Full text available at: http://dx.doi.org/10.1561/2300000021

A Survey on Policy Search for Robotics

A Survey on Policy Search for Robotics

Marc Peter Deisenroth

Technische Universität Darmstadt, Germany
and Imperial College London, UK
marc@ias.tu-darmstadt.de

Gerhard Neumann

Technische Universität Darmstadt,
Germany
neumann@ias.tu-darmstadt.de

Jan Peters

Technische Universität Darmstadt, Germany
and Max Planck Institute for Intelligent Systems,
Germany
peters@ias.tu-darmstadt.de

now
the essence of knowledge
Boston – Delft

Foundations and Trends® in Robotics

Published, sold and distributed by:
now Publishers Inc.
PO Box 1024
Hanover, MA 02339
USA
Tel. +1-781-985-4510
www.nowpublishers.com
sales@nowpublishers.com

Outside North America:
now Publishers Inc.
PO Box 179
2600 AD Delft
The Netherlands
Tel. +31-6-51115274

The preferred citation for this publication is M. P. Deisenroth, G. Neumann and J. Peters, A Survey on Policy Search for Robotics, Foundations and Trends® in Robotics, vol 2, nos 1–2, pp 1–142, 2011.

ISBN: 978-1-60198-702-0

Foundations and Trends® in Robotics

Volume 2 Issues 1–2, 2011

Editorial Board

Editorial Scope

Foundations and Trends® in Robotics publishes survey and tutorial articles in the following topics:

- Foundations
  - Mathematical modelling
  - Kinematics
  - Dynamics
  - Estimation Methods
  - Robot Control
  - Planning
  - Artificial Intelligence in Robotics
  - Software Systems and Architectures
- Mechanisms and Actuators
  - Kinematic Structures
  - Legged Systems
  - Wheeled Systems
  - Hands and Grippers
  - Micro and Nano Systems
- Sensors and Estimation
  - Force Sensing and Control
  - Haptic and Tactile Sensors
  - Proprioceptive Systems
  - Range Sensing
  - Robot Vision
  - Visual Servoing
  - Localization, Mapping and SLAM
- Planning and Control
  - Control of manipulation systems
  - Control of locomotion systems
  - Behaviour based systems
  - Distributed systems
  - Multi-Robot Systems
- Human-Robot Interaction
  - Robot Safety
  - Physical Robot Interaction
  - Dialog Systems
  - Interface design
  - Social Interaction
  - Teaching by demonstration
- Industrial Robotics
  - Welding
  - Finishing
  - Painting
  - Logistics
  - Assembly Systems
  - Electronic manufacturing
- Service Robotics
  - Professional service systems
  - Domestic service robots
  - Field Robot Systems
  - Medical Robotics

Information for Librarians

Foundations and Trends® in Robotics, 2011, Volume 2, 4 issues. ISSN paper version 1935-8253. ISSN online version 1935-8261. Also available as a combined paper and online subscription.

Foundations and Trends® in
Robotics
Vol. 2, Nos. 1–2 (2011) 1–142

DOI: 10.1561/2300000021

now
the essence of knowledge

A Survey on Policy Search for Robotics

Marc Peter Deisenroth[*,1], Gerhard Neumann[*,2] and Jan Peters[3]

[1] *Technische Universität Darmstadt, Germany, and Imperial College London, UK, marc@ias.tu-darmstadt.de*
[2] *Technische Universität Darmstadt, Germany, neumann@ias.tu-darmstadt.de*
[3] *Technische Universität Darmstadt, Germany, and Max Planck Institute for Intelligent Systems, Germany, peters@ias.tu-darmstadt.de*

Abstract

Policy search is a subfield in reinforcement learning which focuses on finding good parameters for a given policy parametrization. It is well suited for robotics as it can cope with high-dimensional state and action spaces, one of the main challenges in robot learning. We review recent successes of both model-free and model-based policy search in robot learning.

Model-free policy search is a general approach to learn policies based on sampled trajectories. We classify model-free methods based on their policy evaluation strategy, policy update strategy, and exploration strategy and present a unified view on existing algorithms. Learning a policy is often easier than learning an accurate forward model, and, hence, model-free methods are more frequently used in practice. However, for each sampled trajectory, it is necessary to interact with the

[*] Both authors contributed equally.

robot, which can be time consuming and challenging in practice. Model-based policy search addresses this problem by first learning a simulator of the robot's dynamics from data. Subsequently, the simulator generates trajectories that are used for policy learning. For both model-free and model-based policy search methods, we review their respective properties and their applicability to robotic systems.

Contents

1

Introduction

From simple house-cleaning robots to robotic wheelchairs and general transport robots the number and variety of robots used in our everyday life are rapidly increasing. To date, the controllers for these robots are largely designed and tuned by a human engineer. Programming robots is a tedious task that requires years of experience and a high degree of expertise. The resulting programmed controllers are based on assuming exact models of both the robot's behavior and its environment. Consequently, hard-coding controller for robots has its limitations when a robot has to adapt to new situations or when the robot/ environment cannot be modeled sufficiently accurately. Hence, there is a gap between the robots currently used and the vision of incorporating fully autonomous robots. In *robot learning*, machine learning methods are used to automatically extract relevant information from data to solve a robotic task. Using the power and flexibility of modern machine learning techniques, the field of robot control can be further automated, and the gap toward autonomous robots, e.g., for general assistance in households, elderly care, and public services can be narrowed substantially.

1.1 Robot Control as a Reinforcement Learning Problem

In most tasks, robots operate in a high-dimensional state space $\boldsymbol{x}$ composed of both internal states (e.g., joint angles, joint velocities, end-effector pose, and body position/orientation) and external states (e.g., object locations, wind conditions, or other robots). The robot selects its motor commands $\boldsymbol{u}$ according to a control policy π. The control policy can either be stochastic, denoted by $\pi(\boldsymbol{u}|\boldsymbol{x})$, or deterministic, which we will denote as $\boldsymbol{u} = \pi(\boldsymbol{x})$. The motor commands $\boldsymbol{u}$ alter the state of the robot and its environment according to the probabilistic transition function $p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{u}_t)$. Jointly, the states and actions of the robot form a *trajectory* $\boldsymbol{\tau} = (\boldsymbol{x}_0, \boldsymbol{u}_0, \boldsymbol{x}_1, \boldsymbol{u}_1, \dots)$, which is often also called a *rollout* or a *path*.

We assume that a numeric scoring system evaluates the performance of the robot system during a task and returns an accumulated reward signal $R(\boldsymbol{\tau})$ for the quality of the robot's trajectory. For example, the reward $R(\boldsymbol{\tau})$ may include a positive reward for a task achievement and negative rewards, i.e., costs, that punish energy consumption. Many of the considered motor tasks are stroke-based movements, such as returning a tennis ball or throwing darts. We will refer to such tasks as *episodic learning tasks* as the execution of the task, the *episode*, ends after a given number T of time steps. Typically, the accumulated reward $R(\boldsymbol{\tau})$ for a trajectory is given as

$$R(\boldsymbol{\tau}) = r_T(\boldsymbol{x}_T) + \sum_{t=0}^{T-1} r_t(\boldsymbol{x}_t, \boldsymbol{u}_t), \tag{1.1}$$

where r_t is an instantaneous reward function, which might be a punishment term for the consumed energy, and r_T is a final reward, such as quadratic punishment term for the deviation to a desired goal posture. For many episodic motor tasks the policy is modeled as time-dependent policy, i.e., either a stochastic policy $\pi(\boldsymbol{u}_t|\boldsymbol{x}_t, t)$ or a deterministic policy $\boldsymbol{u}_t = \pi(\boldsymbol{x}_t, t)$ is used.

In some cases, the infinite-horizon case is considered

$$R(\boldsymbol{\tau}) = \sum_{t=0}^{\infty} \gamma^t r(\boldsymbol{x}_t, \boldsymbol{u}_t), \tag{1.2}$$

where $\gamma \in [0,1)$ is a discount factor that discounts rewards further in the future.

Many tasks in robotics can be phrased as choosing a (locally) optimal control policy π^* that maximizes the expected accumulated reward

$$J_\pi = \mathbb{E}[R(\boldsymbol{\tau})|\pi] = \int R(\boldsymbol{\tau}) p_\pi(\boldsymbol{\tau}) d\boldsymbol{\tau}\,, \tag{1.3}$$

where $R(\boldsymbol{\tau})$ defines the objectives of the task, and $p_\pi(\boldsymbol{\tau})$ is the distribution over trajectories $\boldsymbol{\tau}$. For a stochastic policy $\pi(\boldsymbol{u}_t|\boldsymbol{x}_t,t)$, the trajectory distribution is given as

$$p_\pi(\boldsymbol{\tau}) = p(\boldsymbol{x}_0) \prod_{t=0}^{T-1} p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t,\boldsymbol{u}_t)\pi(\boldsymbol{u}_t|\boldsymbol{x}_t,t), \tag{1.4}$$

where $p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t,\boldsymbol{u}_t)$ is given by the system dynamics of the robot and its environment. For a deterministic policy, $p_\pi(\boldsymbol{\tau})$ is given as

$$p_\pi(\boldsymbol{\tau}) = p(\boldsymbol{x}_0) \prod_{t=0}^{T-1} p\left(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t,\pi(\boldsymbol{x}_t,t)\right). \tag{1.5}$$

With this general reinforcement learning (RL) problem setup, many tasks in robotics can be naturally formulated as *reinforcement learning* (RL) problems. However, robot RL poses three main challenges, which have to be solved: The RL algorithm has to manage (i) high-dimensional continuous state and action spaces, (ii) strong real-time requirements, and (iii) the high costs of robot interactions with its environment.

Traditional methods in RL, such as TD-learning [81], typically try to estimate the expected long-term reward of a policy for each state $\boldsymbol{x}$ and time step t, also called the *value function* $V_t^\pi(\boldsymbol{x})$. The value function is used to calculate the quality of an executing action $\boldsymbol{u}$ in state $\boldsymbol{x}$. This quality assessment is subsequently utilized to directly compute the policy by action selection or to update the policy π. However, value function methods struggle with the challenges encountered in robot RL, as these approaches require filling the complete state–action space with data. In addition, the value function is computed iteratively by the use of bootstrapping, which often results in a bias in the quality assessment of the state–action pairs if we need to resort to value function

approximation techniques as it is the case for continuous state spaces. Consequently, value function approximation turns out to be a very difficult problem in high-dimensional state and action spaces. Another major issue is that value functions are often discontinuous, especially when the non-myopic policy differs from a myopic policy. For instance, the value function of the under-powered pendulum swing-up is discontinuous along the manifold where the applicable torque is just not sufficient to swing the pendulum up [23]. Any error in the value function will eventually propagate through to the policy.

In a classical RL setup, we seek a policy without too specific prior information. Key to successful learning is the exploration strategy of the learner to discover rewarding states and trajectories. In a robotics context, arbitrary exploration is not desired if not discouraged since the robot can easily be damaged. Therefore, the classical RL paradigm in a robotics context is not directly applicable since exploration needs to take hardware constraints into account. Two ways of implementing cautious exploration are to either avoid significant changes in the policy [58] or to explicitly discourage entering undesired regions in the state space [22].

In contrast to value-based methods, *Policy Search* (PS) methods use parametrized policies $\pi_{\boldsymbol{\theta}}$. They directly operate in the parameter space $\boldsymbol{\Theta}$, $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, of parametrized policies, and typically avoid learning a value function. Many methods do so by directly using the experienced reward to come from the rollouts as quality assessment for state–action pairs instead of using the rather dangerous bootstrapping used in value function approximation. The usage of parametrized policies allows for scaling RL into high-dimensional continuous action spaces by reducing the search space of possible policies.

Policy search allows task-appropriate pre-structured policies, such as movement primitives [72], to be integrated straightforwardly. Additionally, imitation learning from an expert's demonstrations can be used to obtain an initial estimate for the policy parameters [59]. Finally, by selecting a suitable policy parametrization, stability and robustness guarantees can be given [11]. All these properties simplify the robot learning problem and permit the successful application of reinforcement learning to robotics. Therefore, PS is often the RL approach of

choice in robotics since it is better at coping with the inherent challenges of robot reinforcement learning. Over the last decade, a series of fast policy search algorithms have been proposed and shown to work well on real systems [7, 17, 22, 39, 54, 59, 87]. In this review, we provide a general overview, summarize the main concepts behind current policy search approaches, and discuss relevant robot applications of these policy search methods. We focus mainly on those aspects of RL that are predominant for robot learning, i.e., learning in high-dimensional continuous state and action spaces and a high data-efficiency and local exploration. Other important aspects of RL, such as the exploration–exploitation trade-off, feature selection, using structured models, or value function approximation, are not covered in this monograph.

1.2 Policy Search Taxonomy

Numerous policy search methods have been proposed in the last decade, and several of them have been used successfully in the domain of robotics. In this monograph, we review several important recent developments in policy search for robotics. We distinguish between model-free policy search methods (Section 2), which learn policies directly based on sampled trajectories, and model-based approaches (Section 3), which use the sampled trajectories to first build a model of the state dynamics, and, subsequently, use this model for policy improvement.

Figure 1.1 categorizes policy search into model-free policy search and model-based policy search and distinguishes between different policy update strategies. The policy updates in both model-free and model-based policy search (green blocks) are based on either policy gradients (PG), expectation–maximization (EM)-based updates, or information-theoretic insights (Inf.Th.). While all three update strategies are fairly well explored in model-free policy search, model-based policy search almost exclusively focuses on PG to update the policy.

Model-free policy search uses stochastic trajectory generation, i.e., the trajectories are generated by "sampling" from the robot $p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{u}_t)$ and the policy $\pi_{\boldsymbol{\theta}}$. This means, a system model is not explicitly required; we just have to be able to sample trajectories from the real robot. In the model-based case (right sub-tree), we can

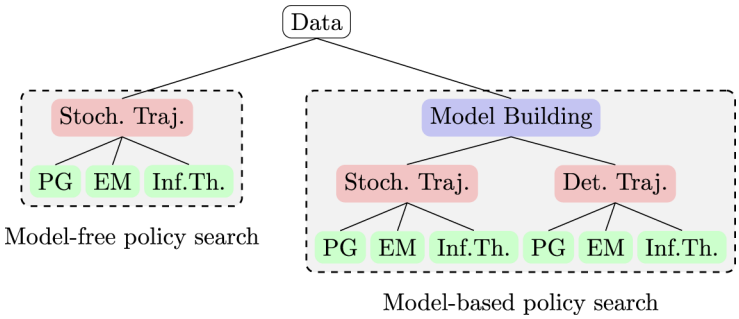

Fig. 1.1 Categorization of policy search into model-free policy search and model-based policy search. In the model-based case (right sub-tree), data from the robot is used to learn a model of the robot (blue box). This model is then used to generate trajectories. Here, we distinguish between stochastic trajectory generation and deterministic trajectory prediction. Model-free policy search (left sub-tree) uses data from the robot directly as a trajectory for updating the policy. The policy updates in both model-free and model-based policy search (green blocks) are based on either policy gradients (PG), expectation–maximization (EM)-based updates, or information-theoretic insights (Inf.Th.).

either use stochastic trajectory generation or deterministic trajectory prediction. In the case of stochastic trajectory generation, the learned models are used as simulator for sampling trajectories. Hence, learned models can easily be combined with model-free policy search approaches by exchanging the "robot" with the learned model of the robot's dynamics. Deterministic trajectory prediction does not sample trajectories, but analytically predicts the trajectory distribution $p_{\boldsymbol{\theta}}(\boldsymbol{\tau})$. Typically, deterministic trajectory prediction is computationally more involved than sampling trajectories from the system. However, for the subsequent policy update, deterministic trajectory prediction can allow for analytic computation of gradients, which can be advantageous over stochastic trajectory generation, where these gradients can only be approximated.

1.2.1 Model-free and Model-based Policy Search

Model-free policy search methods use real robot interactions to create sample trajectories $\boldsymbol{\tau}^{[i]}$. While sampling trajectories is relatively straightforward in computer simulation, when working with robots, the generation of each "sample" typically needs some level of human

supervision. Consequently, trajectory generation with the real system is considerably more time consuming than working with simulated systems. Furthermore, real robot interactions cause wear and tear in non-industrial robots. However, in spite of the relatively high number of required robot interactions for model-free policy search, learning a policy is often easier than learning accurate forward models, and, hence, model-free policy search is more widely used than model-based methods.

Model-based policy search methods attempt to address the problem of sample inefficiency by using the observed trajectories $\boldsymbol{\tau}^{[i]}$ to learn a forward model of the robot's dynamics and its environment. Subsequently, this forward model is used for *internal* simulations of the robot's dynamics and environment, based on which the policy is learned. Model-based PS methods have the potential to require fewer interactions with the robot and to efficiently generalize to unforeseen situations [6]. While the idea of using models in the context of robot learning is well known since the 1980s [2], it has been limited by its strong dependency on the quality of the learned models. In practice, the learned model is *not* exact, but only a more or less accurate approximation to the real dynamics. Since the learned policy is inherently based on internal simulations with the learned model, inaccurate models can, therefore, lead to control strategies that are not robust to model errors. In some cases, learned models may be physically implausible and contain negative masses or negative friction coefficients. These implausible effects are often exploited by the policy search algorithm, resulting in a poor quality of the learned policy. This effect can be alleviated by using models that explicitly account for model errors [21, 73]. We will discuss such methods in Section 3.

1.3 Typical Policy Representations

Typical policy representations, which are used for policy search can be categorized into time-independent representations $\pi(\boldsymbol{x})$ and time-dependent representations $\pi(\boldsymbol{x}, t)$. Time-independent representations use the same policy for all time steps, and, hence, often require a complex parametrization. Time-dependent representations can use different

policies for different time steps, allowing for a potentially simpler structure of the individual policies can be used.

We will describe all policy representations in their deterministic formulation $\pi_{\boldsymbol{\theta}}(\boldsymbol{x}, t)$. In stochastic formulations, typically a zero-mean Gaussian noise vector $\boldsymbol{\epsilon}_t$ is added to $\pi_{\boldsymbol{\theta}}(\boldsymbol{x}, t)$. In this case, the parameter vector $\boldsymbol{\theta}$ typically also includes the (co)variance matrix used for generating the noise $\boldsymbol{\epsilon}_t$. In robot learning, the three main policy representations are linear policies, radial basis function networks, and dynamic movement primitives [72].

Linear Policies. Linear controllers are the most simple time-independent representation. The policy π is a linear policy

$$\pi_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{\theta}^T \boldsymbol{\phi}(\boldsymbol{x}), \tag{1.6}$$

where $\boldsymbol{\phi}$ is a basis function vector. This policy only depends linearly on the policy parameters. However, specifying the basis functions by hand is typically a difficult task, and, hence, the application of linear controllers is limited to problems where appropriate basis functions are known, e.g., for balancing tasks, the basis functions are typically given by the state variables of the robot.

Radial Basis Functions Networks. A typical nonlinear time-independent policy representation is a radial basis function (RBF) network. An RBF policy $\pi_{\boldsymbol{\theta}}(\boldsymbol{x})$ is given as

$$\pi_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}), \quad \boldsymbol{\phi}_i(\boldsymbol{x}) = \exp\left(-\tfrac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_i)^T \boldsymbol{D}_i (\boldsymbol{x} - \boldsymbol{\mu}_i)\right), \tag{1.7}$$

where $\boldsymbol{D}_i = \mathrm{diag}(\boldsymbol{d}_i)$ is a diagonal matrix. Unlike in the linear policy case, the parameters $\boldsymbol{\beta} = \{\boldsymbol{\mu}_i, \boldsymbol{d}_i\}_{i=1,\ldots,n}$ of the basis functions themselves are now considered as free parameters that need to be learned. Hence, the parameter vector $\boldsymbol{\theta}$ of the policy is given by $\boldsymbol{\theta} = \{\boldsymbol{w}, \boldsymbol{\beta}\}$. While RBF networks are powerful policy representations, they are also difficult to learn due to the high number of nonlinear parameters. Furthermore, as RBF networks are local representations, they are hard to scale to high-dimensional state spaces.

Dynamic Movement Primitives. Dynamic Movement Primitives (DMPs) are the most widely used time-dependent policy representation in robotics [32, 72]. DMPs use nonlinear dynamical systems for generating the movement of the robot. The key principle of DMPs is to a use a linear spring–damper system which is modulated by a nonlinear forcing function f_t, i.e.,

$$\ddot{y}_t = \tau^2 \alpha_y (\beta_y (g - y_t) - \dot{y}_t) + \tau^2 f_t \,, \tag{1.8}$$

where the variable y_t directly specifies the desired joint position of the robot. The parameter τ is the time-scaling coefficient of the DMP, the coefficients α_y and β_y define the spring and damping constants of the spring–damper system and the goal parameter g is the unique point-attractor of the spring–damper system. Note that the spring–damper system is equivalent to a standard linear PD-controller that operates on a linear system with zero desired velocity, i.e.,

$$\ddot{y}_t = k_p (g - y_t) - k_d \dot{y}_t,$$

where the P-gain is given by $k_p = \tau^2 \alpha_y \beta_y$ and the D-gain by $k_d = \tau^2 \alpha_y$. The forcing function f_t changes the goal attractor g of the linear PD-controller.

One key innovation of the DMP approach is the use of a phase variable z_t to scale the execution speed of the movement. The phase variable evolves according to $\dot{z} = -\tau \alpha_z z$. It is initially set to $z = 1$ and exponentially converges to 0 as $t \to \infty$. The parameter α_z specifies the speed of the exponential decline of the phase variable. The variable τ can be used to temporally scale the evolution of the phase z_t, and, thus, the evolution of the spring–damper system as shown in Equation (1.8). For each degree of freedom, an individual spring–damper system, and, hence, and individual forcing function f_t is used. The function f_t depends on the phase variable, i.e., $f_t = f(z_t)$ and is constructed by the weighted sum of K basis functions ϕ_i

$$f(z) = \frac{\sum_{i=1}^{K} \phi_i(z) w_i}{\sum_{i=1}^{K} \phi_i(z)} z, \quad \phi_i(z) = \exp\left(-\tfrac{1}{2\sigma_i^2}(z - c_i)^2\right). \tag{1.9}$$

The parameters w_i are denoted as "shape-parameters" of the DMP as they modulate the acceleration profile, and, hence, indirectly specify the shape of the movement. From Equation (1.9), we can see that the basis functions are multiplied with the phase variable z, and, hence, f_t vanishes as $t \to \infty$. Consequently, the nonlinear dynamical system is globally stable as it behaves like a linear spring–damper system for $t \to \infty$. From this argument, we can also conclude that the goal parameter g specifies the final position of the movement while the shape parameters w_i specify how to reach this final position.

Integrating the dynamical systems for each DoF results in a desired trajectory $\boldsymbol{\tau}^* = \{\boldsymbol{y}_t\}_{t=0\ldots T}$ that is, subsequently, followed by feedback control laws [57]. The policy $\pi_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$ that is specified by a DMP, directly controls the acceleration of the joint, and, hence, is given by

$$\pi_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) = \tau^2 \alpha_y(\beta_y(g - y_t) - \dot{y}_t) + \tau^2 f(z_t).$$

Note that the DMP policy is linear in the shape parameters $\boldsymbol{w}$ and the goal attractor g, but nonlinear in the time-scaling constant τ.

The parameters $\boldsymbol{\theta}$ used for learning a DMP are typically given by the weight parameters w_i, but might also contain the goal parameters g as well as the temporal scaling parameter τ. In addition, the DMP approach has been extended in [37] such that the desired final velocity $\dot{g}$ of the joints can also be modulated. Such modulation is, for example, useful for learning hitting movements in robot table tennis. Typically, $K = 5$ to 20 basis functions are used, i.e., 5 to 20 shape weights per degree of freedom of the robot are used.

Miscellaneous Representations. Other representations that have been used in the literature include central pattern generators for robot walking [25] and feed-forward neural networks, which have been used mainly in simulation [31, 90].

1.4 Outline

The structure of this monograph is as follows: In Section 2, we give a detailed overview of model-free policy search methods, where we classify policy search algorithms according to their policy evaluation, policy update, and exploration strategy. For the policy update

strategies, we will follow the taxonomy in Figure 1.1 and discuss policy gradient methods, EM-based approaches, information-theoretic approaches. Additionally, we will discuss miscellaneous important methods such as stochastic optimization and policy search approaches based on the path integral theory. Policy search algorithms can either use a step-based or episode-based policy evaluation strategy. Most policy update strategies presented in Figure 1.1 can be used for both, step-based and episode-based policy evaluation. We will present both types of algorithms if they have been introduced in the literature. Subsequently, we will discuss different exploration strategies for model-free policy search and conclude this section with robot applications of model-free policy search. Section 3 surveys model-based policy search methods in robotics. Here, we introduce two models that are commonly used in policy search: locally weighted regression and Gaussian processes. Furthermore, we detail stochastic and deterministic inference algorithms to compute a probability distribution $p_\pi(\boldsymbol{\tau})$ over trajectories (see the red boxes in Figure 1.1). We conclude this section with examples of model-based policy search methods and their application to robotic systems. In Section 4, we give recommendations for the practitioner and conclude this monograph.

References

[1] P. Abbeel, M. Quigley, and A. Y. Ng, "Using inaccurate models in reinforcement learning," in *Proceedings of the International Conference on Machine Learning*, pp. 1–8, Pittsburgh, PA, USA, June 2006.

[2] E. W. Aboaf, S. M. Drucker, and C. G. Atkeson, "Task-level robot learning: Juggling a tennis ball more accurately," in *Proceedings of the International Conference on Robotics and Automation*, 1989.

[3] S. Amari, "Natural gradient works efficiently in learning," *Neural Computation*, vol. 10, pp. 251–276, February 1998.

[4] B. D. O. Anderson and J. B. Moore, *Optimal Filtering*. Mineola, NY, USA: Dover Publications, 2005.

[5] K. J. Aström and B. Wittenmark, *Adaptive Control*. Dover Publications, 2008.

[6] C. G. Atkeson and J. C. Santamaría, "A comparison of direct and model-based reinforcement learning," in *Proceedings of the International Conference on Robotics and Automation*, 1997.

[7] J. A. Bagnell and J. G. Schneider, "Autonomous helicopter control using reinforcement learning policy search methods," in *Proceedings of the International Conference on Robotics and Automation*, pp. 1615–1620, 2001.

[8] J. A. Bagnell and J. G. Schneider, "Covariant policy search," in *Proceedings of the International Joint Conference on Artifical Intelligence*, August 2003.

[9] J. Baxter and P. Bartlett, "Direct gradient-based reinforcement learning: I. gradient estimation algorithms," Technical report, 1999.

[10] J. Baxter and P. L. Bartlett, "Infinite-horizon policy-gradient estimation," *Journal of Artificial Intelligence Research*, 2001.

[11] D. P. Bertsekas, *Dynamic Programming and Optimal Control,* volume 1 of *Optimization and Computation Series*. Athena Scientific, 3rd ed., 2005.
[12] C. M. Bishop, *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer-Verlag, 2006.
[13] J. A. Boyan, "Least-squares temporal difference learning," in *Proceedings of the Sixteenth International Conference on Machine Learning*, pp. 49–56, 1999.
[14] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.
[15] W. S. Cleveland and S. J. Devlin, "Locally-weighted regression: An approach to regression analysis by local fitting," *Journal of the American Statistical Association*, vol. 83, no. 403, pp. 596–610, 1988.
[16] R. Coulom, "Reinforcement learning using neural networks, with applications to motor control," PhD thesis, Institut National Polytechnique de Grenoble, 2002.
[17] C. Daniel, G. Neumann, and J. Peters, "Hierarchical relative entropy policy search," in *Proceedings of the International Conference of Artificial Intelligence and Statistics*, (N. Lawrence and M. Girolami, eds.), pp. 273–281, 2012.
[18] C. Daniel, G. Neumann, and J. Peters, "Learning concurrent motor skills in versatile solution spaces," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012.
[19] P. Dayan and G. E. Hinton, "Using expectation-maximization for reinforcement learning," *Neural Computation*, vol. 9, no. 2, pp. 271–278, 1997.
[20] M. P. Deisenroth, *Efficient Reinforcement Learning using Gaussian Processes,* volume 9. KIT Scientific Publishing, November 2010. ISBN 978-3-86644-569-7.
[21] M. P. Deisenroth and C. E. Rasmussen, "PILCO: A model-based and data-efficient approach to policy search," in *Proceedings of the International Conference on Machine Learning*, pp. 465–472, New York, NY, USA, June 2011.
[22] M. P. Deisenroth, C. E. Rasmussen, and D. Fox, "Learning to control a low-cost manipulator using data-efficient reinforcement learning," in *Proceedings of the International Conference on Robotics: Science and Systems*, Los Angeles, CA, USA, June 2011.
[23] M. P. Deisenroth, C. E. Rasmussen, and J. Peters, "Gaussian process dynamic programming," *Neurocomputing*, vol. 72, no. 7–9, pp. 1508–1524, March 2009.
[24] K. Doya, "Reinforcement learning in continuous time and space," *Neural Computation*, vol. 12, no. 1, pp. 219–245, January 2000.
[25] G. Endo, J. Morimoto, T. Matsubara, J. Nakanishi, and G. Cheng, "Learning CPG-based biped locomotion with a policy gradient method: Application to a humanoid robot," *International Journal of Robotics Research*, 2008.
[26] S. Fabri and V. Kadirkamanathan, "Dual adaptive control of nonlinear stochastic systems using neural networks," *Automatica*, vol. 34, no. 2, pp. 245–253, 1998.
[27] A. A. Fel'dbaum, "Dual control theory, Parts I and II," *Automation and Remote Control*, vol. 21, no. 11, pp. 874–880, 1961.
[28] E. B. Fox and D. B. Dunson, "Multiresolution Gaussian processes," in *Advances in Neural Information Processing Systems*, The MIT Press, 2012.

[29] N. Hansen, S. Muller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES)," *Evolutionary Computation*, vol. 11, no. 1, pp. 1–18, 2003.

[30] V. Heidrich-Meisner and C. Igel, "Hoeffding and Bernstein races for selecting policies in evolutionary direct policy search," in *Proceedings of the Annual International Conference on Machine Learning*, pp. 401–408, 2009.

[31] V. Heidrich-Meisner and C. Igel, "Neuroevolution strategies for episodic reinforcement learning," *Journal of Algorithms*, vol. 64, no. 4, pp. 152–168, October 2009.

[32] A. J. Ijspeert and S. Schaal, "Learning attractor landscapes for learning motor primitives," in *Advances in Neural Information Processing Systems*, pp. 1523–1530, Cambridge, MA: MIT Press, 2003.

[33] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, March 2004.

[34] H. Kimura and S. Kobayashi, "Efficient non-linear control by combining Q-learning with local linear controllers," in *Proceedings of the International Conference on Machine Learning*, pp. 210–219, 1999.

[35] J. Ko, D. J. Klein, D. Fox, and D. Haehnel, "Gaussian processes and reinforcement learning for identification and control of an autonomous blimp," in *Proceedings of the International Conference on Robotics and Automation*, pp. 742–747, 2007.

[36] J. Kober, B. J. Mohler, and J. Peters, "Learning perceptual coupling for motor primitives," in *Intelligent Robots and Systems*, pp. 834–839, 2008.

[37] J. Kober, K. Mülling, O. Kroemer, C. H. Lampert, B. Schölkopf, and J. Peters, "Movement templates for learning of hitting and batting," in *International Conference on Robotics and Automation*, pp. 853–858, 2010.

[38] J. Kober, E. Oztop, and J. Peters, "Reinforcement learning to adjust robot movements to new situations," in *Proceedings of the 2010 Robotics: Science and Systems Conference*, 2010.

[39] J. Kober and J. Peters, "Policy search for motor primitives in robotics," *Machine Learning*, pp. 1–33, 2010.

[40] N. Kohl and P. Stone, "Policy gradient reinforcement learning for fast quadrupedal locomotion," in *Proceedings of the International Conference on Robotics and Automation*, 2003.

[41] P. Kormushev, S. Calinon, and D. G. Caldwell, "Robot motor skill coordination with EM-based reinforcement learning," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010.

[42] A. Kupcsik, M. P. Deisenroth, J. Peters, and G. Neumann, "Data-efficient generalization of robot skills with contextual policy search," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2013.

[43] M. G. Lagoudakis and R. Parr, "Least-squares policy iteration," *Journal of Machine Learning Research*, vol. 4, pp. 1107–1149, December 2003.

[44] D. J. C. MacKay, *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

[45] D. C. McFarlane and K. Glover, *Lecture Notes in Control and Information Sciences,* volume 138, chapter Robust Controller Design using Normalised Coprime Factor Plant Descriptions. Springer-Verlag, 1989.

[46] J. Morimoto and C. G. Atkeson, "Minimax differential dynamic programming: An application to robust biped walking," in *Advances in Neural Information Processing Systems*, (S. Becker, S. Thrun, and K. Obermayer, eds.), The MIT Press, 2003.

[47] R. Neal and G. E. Hinton, "A view of the EM algorithm that justifies incremental, sparse, and other variants," in *Learning in Graphical Models*, pp. 355–368, Kluwer Academic Publishers, 1998.

[48] J. A. Nelder and R. Mead, "A simplex method for function minimization," *Computer Journal*, vol. 7, pp. 308–313, 1965.

[49] G. Neumann, "Variational inference for policy search in changing situations," in *Proceedings of the International Conference on Machine Learning*, pp. 817–824, New York, NY, USA, June 2011.

[50] G. Neumann and J. Peters, "Fitted Q-iteration by advantage weighted regression," in *Neural Information Processing Systems*, MA: MIT Press, 2009.

[51] A. Y. Ng, "Stanford engineering everywhere CS229 — machine learning," Lecture 20, http://see.stanford.edu/materials/aimlcs229/transcripts/MachineLearning-Lecture20.html, 2008.

[52] A. Y. Ng, A. Coates, M. Diel, V. Ganapathi, J. Schulte, B. Tse, E. Berger, and E. Liang, "Autonomous inverted helicopter flight via reinforcement learning," in *International Symposium on Experimental Robotics,* volume 21 of *Springer Tracts in Advanced Robotics*, (M. H. Ang Jr. and O. Khatib, eds.), pp. 363–372, Springer, 2004.

[53] A. Y. Ng and M. Jordan, "Pegasus: A policy search method for large MDPs and POMDPs," in *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pp. 406–415, 2000.

[54] A. Y. Ng, H. J. Kim, M. I. Jordan, and S. Sastry, "Autonomous helicopter flight via reinforcement learning," in *Advances in Neural Information Processing Systems*, (S. Thrun, L. K. Saul, and B. Schölkopf, eds.), Cambridge, MA, USA: The MIT Press, 2004.

[55] D. Nguyen-Tuong, M. Seeger, and J. Peters, "Model learning with local Gaussian process regression," *Advanced Robotics*, vol. 23, no. 15, pp. 2015–2034, 2009.

[56] B. Øksendal, *Stochastic Differential Equations: An Introduction with Applications (Universitext)*. Springer, 6th ed., September 2010.

[57] J. Peters, M. Mistry, F. E. Udwadia, J. Nakanishi, and S. Schaal, "A unifying methodology for robot control with redundant DOFs," *Autonomous Robots*, vol. 1, pp. 1–12, 2008.

[58] J. Peters, K. Mülling, and Y. Altun, "Relative entropy policy search," in *Proceedings of the National Conference on Artificial Intelligence*, 2010.

[59] J. Peters and S. Schaal, "Policy gradient methods for robotics," in *Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robotics Systems*, pp. 2219–2225, Beijing, China, 2006.

[60] J. Peters and S. Schaal, "Applying the episodic natural actor-critic architecture to motor primitive learning," in *Proceedings of the European Symposium on Artificial Neural Networks*, 2007.

[61] J. Peters and S. Schaal, "Natural actor-critic," *Neurocomputation*, vol. 71, no. 7–9, pp. 1180–1190, 2008.

[62] J. Peters and S. Schaal, "Reinforcement learning of motor skills with policy gradients," *Neural Networks*, vol. 4, pp. 682–697, 2008.
[63] J. Peters, S. Vijayakumar, and S. Schaal, "Reinforcement learning for humanoid robotics," in *IEEE-RAS International Conference on Humanoid Robots*, September 2003.
[64] J. Quiñonero-Candela and C. E. Rasmussen, "A unifying view of sparse approximate gaussian process regression," *Journal of Machine Learning Research*, vol. 6, no. 2, pp. 1939–1960, 2005.
[65] T. Raiko and M. Tornio, "Variational Bayesian learning of nonlinear hidden state-space models for model predictive control," *Neurocomputing*, vol. 72, no. 16–18, pp. 3702–3712, 2009.
[66] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. The MIT Press, 2006.
[67] L. Rozo, S. Calinon, D. G. Caldwell, P. Jimenez, and C. Torras, "Learning collaborative impedance-based robot behaviors," in *AAAI Conference on Artificial Intelligence*, Bellevue, Washington, USA, 2013.
[68] T. Rückstieß, M. Felder, and J. Schmidhuber, "State-dependent exploration for policy gradient methods," in *European Conference on Machine Learning*, pp. 234–249, 2008.
[69] T. Rückstieß, F. Sehnke, T. Schaul, D. Wierstra, Y. Sun, and J. Schmidhuber, "Exploring parameter space in reinforcement learning," *Paladyn*, vol. 1, no. 1, pp. 14–24, March 2010.
[70] S. J. Russel and P. Norvig, *Artificial Intelligence: A Modern Approach*. Pearson Education, 2003.
[71] S. Schaal and C. G. Atkeson, "Constructive incremental learning from only local information," *Neural Computation*, vol. 10, no. 8, pp. 2047–2084, 1998.
[72] S. Schaal, J. Peters, J. Nakanishi, and A. Ijspeert, "Learning movement primitives," in *International Symposium on Robotics Research*, pp. 561–572, 2003.
[73] J. G. Schneider, "Exploiting model uncertainty estimates for safe dynamic control learning," in *Advances in Neural Information Processing Systems*, Morgan Kaufman Publishers, 1997.
[74] F. Sehnke, C. Osendorfer, T. Rückstieß, A. Graves, J. Peters, and J. Schmidhuber, "Policy gradients with parameter-based exploration for control," in *Proceedings of the International Conference on Artificial Neural Networks*, 2008.
[75] F. Sehnke, C. Osendorfer, T. Rückstieß, A. Graves, J. Peters, and J. Schmidhuber, "Parameter-exploring policy gradients," *Neural Networks*, vol. 23, no. 4, pp. 551–559, 2010.
[76] C. Shu, H. Ding, and N. Zhao, "Numerical comparison of least square-based finite-difference (LSFD) and radial basis function-based finite-difference (RBFFD) methods," *Computers & Mathematics with Applications*, vol. 51, no. 8, pp. 1297–1310, April 2006.
[77] E. Snelson and Z. Ghahramani, "Sparse Gaussian processes using pseudo-inputs," in *Advances in Neural Information Processing Systems 18*, (Y. Weiss, B. Schölkopf, and J. C. Platt, eds.), pp. 1257–1264, Cambridge, MA, USA: The MIT Press, 2006.

[78] F. Stulp and O. Sigaud, "Path integral policy improvement with covariance matrix adaptation," in *Proceedings of the International Conference on Machine Learning*, 2012.

[79] Y. Sun, D. Wierstra, T. Schaul, and J. Schmidhuber, "Efficient natural evolution strategies," in *Proceedings of the Annual conference on Genetic and Evolutionary Computation*, pp. 539–546, New York, NY, USA, 2009.

[80] R. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Neural Information Processing Systems*, 1999.

[81] R. S. Sutton and A. G. Barto, *Reinforcement Learning*. The MIT Press, 1998.

[82] E. Theodorou, J. Buchli, and S. Schaal, "A generalized path integral control approach to reinforcement learning," *Journal of Machine Learning Research*, vol. 11, pp. 3137–3181, 2010.

[83] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. The MIT Press, 2005.

[84] E. Todorov, "Optimal control theory," *Bayesian Brain*, 2006.

[85] M. Toussaint, "Robot trajectory optimization using approximate inference," in *Proceedings of the International Conference on Machine Learning*, 2009.

[86] N. Vlassis and M. Toussaint, "Model-free reinforcement learning as mixture learning," in *Proceedings of the International Conference on Machine Learning*, 2009.

[87] N. Vlassis, M. Toussaint, G. Kontes, and S. Piperidis, "Learning model-free robot control by a Monte Carlo EM algorithm," *Autonomous Robots*, vol. 27, no. 2, pp. 123–130, 2009.

[88] P. Wawrzynski and A. Pacut, "Model-free off-policy reinforcement learning in continuous environment," in *Proceedings of the INNS-IEEE International Joint Conference on Neural Networks*, pp. 1091–1096, 2004.

[89] D. Wierstra, T. Schaul, J. Peters, and J. Schmidhuber, "Natural evolution strategies," in *IEEE Congress on Evolutionary Computation*, pp. 3381–3387, 2008.

[90] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine Learning*, vol. 8, pp. 229–256, 1992.

[91] B. Wittenmark, "Adaptive dual control methods: An overview," in *Proceedings of the 5th IFAC Symposium on Adaptive Systems in Control and Signal Processing*, pp. 67–72, 1995.

[92] K. Xiong, H.-Y. Zhang, and C. W. Chan, "Performance evaluation of UKF-based nonlinear filtering," *Automatica*, vol. 42, pp. 261–270, 2006.