Deutsches Forschungszentrum für Künstliche Intelligenz GmbH
Robotics Innovation Center

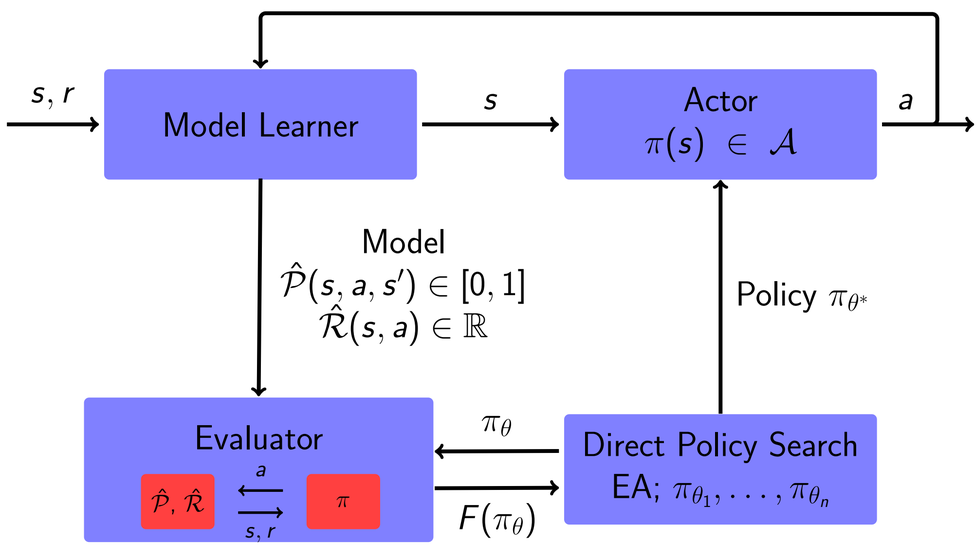
Model-based Direct Policy Search

Jan Hendrik Metzen and Frank Kirchner

Abstract

Direct Policy Search (DPS) denotes a class of Reinforcement Learning algorithms that allow to learn directly optimal policies without approximating their value functions. DPS has been particularly successful in continuous, noisy, and potentially non-markovian real-world domains. Unfortunately, DPS requires often a large number of interactions with the environment to learn a good policy. In this poster, we present a novel combination of DPS with model-based learning that significantly improves sample-efficiency.

Model-based Direct Policy Search



Model Learner

- Remember all transitions $T = \{(s_0, a_0, r_0, s'_0), \ldots, (s_n, a_n, r_n, s'_n)\}$, and partition for actions $T_a = \{(s_i, r_i, s'_i) | (s_i, a_i, r_i, s'_i) \in T \wedge a_i = a\}$
- Sample instances in state s with probability $P_{s,a} : T_a \mapsto [0,1]$, $P_{s,a}((s_i, r_i, s'_i)) = w_{s_i,s} / \sum_{(s_j,r_j,s'_j) \in T_a} w_{s_j,s}$ with $w_{s_1,s_2} = exp\left(-\left(\frac{||s_1 - s_2||_2}{b_s}\right)^2\right)$
- Sample successor state s' and reward r for applying action a in state s as: $s' = s + (s'_i - s_i)$ and $r = r_i$
- R_{max}-based exploration: $r = \begin{cases} R_{max} & \text{if } \sum_{(s_i,r_i,s'_i) \in T_a} w_{s_i,s} < T_{expl} \\ r_i & \text{else} \end{cases}$

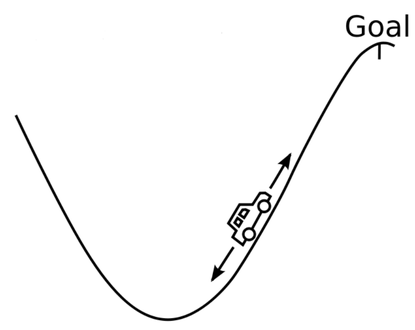
Direct Policy Search

- Given a class of policies Π parametrized by a vector θ
- Use metaheuristic to search in parameter space for a vector θ^* that maximizes $F(\pi_\theta)$
- MBDPS: Compute $F(\pi_{\theta^*})$ solely based on trajectories sampled from internal model

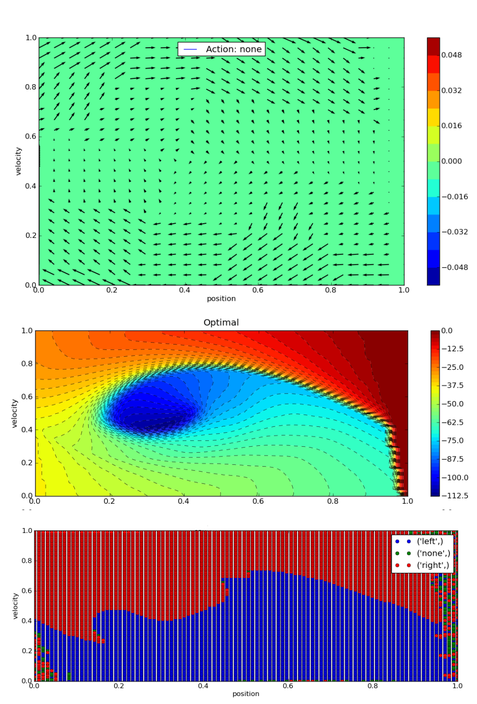
Evaluator

- For a given policy and start state distribution, use accumulated reward which is obtained when state transitions and rewards are sampled from model and actions are chosen based on policy as the policy's fitness estimate.

Benchmark: Mountain Car



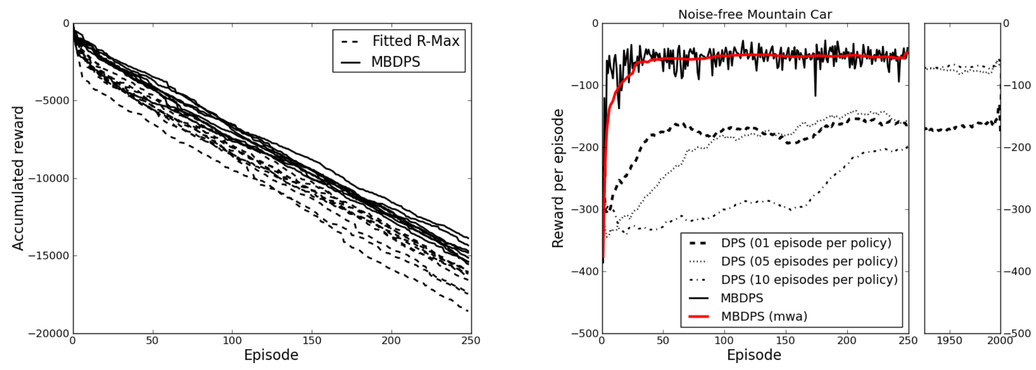
- 2 continuous state dimensions: position and velocity
- 3 discrete actions: accelerate left, right, and coast
- Reward -1 per time step
- Random start state, otherwise deterministic



Experimental Setup

- Deterministic, linear policy with bias: $\pi(s) = \arg\max_{a \in A}(\theta_a s + b_a)$
- Metaheuristic: Covariance Matrix Adaptation Evolution Strategy (CMA-ES)
- Each policy tested for one episode consisting of at most 500 steps
- Parameter setting: $b_S = 0.03$, $R_{max} = 0.0$, and $T_{expl} = 1.0$
- Open source implementation available in Maja Machine Learning Framework (`http://mmlf.sourceforge.net/`)

Results



References

- Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9:159–195, 2001.
- Nicholas K. Jong and Peter Stone. Model-based function approximation in reinforcement learning. In *Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1–8, Honolulu, Hawaii, 2007. ACM.
- Jan Hendrik Metzen and Mark Edgington. Maja Machine Learning Framework. `http://mmlf.sourceforge.net/`.

Universität Bremen

Contact:
Jan Hendrik Metzen
DFKI Bremen & University Bremen
Robotics Innovation Center
Director: Prof. Dr. Frank Kirchner
www.dfki.de/robotics
jhm@informatik.uni-bremen.de