsymbol{a}) d\boldsymbol{a} \right]$$

$$\text{s.t.:} \quad \mathbb{E}_{\mu_{\pi_{\text{old}}}(\boldsymbol{s})} \left[\text{KL}\left(\pi(\cdot|\boldsymbol{s}) || \pi_{\text{old}}(\cdot|\boldsymbol{s}) \right) \right] \leq \epsilon$$

$$H(\pi_{\text{old}}) - H(\pi) \leq \gamma$$

Solution:

$$\pi(\boldsymbol{a}|\boldsymbol{s}) \propto \pi_{\text{old}}(\boldsymbol{a}|\boldsymbol{s})^{\frac{\eta}{\eta+\omega}} \exp\left(\frac{A(\boldsymbol{s}, \boldsymbol{a})}{\eta + \omega} \right)$$

- Not a direct solution as it is hard to sample from this policy
- Yet, we can exploit special (compatible) structure of A(s,a)

J. Pajarinen, L. Hong Thai, R. Arkour, J. Peters, G. Neumann, *Compatible Natural Gradient Policy Search, Machine Learning, 2019*

Compatible Policy Search

Solution (for log linear case):

$$\pi(\boldsymbol{a}|\boldsymbol{s}) \propto \pi_{\text{old}}(\boldsymbol{a}|\boldsymbol{s})^{\frac{\eta}{\eta+\omega}} \exp\left(\frac{A(\boldsymbol{s},\boldsymbol{a})}{\eta+\omega}\right)$$

1. Move π_{old} into exp:

$$\pi(\boldsymbol{a}|\boldsymbol{s}) \propto \exp\left(\frac{\eta \log \pi_{\text{old}}(\boldsymbol{a}|\boldsymbol{s}) + A(\boldsymbol{s},\boldsymbol{a})}{\eta+\omega}\right)$$

2. Log Linear Model:

$$\pi(\boldsymbol{a}|\boldsymbol{s}) \propto \exp\left(\frac{\eta \boldsymbol{\phi}_{\boldsymbol{\beta}}(\boldsymbol{s},\boldsymbol{a})^T \boldsymbol{\theta}_{\text{old}} + A(\boldsymbol{s},\boldsymbol{a})}{\eta+\omega}\right)$$

3. Compatible approximation:

$$\pi(\boldsymbol{a}|\boldsymbol{s}) \propto \exp\left(\boldsymbol{\phi}_{\boldsymbol{\beta}}(\boldsymbol{s},\boldsymbol{a})^T \frac{\eta\boldsymbol{\theta}_{\text{old}} + \boldsymbol{w}_A}{\eta+\omega}\right)$$

Log-Linear Models:

$$\log \pi_{\text{old}}(\boldsymbol{a}|\boldsymbol{s}) = \boldsymbol{\phi}_{\beta}(\boldsymbol{s},\boldsymbol{a})^T \boldsymbol{\theta}_{\text{old}} - \log Z$$

Compatible Function Approximation

$$A(\boldsymbol{s},\boldsymbol{a}) \approx \boldsymbol{\phi}_{\beta}(\boldsymbol{s},\boldsymbol{a})^T \boldsymbol{w}_A$$

- For **Gaussians:** all **quadratic**, **linear** and **constant** terms

Compatible Policy Update:

$$\boldsymbol{\theta}_{\text{new}} = \frac{\eta\boldsymbol{\theta}_{\text{old}} + \boldsymbol{w}_A}{\eta+\omega}$$

- Can also be obtained for non-linear term using Taylor expansion
- Equivalent to natural gradients if omega = 0 (no entropy bound)

J. Pajarinen, L. Hong Thai, R. Arkour, J. Peters, G. Neumann, *Compatible Natural Gradient Policy Search, Machine Learning, 2019*

Compatible Policy Search

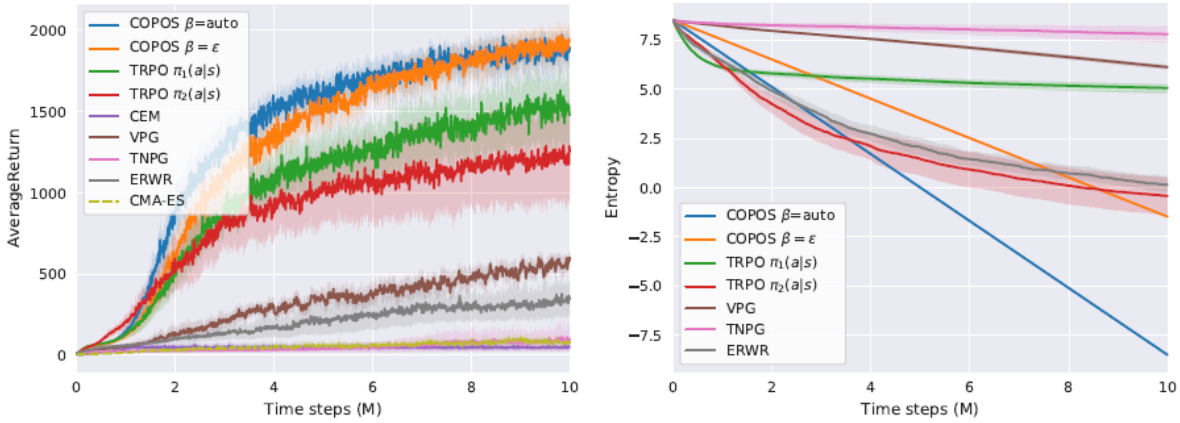
COPOS:

- Can compute KL and entropy constraints in closed form
- Entropy can be directly controlled
- Leads to improved exploration performance

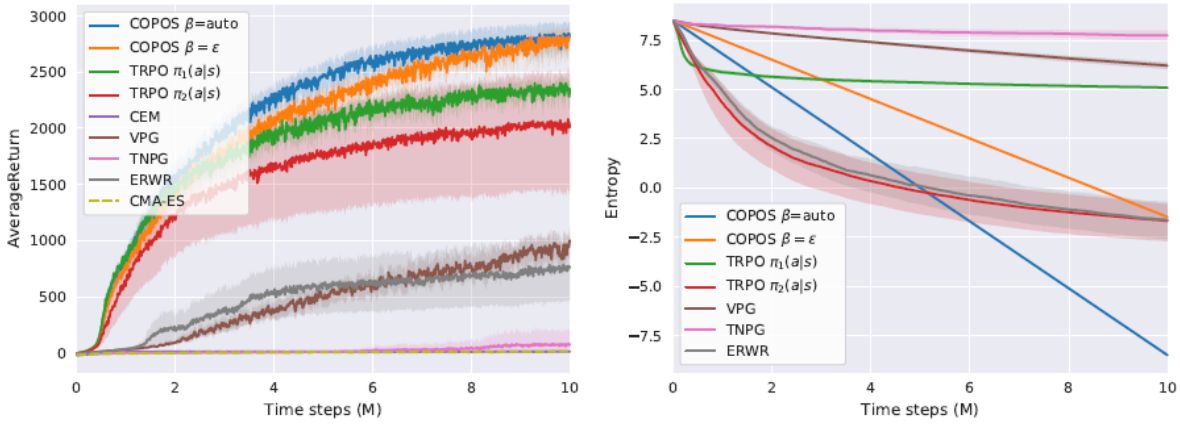
But:

- Still uses approximations for non-linear part of NN!
- Compatible approximation quite complex to implement

(b) RoboschoolHopper-v1



(c) RoboschoolWalker2d-v1

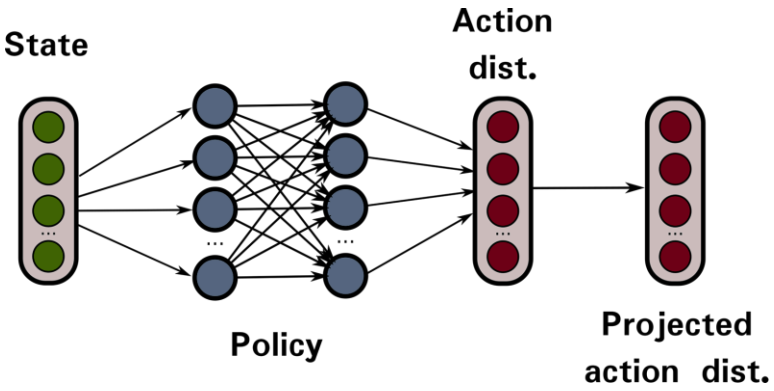


(d) RoboschoolHalfCheetah-v1

Projections in Policy search

Can we ”control” the entropy of the policy in a simpler way?

- Projections hard constraining Shannon entropy of Gaussian or soft-max policies

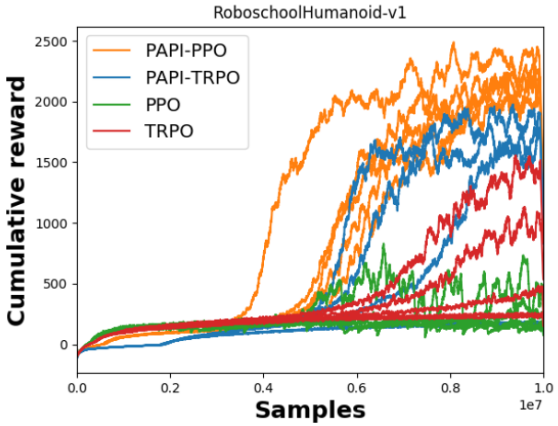


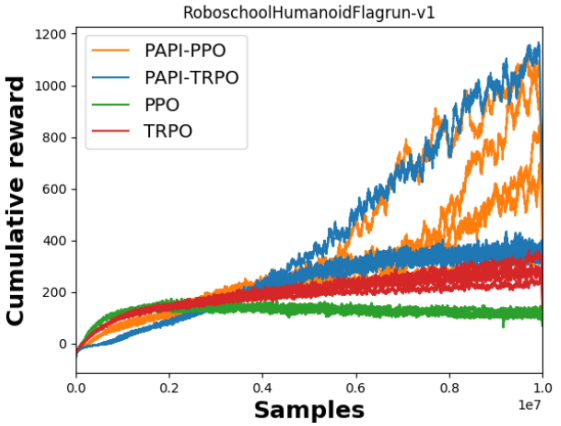
- Project solution of RL algorithm on the solution space given by the constraint
- Can be used on top of any other RL algorithm (TRPO, PPO,…)
- Projections that outperform other KL-constrained optimizers used in deep RL

Riad Akrour, Joni Pajarinen, Gerhard Neumann, Jan Peters, “Projections for Approximate Policy Iteration Algorithms”

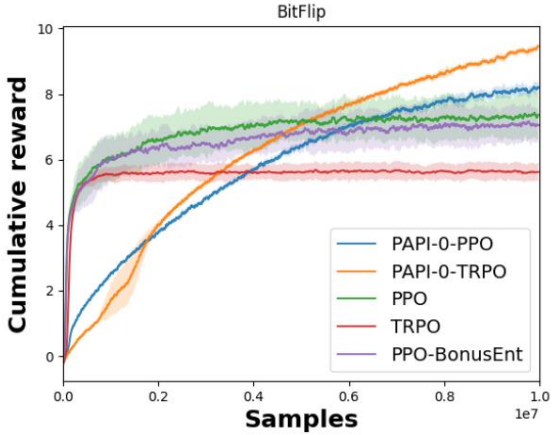
Results for PAPI

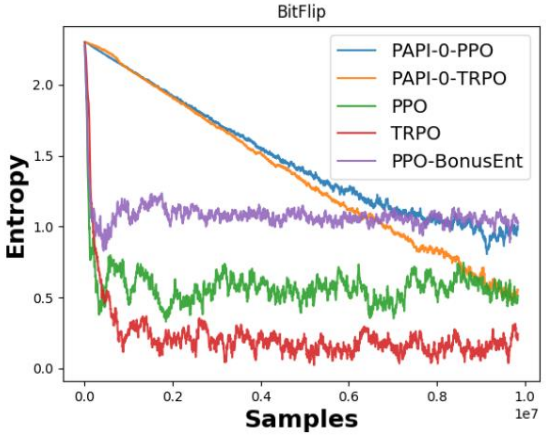
Mojoco:





Complex exploration problem (bit flip):



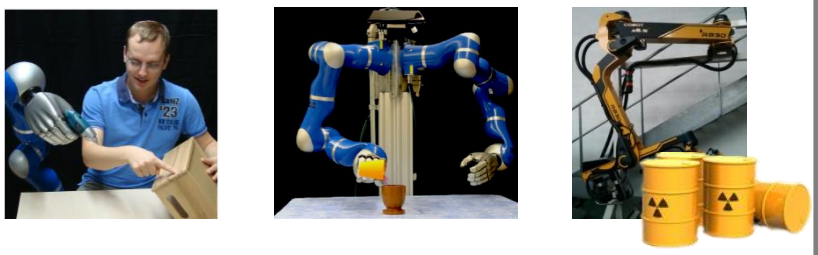


Conclusion: Motor Skill Learning

Information-Geometric Motor Skill Learning:

- Efficient local search with Gaussian distributions
  - Entropy Loss Regularization
  - Compatible Function Approximation
- Skill generalization using contextual search
- Local point cloud features for grasp adaptation
- Hierarchical policy for grasp selection
- **Extends to deep RL:** Controlling entropy is key for good exploration!

How can we accomplish these challenges?



Machine Learning

Data-Driven Algorithms

Autonomous Robots

Domain Knowledge

Data-driven movement primitives	× Flexible Skills
Reinforcement Learning	× Self-improvement
Multi-Agent Learning	× Cooperation
Representation Learning	× **Perception**

Learning in Robot Swarms

Swarm Systems:

- Large number of heterogeneous agents
- Permutation invariance (no order of agents)
- Invariant to exact number of agents

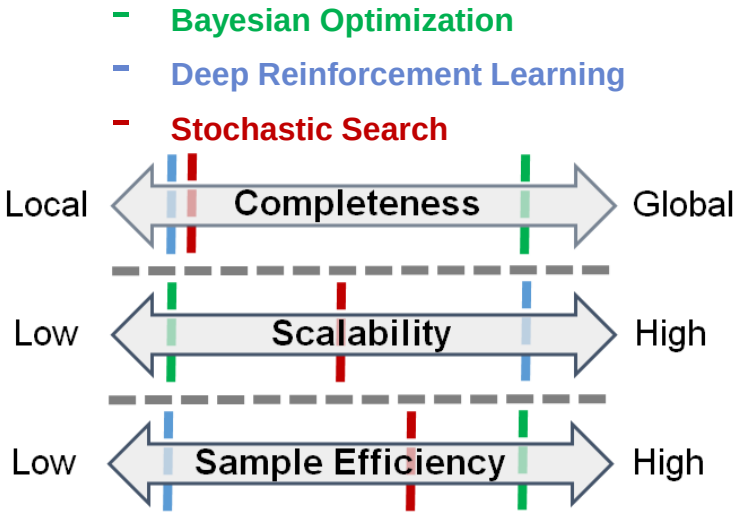
Deep Reinforcement Learning:

- Trust Region Policy Optimization (TRPO):
- Use transitions from all agents to estimate s_i gradient / Policies are shared across agents

✓ Can deal with complex observations

× Can not deal with varying number of agents

× Can not deal with permutation invariance



Learning in Robot Swarms

Swarm Systems:

- Large number of heterogeneous agents
- Learn decentralized control policies
- Each agent is described by state $\boldsymbol{s}_i$

Local Observation Model: Agent can observe (relative) state of their neighbors

$$o^{i,j} = f(\boldsymbol{s}_i, \boldsymbol{s}_j)$$ E.g., distance, bearing to neighbours

Local Communication Model: Agents can communicate further information about the interaction graph

$$o^{i,j} = f(\boldsymbol{s}_i, \boldsymbol{s}_j, \mathcal{G})$$ E.g., number of neighbors of agent j

Observation Set:

$$O^i = \{o^{i,j} | j \in \mathcal{N}(i)\}$$

Communication Models

Local Observation Model: Agent can observe (relative) state of their neighbors

$$o^{i,j} = f(\boldsymbol{s}_i, \boldsymbol{s}_j)$$ E.g., distance, bearing to neighbours

Local Communication Model: Agents can communicate further information about the interaction graph

$$o^{i,j} = f(\boldsymbol{s}_i, \boldsymbol{s}_j, \mathcal{G})$$ E.g., number of neighbors of agent j

Observation Set:

$$O^i = \{o^{i,j} | j \in \mathcal{N}(i)\}$$

- Similar properties than samples of distribution

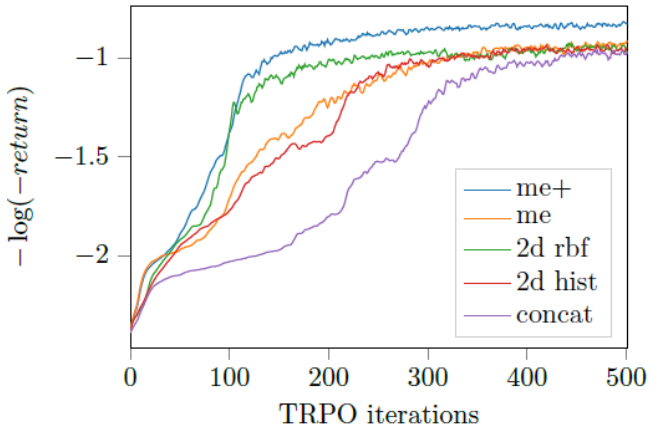
Mean Embedding of distribution $p_i(o|s)$: $\mu_o^i = \frac{1}{|O^i|} \sum_{o^{i,j} \in O^i} \varphi(o^{i,j})$

Feature vector: $\varphi(o)$

- Defined by kernels: $k(o_i, o_j) = \langle \varphi(o_i), \varphi(o_j) \rangle$
  [Gebhard & Neumann, ICRA 2018]
- Defined by neural networks: $\varphi(o) = \phi_{\boldsymbol{\theta}}(o)$
  [Huettenrauch & Neumann, submitted to JMLR]

Used as input to neural network policy

Experiments: Pursuit Evasion

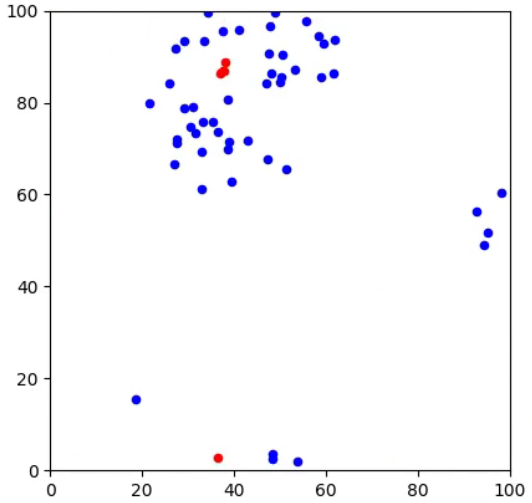


Conclusion:

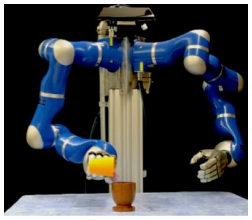
- New approach for “set inputs”
- Distribution-based representation of agents
- Scales to a large number of agents
- More potential applications

10 agents

50 agents / 5 evader



How can we accomplish these challenges?


Machine Learning

Data-Driven Algorithms

Autonomous Robots

Domain Knowledge

Data-driven movement primitives	× Flexible Skills
Reinforcement Learning	× Self-improvement
Multi-Agent Learning	× Cooperation
Representation Learning	× **Perception**

Recurrent Kalman Networks

Goal: State estimation for dynamical systems:

- Filtering
- Prediction

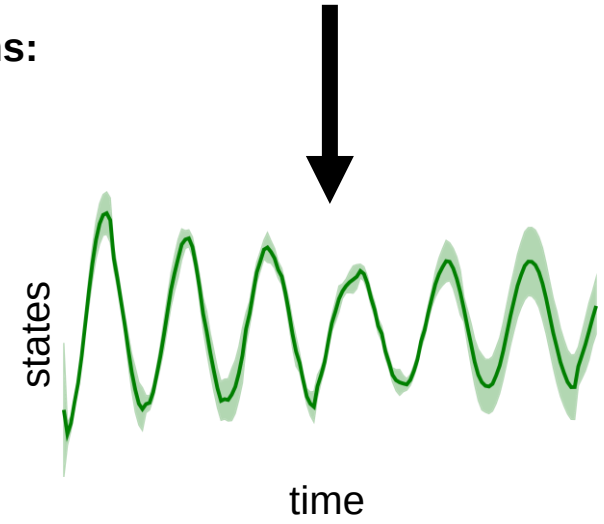
} First step for decision making under uncertainty

Challenges:

- High dimensional observations
- Partially observable
- Nonlinear dynamics
- Uncertainty

(Deep Learning) Solutions:

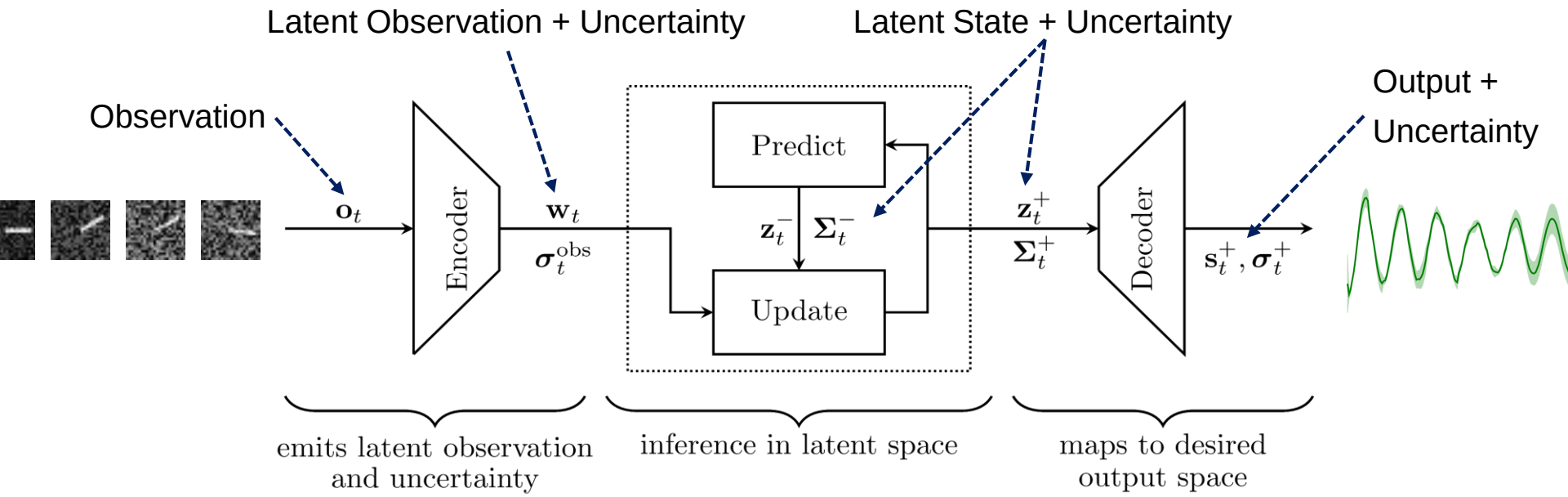
- CNNs
- RNNs
- Variational Inference (approximate)



How can we propagate uncertainty through RNNs without approximations?

- Recurrent cell based on Kalman filter

Overview



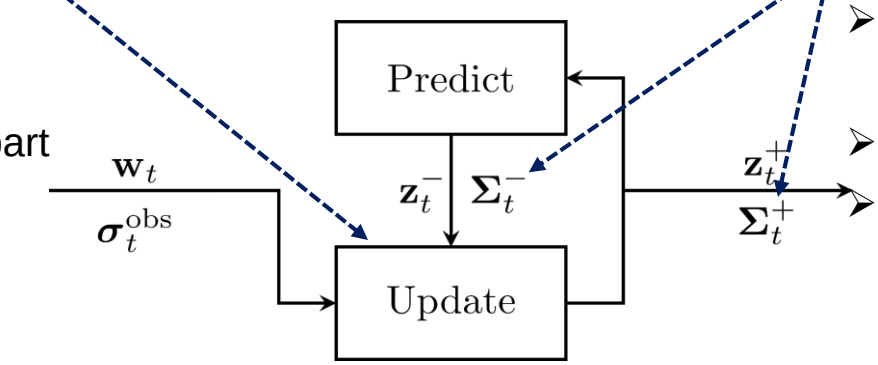
Make backpropagation through Kalman filter feasible?

- Locally linear transition models, even for highly nonlinear systems
- High dimensional latent spaces
- Factorized state representation to avoid expensive and unstable matrix inversions

Factorized State Representation

Observation Model

- $\mathbf{H} = \begin{pmatrix} \mathbf{I}_m & \mathbf{0}_{m \times m} \end{pmatrix}$
- Splits latent state
  1. Observable part
  2. Memory part



Factorized Representation

- $\boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\sigma}_u & \boldsymbol{\sigma}_s \\ \boldsymbol{\sigma}_s & \boldsymbol{\sigma}_l \end{pmatrix}$
- $\boldsymbol{\sigma}_u, \boldsymbol{\sigma}_l, \boldsymbol{\sigma}_s$ diagonal matrices
- $\boldsymbol{\sigma}_s$ correlates parts

Results in simplified Kalman Update

- No matrix inversion
- Instead only pointwise operations

} Makes inference and back-propagation feasible

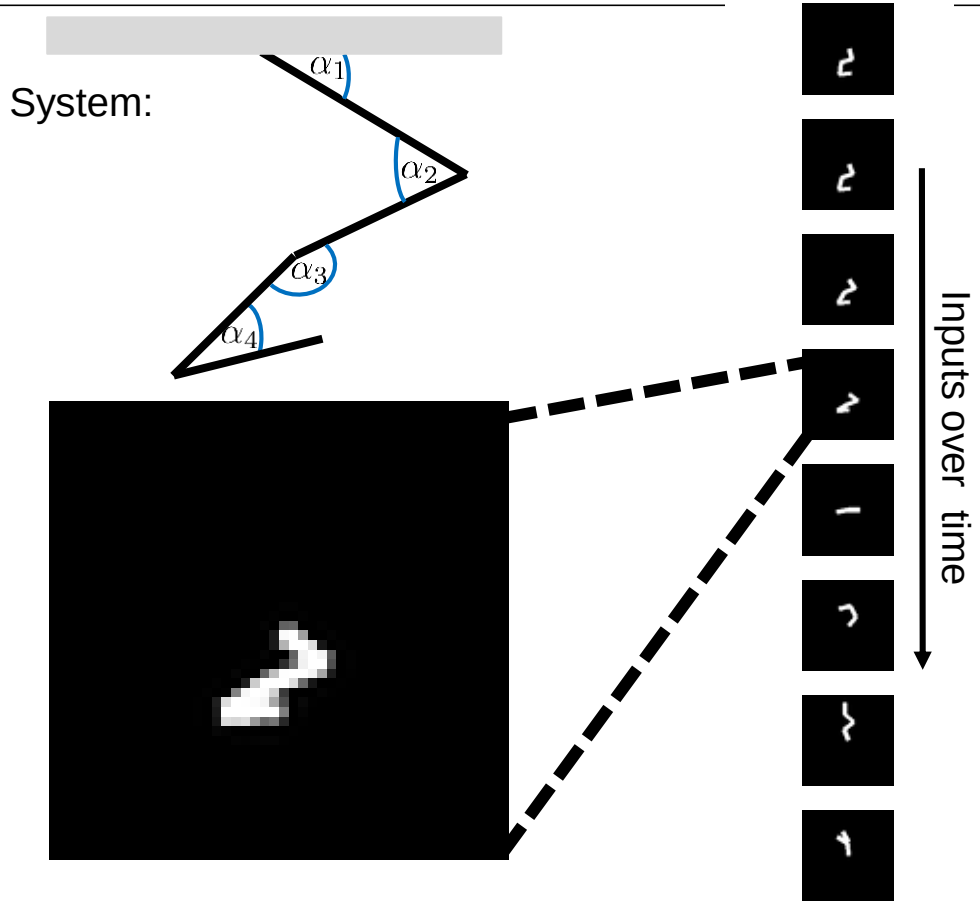
- Assumptions not restrictive since latent space is learned

Quad Link Pendulum

- State (4 joint angles + velocity)
- Highly nonlinear dynamics
- Links occlude each other
- Estimate joint angles of all 4 links

	RKN	LSTM	GRU
Log Likelihood	14.534	11.960	10.346
RMSE	0.103	0.118	0.121

- Significantly better uncertainty estimate (higher log-likelihood)
- Better prediction (smaller RMSE)



Summary & Conclusion

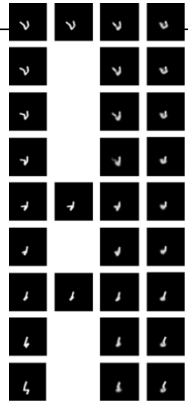
Recurrent Kalman Networks…

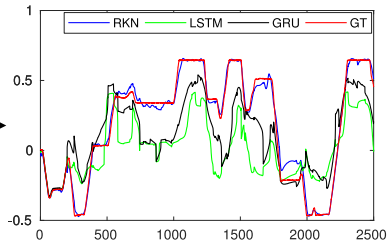
- … scale to real world systems
- … allow direct state estimation from images
- … use uncertainty in a principled manner to handle noise
- … can be trained end-to-end without approximations

… use as basic building block for decision making under uncertainty in the future

Additional Experiments

- Pendulum
- Image Imputation
- KITTI-Dataset for visual odometry
- Prediction for real pneumatic joint





- Comparison to recent approaches
- Code available

Conclusion

- Information-theoretic methods are powerful tools to formalize the RL problem
- **Direct Policy Search:**
  - Data-efficient but limited complexity (feedback, perception, …)
- **Deep RL:**
  - Data hungry but works with complex policies
- “Controlling” exploration is key for both approaches
- Still we need to close the “gap” and find the best of both worlds
- Using “algorithmic priors” in deep learning is a promising solution
  - Represent distributions: Mean embeddings
  - Represent uncertainty: Kalman Filtering

We are hiring!

New Robot Manipulation group at the University of Tuebingen:

- Funded by Bosch (BCAI). Competitive Bosch salary

Topics:

- Robot reinforcement learning
- Imitation learning
- Information theoretic methods
- Grasping and manipulation
- Robot vision
- Human-robot interaction

Open positions :

- 6 PhD positions
- 2 Research Scientist (permanent positions)
- Many more open positions (PhD + research scientist) @ BCAI headquarter in Renningen