Model Learning for Robot Control

Summary of the PhD Thesis

Author:
BOTOND ATTILA BÓCSI

Supervisor:
PROF. DR. HORIA F. POP

DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE,
BABES-BOLYAI UNIVERSITY, CLUJ-NAPOCA, 2012

Table of Contents of the Summary

Table of Contents of the Thesis

List of Publications

- B. Bócsi, L. Csató, B. Schölkopf, and J. Peters. Indirect robot model learning for tracking control. *Robotics and Autonomous Systems (submitted on 10 September 2012)*, 2012a. Submitted.
- B. Bócsi, P. Hennig, L. Csató, and J. Peters. Learning tracking control with forward models. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 259–264, St. Paul, MN, USA, 2012b
- B. Bócsi, D. Nguyen-Tuong, L. Csató, B. Schoelkopf, and J. Peters. Learning inverse kinematics with structured prediction. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 698–703, San Francisco, USA, 2011b
- H. Jakab, B. A. Bócsi, and L. Csató. Non-parametric value function approximation in robotics. In H. F. Pop, editor, *MACS2010: The 8th Joint Conference on Mathematics and Computer Science*, volume Selected Papers, pages 235–248, Komarno, Slovakia, 2011. Györ:NOVADAT
- B. Bócsi and L. Csató. Reinforcement learning algorithms in robotics. In M. Frentiu, H. F. Pop, and S. Motogna, editors, *KEPT-2011:Knowledge Engineering Principles and Techniques International Conference, Selected Papers.*, pages 131–143. Presa Universitara Clujeana, 2011a
- B. Bócsi, L. Csató, and Jan Peters. Structured output Gaussian processes. Technical report, Babes-Bolyai University, 2011a. URL `http://www.cs.ubbcluj.ro/~bboti/pubs/sogp_2011.pdf`
- B. A. Bócsi and L. Csató. Reinforcement learning algorithms in robotics. *Studia Universitatis Babes-Bolyai Series Informatica*, LVI(2):61–67, 2011b
- B. A. Bócsi, H. Jakab, and L. Csató. Nonparametric methods in robotics. In *Proceedings of the 8th Joint Conference on Mathematics and Computer Science (Abstract)*, page 8, Komarno, Slovakia, 2010

Research grants related to the thesis

- Grant **PN-II-RU-TE-2011-3-0278** of the Romanian Ministry of Education: Nonparametric methods in machine learning: application to robotic and data analysis.

1 Introduction

In the last few decades, robotics related research became more and more popular. The reason is the extensive use of robots on a large variety of fields. As an example, the high level automation in industry is mainly achieved by means of robots. Using robots has several benefits over human work force, *e.g.*, it can be faster, cheaper, and more accurate. Despite the appealing features, automated robotics has significant drawbacks as well. In general, industrial robots perform well only in predefined environments where nothing *unexpected* can happen. Humans usually do not have this kind of limitations. As a solution we need to enhance automated robotics such that they work well in uncontrolled environments as well.

From the limitations above, the application of robots in stochastic environments became a novel requirement. For example, when the terrain of the execution is unknown, the working environment of the robot should be considered stochastic. Stochastic modeling of the environments is also necessary when the interaction with humans is considered since the deterministic modeling of human behavior is too complex. The goal is to let the robots solve a broader class of problems without preprogramming all possible scenarios that can encounter during the task. We can state in general that in a constantly changing environment robots must possess an adaptive *intelligent like* behavior.

Controlling robots in an adaptive way that result in adaptive behavior led to a relatively new approach to robotics [Thrun et al., 2005]. Approaches and algorithms have been borrowed from control theory [Liberzon, 2012; Sontag, 1998], artificial intelligence [Russell and Norvig, 2003], and machine learning [Bishop, 2006]. This new field faced a plethora of new challenges. The methods had to be formed such as they meet the new requirements. For example, high degree of stability is an essential requirement to avoided the damage of the robot in certain situations, *e.g.*, the physical limits of the robot must be avoided [Nakanishi et al., 2008]. Another expected property is fast evaluation of the method [Nakanishi et al., 2008]. Speed is important since high technology robots may require a control command at high rates, *e.g.*, a robot that operates at 500Hz requires a command at every two milliseconds.

Within adaptive robot control, several problems are formulated such as self localization, planning, motion control [Thrun et al., 2005]. Another interesting branch is modeling of higher level behavior called neurorobotics [Steels and Hild, 2012; von Twickel et al., 2012]. By higher level behavior we mean human like cognitive capabilities. The field has grown in such a huge rate that an exhaustive overview is beyond the limits of this work.

This thesis focuses on lower level robot control. More precisely, we aim to define algorithms that provide low level motor commands to the robot without achieving very high level goals (*e.g.*, a low level goal can be moving a robot arm from one position to another, while a high level goal can be grabbing a cup [Detry et al., 2011]). By low level control we mean defining directly the torque values provided to the motors (high level control does not deal with the exact torque values, rather they defined higher level desired effects such as grabbing a cup [Detry et al., 2011] as mentioned before). As a result the experiments conducted in this field seem to solve relatively simple problems. One may think that bal-

ancing a pole on a moving car or tracking a predefined figure with a robot arm are simple problems. However, providing efficient *adaptive* solutions is still challenging. For example, imagine that we are tracking a predefined figure with a robot arm (see Section 4 for more details). Then, the parameters of the robot suddenly change (*e.g.*, we put an additional weight on the arm), thus, the provided working parameters are not adequate any more to achieve the desired goal. The algorithm has to (explicitly or implicitly) recognize that some physical parameters changed and consequently adapt its behavior to the new values.

In this thesis we focus on how the robot model can be approximated from sampled data. Since the control algorithm is based on the robot model, the control method is also approximated. We say that both the robot model and the control algorithm are *learned* from sampled data. Learning robot models from sampled data has the benefit over analytic solutions that it implicitly adjusts itself to the changing sensory informations. The robot model is based on empirical observations, thus, if the underlying data generation process changes, the observations change, and changes the robot model too.

To learn the robot model from sampled data, we apply machine learning methods. When applying a machine learning method in this context, one has to consider the limitations and the drawback of the given method in the view of its application. For example, a regression method may give very accurate predictions but the computation time may be too high for robot applications. Another important feature is the convergence time: the time to collect the necessary number of observations until convergence cannot exceed a reasonable time limit.

The summary is organized as follows: in Section 2, we introduce Gaussian processes since we use them extensively for many purposes (*e.g.*, value-function approximation, forward kinematics modeling). Section 3 addresses the theoretical point of view of robot learning. It discusses reinforcement learning as the most general framework of machine learning. Section 4 presents a more practical approach to robot learning. The inverse kinematics model of a robot arm is learned from sampled data. Section 5 extends the general framework of robot model learning. Methods from transfer learning are used to improve inverse kinematics learning. Section 6 concludes this summary and shows interesting future directions where additional improvements can be gained in robot model learning.

Contributions of the Thesis

This thesis has the following contributions:

1. We made a comparative study of RL algorithms in the context of robot control. We investigated how different RL approaches work on high dimensional continuous domains such as robot control. Experimental results show that the direct modeling of the control algorithms lead to more accurate control opposed to traditional approaches. [Bócsi and Csató, 2011b] [Bócsi and Csató, 2011a].

2. We used Gaussian processes to approximate the state-action value function in RL setup with Q-learning. The online update and the efficient data handling of Gaussian pro-

cesses has been achieved by using sparse online Gaussian process approximations. Results show that the improved model converged to a more accurate control policy. [Jakab et al., 2011] [Bócsi et al., 2010].

3. An indirect way of modeling inverse kinematics of robot manipulators have been proposed. We used a joint energy function of inputs and outputs and obtained prediction by using local minimization of the energy function. Three different ways to model the energy function have been proposed using support vector machines , Gaussian processes, and Gaussian processes for the forward kinematics model. Experiments show that accurate inverse kinematics models can be obtained with the proposed approach that outperforms the state-of-the-art algorithms. The method works for non-rigid robot as well where other algorithms fail [Bócsi et al., 2011b], [Bócsi et al., 2012b], [Bócsi et al., 2011a], [Bócsi et al., 2012a]. Furthermore, the method called structured output Gaussian process can be used to solve general structured output learning problems [Bócsi et al., 2011a].

4. Robot model learning has been improved using transfer learning algorithms. The main idea is to use knowledge gained from other experiments of the robot or even from experiments of other robots. The additional data is transformed such that useful information can be gained from it. Experiments show that more accurate forward models can be obtained when additional data-sets are also considered. Results will be published in the future.

2 Non-parametric Learning with Gaussian Processes

In the last few decades non-parametric methods (*e.g.*, support vector machines (SVMs) [Boser et al., 1992; Vapnik, 1999; Schölkopf and Smola, 2002], kernel principal component analysis [Schölkopf and Smola, 2002], kernel density estimation [Parzen, 1962], Dirichlet processes [Ferguson, 1973], Gaussian processes [Rasmussen and Williams, 2005]) gained significant attention in machine learning beside traditional parametric methods, such as principal component analysis [Lee and Verleysen, 2007] or neural networks [Barber and Bishop, 1998]. Experiments show that in many cases these algorithms lead to better results.

In the traditional parametric approach, the model of the data is defined with a fixed number of parameters (*e.g.*, weights of a neural network) and these parameters are adjusted to the data by minimizing a loss function (*e.g.*, squared loss, hinge loss, negative log likelihood). Since the number of the parameters is fixed, the complexity of the model is not too flexible. In the non-parametric approach the parameters of the data model is not fixed a-priori, but it depends on the data. The adaptive parameter number results in more flexible models in the sense that the complexity of the model depends on the data.

Gaussian processes (GPs) are non-parametric Bayesian models which define a distribution over functions characterized by the mean function $\mu(\cdot)$ and covariance (or kernel) function $k(\cdot,\cdot)$, see [Rasmussen and Williams, 2005]. Given a training set $\mathbf{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^m$, it

aims to find the mapping $\mathbf{x} \rightarrow \mathbf{y}$ that explains the data-set best. One of the main advantages of GPs is that the solution will not only be a function, but a Gaussian-distributed probability distribution at every point of the function. The posterior distribution of a test point $\mathbf{x}_*$ is Gaussian-distributed with mean μ_* and variance σ_*^2:

$$\begin{aligned} \mu_* &= \mathbf{k}_*^\top (\mathbf{K} + \sigma_0^2 \mathbf{I}_m)^{-1} \mathbf{y} \\ \sigma_*^2 &= k_{**} - \mathbf{k}_*^\top (\mathbf{K} + \sigma_0^2 \mathbf{I}_m)^{-1} \mathbf{k}_*, \end{aligned}$$

where $\mathbf{K} \in \Re^{m \times m}$ with $\mathbf{K}^{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, $\mathbf{k}_* \in \Re^{m \times 1}$ with $\mathbf{k}_*^i = k(\mathbf{x}_i, \mathbf{x}_*)$, $k_{**} = k(\mathbf{x}_*, \mathbf{x}_*)$, $\mathbf{I}_m$ is the identity matrix of size m, and σ_0^2 is the variance of the measurement noise. By having not only point-wise estimates but a posterior probability distribution as prediction, we have at disposal the posterior variance σ_*^2 that quantifies how certain the prediction is.

Kernel functions – $k(\cdot, \cdot)$ from the notations above – play a crucial role in several machine learning methods. Usually, kernel based method use *simple* algorithms to make predictions. The additional improvement is gained by transforming the data into different spaces – using a function $\phi(\cdot)$ – and calculating the pairwise scalar product in that space. The space the data is transformed into is called the *feature space*. A possible way to define the kernel is to select a transformation function $\phi(\cdot)$ and use the scalar product. However, the real advantage of using kernels comes from the property that kernels can be expressed without explicitly choosing a $\phi(\cdot)$ function. Note that the feature space may be infinite dimensional, thus, the explicit representation of $\phi(\cdot)$ may be impossible.

It has been shown that for every positive definite function $k(\cdot, \cdot)$, a construction of a feature function is possible that leads to the given kernel [Shawe-Taylor and Cristianini, 2004]. A frequently used kernel where the explicit definition of the kernel via $\phi(\cdot)$ would require an infinite dimensional feature function is the squared exponential kernel:

$$k(\mathbf{x}_1, \mathbf{x}_2) = C \exp\left\{ -\frac{\|\mathbf{x}_1 - \mathbf{x}_2\|^2}{2\omega} \right\},$$

where C is the amplitude and ω is the characteristic length-scale. Note that $\boldsymbol{\omega}$ can be vector valued. Thus, we can define different lengths scale in each input dimension, *i.e.*, $k(\mathbf{x}_1, \mathbf{x}_2) = C \exp\left\{ \sum_i (\mathbf{x}_1 - \mathbf{x}_2)_i^2 / 2\boldsymbol{\omega}_i \right\}$, called an anisotropic squared exponential kernel. We mention this kernel since it is extensively used in our experiments.

The main disadvantage of GPs is that the time and space complexity of the inference grows cubically with the number of the data points. To overcome this problem, several sparsification methods have been proposed [Csató and Opper, 2002; Quiñonero Candela and Rasmussen, 2005; Lawrence et al., 2002; Snelson and Ghahramani, 2006; Titsias, 2009; Lázaro-Gredilla et al., 2010; Ranganathan et al., 2011; Snelson, 2006; Smola and Bartlett, 2001]. We adopt a method proposed by Csató and Opper [2002], which can be applied online. The method is the online approximation to the posterior distribution using a sequential algorithm [Opper, 1998] where we combine the likelihood of a single data with the GP prior from the result of the previous approximation step. Given a dataset $\mathbf{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^m$, a $\mathcal{GP}_m$ model is defined based on m data points. When a new point arrives, $\mathcal{GP}_{m+1}$ is obtained by

using the Bayes theorem where $\mathcal{GP}_m$ plays the role of the prior model and $(\mathbf{x}_{m+1}, \mathbf{y}_{m+1})$ is the observation, *i.e.*,

$$\mathcal{GP}_{m+1}(f) \quad \propto \quad p((\mathbf{x}_{m+1}, \mathbf{y}_{m+1})|\mathcal{GP}_m(f))\ \mathcal{GP}_m(f).$$

where f is a sample drawn from the model $\mathcal{GP}$. Assuming additive Gaussian noise, this assumption leads to an analytically tractable model [Rasmussen and Williams, 2005].

The sparsification algorithm defines an approximation to the exact Gaussian process posterior, chosen such that the discrepancy between the exact and the approximate posterior is minimized. An important feature of the method is that the parameters of the GP are updated even when the new data point is not included into the base set. Thus, the accuracy of the approximation improves during the learning process while the base points might not change.

3 Reinforcement Learning

In the machine learning community reinforcement learning (RL) is the most appropriate to the requirements of robot learning [Sutton and Barto, 1998]. The reason is that the learning process is embedded into a more general environment than supervised or unsupervised learning. In supervised learning we aim to find a mapping from an input space to an output space, while in the unsupervised setup we are looking for some structure in the data [Bishop, 2006]. In contrast, in RL, simple finding a mapping may not be sufficient since the prediction will affect our future decisions as well. The algorithm has to take into account the time perspective as well.

The specific of RL is that there is an *exploration* phase. At the beginning of a RL learning process, we have to make random *actions* in order to *explore* the *world*. Then, once we have *sufficient knowledge* about the world, we act such that get the most *positive feedback*. Note that the very intuitive words *action*, *exploration*, *world*, *sufficient knowledge*, and *positive feedback* have been used. To have mathematically formulated algorithms, these concepts have to be defined rigorously. The framework that provides the mathematical treatment of the concepts are Markov decision processes [Puterman, 1994].

Among the machine learning approaches, RL is the most similar to human learning. Infants improve their skills in a very similar way as RL works. They make actions in the world and observe the results of these actions. In the beginning, the actions are almost random, but in time they are improved based on the feedback taken from the world [Skinner, 2011]. For humans, the learning works on much larger scales than in RL (*e.g.*, the human body has more degrees of freedom that can use efficiently than current RL can learn to control for robots). Many times, in machine learning we do not have the computational power, nor the sufficient prior knowledge to solve such complex problems with such a good accuracy as humans do. For example, theoretically, the state-of-the-art RL methods can learn how to control a humanoid robot, however, the algorithm would not converge in the lifetime of the robot.

Examples where RL has been used with success can be found on various fields. Strategies for board games, such as backgammon, are learned as they compete best human players and beat them [Tesauro, 1995]. However, not all board games can be learned with RL. For example, chess or go are too complex and the current RL methods are far from human players. Another interesting application is advising adverts for users based on the adverts selected before [Gittins et al., 2011].

Recently, RL has gained significant attention in robotics [Peters et al., 2003; Peters and Schaal, 2008b; Bócsi and Csató, 2011a]. Challenging problems in robot control such as pole balancing [Deisenroth and Rasmussen, 2011], ball beam [Benbrahim et al., 1992], and mountain car robot control [Rasmussen and Kuss, 2004] have been solved with success and high accuracy.

The main advantage of RL based robot control over analytical ones is that the exact parameters of robots (*e.g.*, weights and widths of the robot elements) do not have be known in advance. The algorithm adapts its behavior to the actual values. Furthermore, it may happen that these parameters change over time. The adaptation for RL algorithms is implicit.

Next, we present the theoretical background of RL formulated as optimal control in Markov decision processes and analyze the state-of-the-art RL algorithms from the perspective of robot control.

Markov Decision Processes

Formally RL problems are defined in terms of a *Markov Decision Process* (MDP) [Puterman, 1994] consisting of a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \pi)$ where (1) $\mathcal{S}$ is the state space; (2) $\mathcal{A}$ is the action space; (3) $\mathcal{P}^{a}_{ss'} : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to \mathfrak{R}$, with $\mathcal{P}^{a}_{ss'} = P(s'|s, a)$ are the transition probabilities, *i.e.*, the probability of going from state s to s' by taking action a; (4) $\mathcal{R}^{a}_{ss'} : \mathcal{S} \times \mathcal{S} \times \mathcal{A} \to \mathfrak{R}$ is the reward received when action a was taken in state s followed by state s'; (5) $\pi(s, a) : \mathcal{S} \times \mathcal{A} \to [0, 1]$, $\pi(s, a) = P(a|s)$ is called the policy, that is the probability of taking action a in state s. A trajectory – a.k.a episode or roll-out – τ is a sequence of triplets $(s_t, a_t, r_t) \in \mathcal{S} \times \mathcal{A} \times \mathcal{R}$ with t the time index. The values of the triplets a_{t+1}, s_{t+1}, r_{t+1} are obtained from sampling based on the policy $\pi(s, a)$ and the transition probabilities. By solving an MDP, we understand the finding of a policy π' that maximizes the long-term (discounted) reward along a trajectory generated by the respective policy

$$\pi' = \arg\max_{\pi} \; E_{\pi}\left[\sum_{t} \gamma^{t} r_{t}\right],$$

where $r_t \in \tau$, E_π denotes expectation conditioned on π, and $\gamma \in (0, 1]$ is a discount factor. The objective of RL is to solve the MDP underlying the problem. Solving the MDP is not straightforward. To tackle the problem, different algorithms has been introduced, approaching the problem from different points of views.

Reinforcement Learning Algorithms

RL methods tackle the solution of a MDP from different points of view. Next, we present the prevailing approaches.

1. **Value function based methods** [Sutton and Barto, 1998] model the optimal policy indirectly via so called value functions. The key insight is that we measure the utility of states and actions respectively, and based on these values, the optimal policy chooses the action that has to higher utility. Once a function $Q(s, a)$, that expresses the utility of action a in state s is known, the optimal policy can be easily computed by taking the most valuable action in every state, *i.e.*, $\pi(s, a) \sim \text{argmax}_a\, Q(s, a)$. There are different rules for updating the Q function, *e.g.*, Q-learning [Sutton and Barto, 1998].

2. **Policy gradient methods** [Peters and Schaal, 2008b] model the policy directly as a parametric function π_η, *e.g.*, by a neural network, and update its parameters using steepest gradient descent [Snyman, 2005], based on the gradient of the average expected reward: $J(\eta) = E_{\pi_\eta}\left[\sum_t \gamma^t r_t\right]$. The computation of the gradient of $J(\eta)$ is not tractable, thus, to ease the difficulties related to calculating the expected return, several approximations of the gradient have been suggested, *e.g.*, finite difference methods, vanilla policy gradient, natural policy gradient [Peters and Schaal, 2008b].

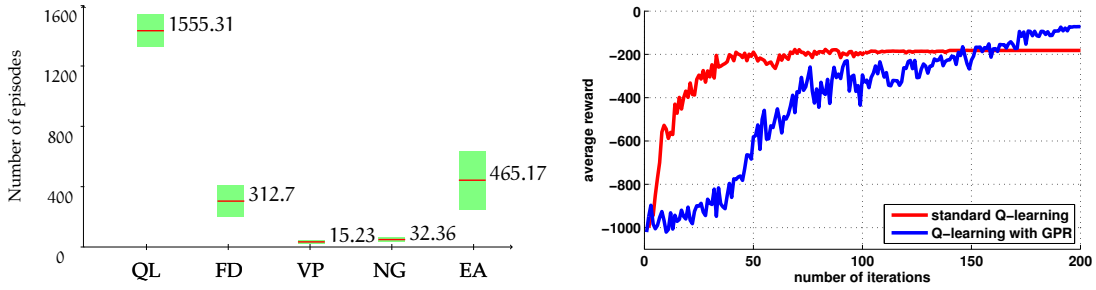
3. **Evolutionary methods** [Gomez and Miikkulainen, 2003] are black-box optimization algorithms. They optimize a parametric function by keeping a population of possible function parameters – called individuals –, and combining them based on the corresponding function values. In RL, individuals are policies and the function values are the corresponding average expected rewards [Moriarty et al., 1999]. Evolutionary algorithms are used with success in RL [Gomez et al., 2009; Koza and Rice, 1992], since they need no prior knowledge about the learning task.

Is has been shown that value function based methods converges asymptotically to the optimal policy when the accurate representation of the action value function is possible [Sutton and Barto, 1998]. However, when the action-state space is continuous – *e.g.*, joint angles, joint torques, in the case of robot control – function approximation has to be applied to model the value function.

We propose a Gaussian process regression approximation of the state-action function and a corresponding Q-learning based algorithm. We start the learning process by taking a GP with no training points – by definition, we assign the value of the prior for every state-action input uniformly. Then, the GP is gradually updated after each Q-learning step. Consider an episode τ consisting of a sequence of $\{(s_t, a_t, r_t)\}$ state-action-reward triplets. Then, the update looks as follows:

$$q \leftarrow Q_{\text{pred}}(s_t, a_t) + \alpha \left[r_{t+1} + \gamma \max_a Q_{\text{pred}}(s_{t+1}, a) - Q_{\text{pred}}(s_t, a_t) \right],$$

where $\alpha \in [0, 1]$ is the learning rate and $Q_{\text{pred}}(s, a)$ is the prediction of the current GP for the state-action pair (s, a). Then, we update this GP with the training point $((s_t, a_t), q)$.



(a) Pole balancing.

(b) Mountain car.

Figure 1: Experimental results. The performance for pole balancing has been measured by the number of episodes until convergence, for Q-learning (QL), finite difference method (FD), vanilla policy gradient method (VP), natural gradient method (NG) and evolutionary algorithm (EA).

Experiments

We present two simulated robot experiments on the pole balancing problem [Deisenroth and Rasmussen, 2011] and the mountain car problem [Rasmussen and Kuss, 2004]. We highlight the advantages and drawbacks of the presented learning methods in a robot control framework.

For the pole balancing problem, results based on 393 experiments are shown on Figure 1a where the variance of the convergence is displayed as well. As a measure of performance we used the average number of episodes needed by the algorithms to find a good policy. From the simulations reveals that the policy gradient based methods outperformed the other algorithms. The finite difference method is rather slow and the time of convergence has a huge variance. The vanilla policy gradient algorithm produced better results than the natural policy gradient, however, it failed to converge in 10% of the simulations which did not happen in the case of natural policy gradient. Divergence occurred when the robot was close to the optimal policy and the magnitude of the gradient was too small, therefore, the update of the parameters had almost no effect. Q-learning happened to be stable but produced the worst results since it does not scale well with high dimensional continuous state spaces.

Based on 400 runs of the mountain car problem, experiments show that simple Q-learning converges faster if we do not use GP function approximation – 61 episodes were needed by simple Q-learning whereas 146 episodes with the GP extension. This is not a surprising result since by using function approximations, we always loose accuracy. It is much more important to note that better results – fewer steps until the goal state was reached – were obtained in the GP case. To find an average solution, the Q-learning algorithm took 358 steps whereas only 171 steps were taken by the GP extension. Results show – see Figure 1b – that using GPs for the approximation of the action-value function in the mountain car task might lead to better solutions, however, longer time is needed until convergence.

4 Robot Model Learning for Tracking Control

In this section, we turn our attention to more realistic robot control problems: efficient control of robot architectures for tracking control.

The efficient control of robot architectures require optimal control methods. Control methods are based on the *model* of the robot. In the traditional setup, the model is defined by the physical parameters of the robot. Using the physical architecture based model, the desired joint-configurations (known as kinematic model) or the desired torque values (known as dynamic model) also have analytical forms. It has been applied with success mainly in the fields where the robot architecture was fixed and the external conditions were unchanging. Under the condition of fixed robot architecture and unchanging environment, analytical models provide accurate and efficient control algorithms.

However, under different conditions analytical solutions have several significant drawbacks: (1) Modern robots are getting more and more complex and expressing the robot model in terms of all parameters may be too complex. Although the analytical solution may exist, the computational requirements may be too expensive. (2) Analytic solutions make linearity assumptions about the robot model that may lead to inaccurate control and therefore might even fail the model. (3) The analytical solution will also fail when the physical model parameters are inaccurate or unknown. (4) Inaccuracy in the robot model can be caused by the noisy sensory inputs as a result of, *e.g.*, low quality sensors or uncalibrated cameras.

To overcome the presented drawbacks, an adaptive approach has been proposed to obtain accurate robot models. The goal is not to define the control model as a representation of the physical structure of the robot. Rather, we aim to find a mapping from sensory inputs to control outputs. We are not interested in the actual representation of the function, only its inputs and outputs are relevant. The function space we have chosen together with sampled data serve as the *model of the robot*. Next, we define more precisely the problem of tracking control we deal with.

Tracking Control

Tracking control tasks are usually formulated in the task-space when the end-effector of the robot arm has to follow a given trajectory. The solution for this task is not unique. For redundant robots, a mapping from task-space to configuration space is always non-unique: given a task-space position, there will frequently be several possible joint-space configurations forming a non-convex solution space [D'Souza et al., 2001] and one has to choose from these solutions in a *clever* way.

In this section, we propose a tacking control framework that is based on machine learning techniques and it is capable of controlling non-rigid robot architectures where standard methods fail. We also address the non-uniqueness of the solution and propose a *clever* way to choose among them.

The presented tracking control algorithm has three steps: (1) we model a joint model

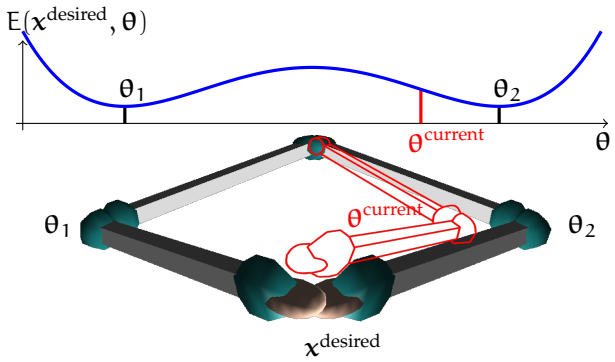


Figure 2: Illustration of the inverse kinematics prediction scheme. In the training set $x^{desired}$ has been reached by two different joint configurations θ_1 and θ_2, therefore, $E(x^{desired}, \theta_1) = E(x^{desired}, \theta_2)$. As prediction our algorithm will chose θ_2 since the current joint configuration $\theta^{current}$ is in the attraction range of θ_2.

of task-space joint-space coordinates using machine learning techniques; (2) apply local optimization to obtain the inverse kinematics mapping denoted with f^{-1}; (3) based on the inverse kinematics mapping, we use a joint-space controller on the controllable degrees of freedom (DoF) of the robot to obtain the desired torques. Next, we detail the first and the second step of the algorithm, since the third step is significantly to solve [Nguyen-Tuong and Peters, 2010].

Indirect Robot Model Learning

The key insight is that we can learn a model $E(x, \theta)$ in the joint input-output space and obtain prediction for target outputs by minimizing this model for a given input point, *i.e.*,

$$f^{-1}(x) \triangleq \underset{\theta \in \Theta}{\operatorname{argmin}} E(x, \theta), \tag{1}$$

where x are task-space positions and θ are joint-space positions.

An important question is how to perform the minimization from Equation (1). This question relates to the non-uniqueness of the inverse kinematics function: how does the minimization ensure that a convenient solution will be chosen when a desired end-effector $x^{desired}$ position can be reached by two different joint configurations θ_1 and θ_2 (see Figure 2)? In this case, the algorithms should predict θ_2 to avoid jerky movements. This behavior is desirable to achieve smooth trajectories in joint-space. We propose to start a gradient search from the current joint positions $\theta^{current}$.

A second question to answer is how to model $E(\cdot, \cdot)$ as to obtain an algorithm that can be used efficiently in real world setting. We propose three different approaches, *joint kernel support estimation*, *structured output Gaussian processes*, and an approximation using the forward kinematics model.

Using **Joint kernel support estimation** (JKSE), we model the energy function as the neg-

ative joint probability of $\mathbf{x}$ and θ, *i.e.*,

$$E(\mathbf{x}, \theta) \quad \stackrel{\circ}{=} \quad -p(\mathbf{x}, \theta). \tag{2}$$

JKSE models the joint probability distribution of inputs and outputs as a log-linear model of a joint feature function [Lampert and Blaschko, 2009]. Then, after simplifications, a prediction for one inverse kinematics prediction looks as follows:

$$f^{-1}(\mathbf{x}) \quad = \quad \underset{\theta \in \Theta}{\operatorname{argmax}} \; \boldsymbol{w}^\top \phi(\mathbf{x}, \theta),$$

where $\boldsymbol{w}$ are the parameters which generates a $p(\mathbf{x}, \theta)$ that explains the given training dataset $\mathbf{D} = \{(\mathbf{x}_i, \theta_i)\}_{i=1}^m$ with m training points best. The values of $\boldsymbol{w}$ are obtained using one-class support vector machines [Schölkopf et al., 2001]. The function $\phi(\cdot, \cdot)$ is the same feature function introduced in the context of kernels in Section 2 defined on the joint data.

Using **structured output Gaussian processes** (SOGP), the energy function $E(\cdot, \cdot)$ is expressed as the negative posterior mean of a GP, *i.e.*,

$$E(\mathbf{x}, \theta) \quad \stackrel{\circ}{=} \quad -\mu_{(\mathbf{x},\theta)}. \tag{3}$$

The inputs of the GP are the joint input-output data represented by a feature function $\phi(\mathbf{x}, \theta)$ and the outputs are all $\mathbf{1}$. Such an unbalanced training set can easily lead to over-fitting. To avoid over-fitting, the definition of a strong prior is essential. In the rest of this section, we assume a zero mean prior since it keeps the notations simple. The posterior mean of GP looks as follows:

$$\mu_{(\mathbf{x},\theta)} \quad = \quad \mathbf{k}_{(\mathbf{x},\theta)}^\top (\mathbf{K} + \sigma_0^2 \mathbf{I}_m)^{-1} \mathbf{1},$$

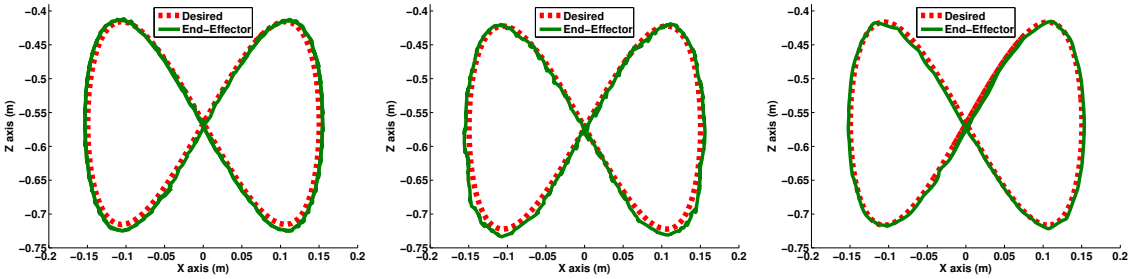
where $\mathbf{K} \in \mathfrak{R}^{m \times m}$ with $\mathbf{K}^{ij} = k((\mathbf{x}_i, \theta_i), (\mathbf{x}_j, \theta_j))$, $\mathbf{k}_{(\mathbf{x},\theta)} \in \mathfrak{R}^{m \times 1}$ with $\mathbf{k}_{(\mathbf{x},\theta)}^i = k((\mathbf{x}_i, \theta_i), (\mathbf{x}, \theta))$, $k_{(\mathbf{x},\theta)(\mathbf{x},\theta)} = k((\mathbf{x}, \theta), (\mathbf{x}, \theta))$, $\mathbf{I}_m$ is the identity matrix of size m, σ_0^2 is the variance of the measurement noise, and $\mathbf{1}$ is the unit vector of length m.

The third approach called "**forward Gaussian process modeling**" FWGP) is based on the observation that forward kinematics models – denoted with $f(\cdot)$ – are significantly simpler functions than the inverse mappings. Based on this insight, we construct our energy function such that it should depend on the forward model explicitly. Once the forward kinematics model is known, the energy function is defined as the Euclidean distance between the desired task-space position and the one predicted by the forward model, *i.e.*,

$$E(\mathbf{x}, \theta) \quad \stackrel{\circ}{=} \quad \|\mathbf{x} - f(\theta)\|^2. \tag{4}$$

We model the forward kinematics model with GPs. Given a training set $\mathbf{D} = \{(\mathbf{x}_i, \theta_i)\}_{i=1}^m$ with inputs θ_i and labels $\mathbf{x}_i$, the prediction for a new θ is a Gaussian–distributed random variable with mean μ_θ

$$\mu_\theta \quad = \quad \sum_{i=1}^{m} \alpha^i k(\theta, \theta_i) = \mathbf{k}_\theta^\top \boldsymbol{\alpha}, \tag{5}$$



(a) JKSE. (b) SOGP. (c) FWGP.

Figure 3: Results of tracking control for figure eight tracking, when the models have been trained offline. Good accuracy have been achieved with all three approaches.

where $k_{\theta\theta} = k(\theta, \theta)$, $\mathbf{k}_\theta \in \Re^{m\times 1}$ is a vector with elements $\mathbf{k}_\theta^i = k(\theta, \theta_i)$ and $\mathbf{K} \in \Re^{m\times m}$ is a matrix with elements $\mathbf{K}^{ij} = k(\theta_i, \theta_j)$. The function $k : \Re^n \times \Re^n \to \Re$ is a positive definite kernel and $\boldsymbol{\alpha} \in \Re^{m\times 1}$, where $\boldsymbol{\alpha} = (\mathbf{K} + \mathbf{I}_m \sigma_0^2)^{-1}\mathbf{x}$ are the parameters of the GP. The predictive mean of the GPs is used as the prediction of the forward model from Equation (5), *i.e.*, $f(\theta) = \mu_\theta$ [1].

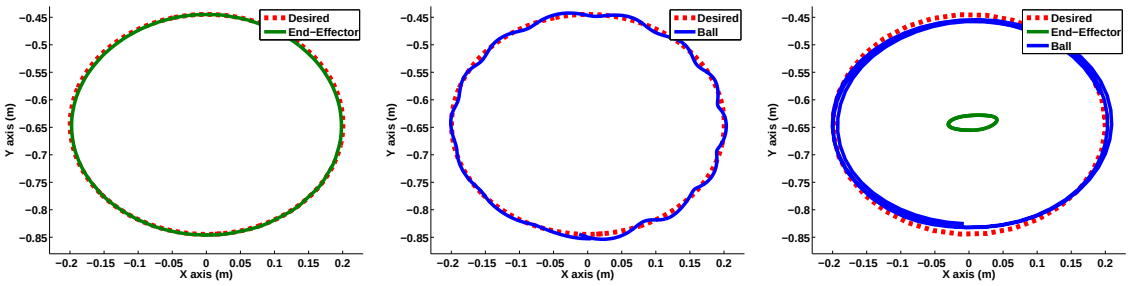
Note that the gradients of all the energy function from Equation (2), Equation (3), and Equation (4) have analytical form, thus, the gradient search from Equation (1) can be done efficiently using second order gradient search algorithms [Snyman, 2005].

Experiments

We present an empirical evaluation of the proposed methods for task-space tracking. The algorithm is applied to learn the inverse kinematics of a Barrett WAM robot arm [Bócsi et al., 2011b], and track a figure eight with different settings. The results of figure eight tracking are shown on Figure 3, where it can be seen that good accuracy was achieved. We measured the accuracy of tracking control. Experiments show that FWGP outperformed JKSE and SOGP. Furthermore, FWGP almost achieved the accuracy of the analytical model. This result is slightly surprising in the light of the number of the points the models were based on. The JKSE model was based on 8456 support vectors, the SOGP model on 200 base points, and the FWGP model on 31 base points.

In a different experiment, we made the simulated model more complex by attaching a ball to the end-effector of the arm with a 20 cm string. The swinging motion of the ball produced a non-linear system. In this new setting, we performed task-space tracking where the position of the ball was considered as the desired position. The task was to track a circle figure with 20 cm radius, see Figure 4a, on the horizontal plane. The task was performed in two different settings: (1) when the desired point moved slowly along the circle (one turn took 24 seconds) and (2) when the desired point moved fast along the circle (one turn took 0.62 seconds).

[1] We used a different GP for each output dimension, *i.e.*, $f_j(\theta) = \mu_\theta^j$, $j = \overline{1,3}$ where $f_j(\theta)$ is the j-th component of $f(\theta)$.



(a) End-effector trajectory with slow circle learning.

(b) Ball trajectory with slow circle learning.

(c) End-effector and ball trajectory with fast circle learning.

Figure 4: Online task-space tracking learning of a circle following task by a simulated 7-DoF Barrett WAM robot arm with a ball attached on it. (a) When the movement of the desired point on the circle is slow, the end-effector is placed above the desired trajectory while (b) the ball is hanging down and its swinging is damped – the swinging is considered noise by FWGP. (c) When the desired point moves fast on the circle the end-effector moves inside the desired circle while the ball swings around along the desired trajectory – the centrifugal force is incorporated into the control model.

In the first case, FWGP learned to move the end-effector of the arm right above the desired circle while the ball was moving along the desired trajectory since the swinging of the ball was damped. The speed of the end-effector was the same as the speed of the ball. It took four minutes of learning to achieve the learning accuracy presented on Figure 4a and Figure 4b. The GP model was built from 20-25 data points. In the second case, the trajectory of the end-effector of the arm was moving fast inside the desired circle and used the centrifugal force to swing the ball around, see Figure 4c. After 20 minutes of learning, the GP model was built from 13-15 data points. Following the fast circle movement with the arm itself would be impossible, since it would reach the physical limits of the robot.

We emphasize that the same parameter settings were used for both experiments. Thus, the adaptive behavior depends weakly on the hyper-parameters of the GP. In the first case, FWGP considered the swinging motion of the ball as noise and the trajectory of the ball followed a similar trajectory as the end-effector. On the other hand, in the second experiment, the GP model incorporated into the control model the centrifugal force as well.

5 Transfer Learning for Robot Models

The motivation of this section comes from human learning. A fundamental difference between human and machine learning is that robots have no prior knowledge of the world, whereas humans have at their disposal the past experiences. In this context, even though the new task has not yet been performed by the human learner, the skills acquired from previous experience would certainly speed up learning the new task. For example, when one has to learn how to play table tennis, the fact that he uses his hand regularly improves

the learning process. Furthermore, if he has done *similar* activities before, such as tennis, the improvement will be more significant. In this section, we consider the scenario when the robot uses its own – or other robots – past experiences to improve its learning speed. Similar to the previous example, let us consider a humanoid robot with two arms which learned a given task (*e.g.*, grabbing a cup) with one of its arms. We propose a method that helps to learn a similar task with the other hand based on the knowledge gained from the first task.

The aim of transferring knowledge between different tasks is not novel, it has been considered in machine learning under the name of *transfer learning* [Pan and Yang, 2010; Arnold et al., 2007; Pan et al., 2008; Taylor and Stone, 2009]. Transfer learning is based on the insight that the learning process of a given task can be improved when *knowledge* available from other learning problems is reused. It has been shows that the usage of such knowledge can be beneficial in many machine learning frameworks [Pan and Yang, 2010]. In robotics, it was mainly applied to improve reinforcement learning tasks as part of lifelong learning [Thrun and Mitchell, 1993]. This is the general idea that any learning process must be based on information gained from previously learned tasks.

Transfer Learning in Robotics

We aim to improve the learning process of robot models (such as forward kinematics, inverse kinematics, and inverse dynamics) when we assume that information can be transferred from past experiments in the form of additional data-sets. We define a source task, a task that has already been learned, and a target task, a problem that is difficult to learn for some reasons. We consider the case when a source task has a data-set $\mathbf{D}^s = \{(\boldsymbol{\theta}_i^s, \mathbf{x}_i^s)\}_{i=1}^N$ with N training points, and we aim to improve the learning of a target task that has a training set $\mathbf{D}^t = \{(\boldsymbol{\theta}_i^t, \mathbf{x}_i^t)\}_{i=1}^K$ that has K training points. We aim to improve learning forward kinematics and we assume the inputs are joint configurations and the labels are end-effector positions.

In the first step, we reduced the dimension of both data-sets to the same dimension. In our experiments, we used principal component analysis as the dimensionality reduction method. By applying PCA, we assume linear relationship between the low and high dimensional subspaces. First, we center the data, *i.e.*, subtract the mean, then whiten it, *i.e.*, divide it with the standard deviation. The projection looks as follows

$$\begin{aligned} \mathbf{s} &= \mathbf{B}_s(\mathbf{d}^s - \boldsymbol{\mu}_s) \\ \mathbf{t} &= \mathbf{B}_t(\mathbf{d}^t - \boldsymbol{\mu}_t), \end{aligned}$$

where $\mathbf{d}^s \in \mathbf{D}^s$ and $\mathbf{d}^t \in \mathbf{D}^t$ are the data points of the source data-set and target data-set, and $\mathbf{s} \in \mathbf{M}^s$ and $\mathbf{t} \in \mathbf{M}^t$ are the respective data points of the low dimensional manifolds. The values $\boldsymbol{\mu}_s = \mathbf{E}\{\mathbf{D}^s\}$ and $\boldsymbol{\mu}_t = \mathbf{E}\{\mathbf{D}^t\}$ are the means of the original data. Matrices $\mathbf{B}_s$ and $\mathbf{B}_t$ are transformation matrices obtained such that the variances of $\mathbf{M}^s$ and $\mathbf{M}^t$ are maximized. For details consult Lee and Verleysen [2007].

In the second step, we model the manifold alignment function as a linear mapping f :

$\mathbf{M}^s \rightarrow \mathbf{M}^t$, with

$$f(\mathbf{s}) = \mathbf{A}\mathbf{s}, \tag{6}$$

where $\mathbf{A} \in \Re^{J \times J}$ is a transformation matrix with J the dimension of the manifolds. We define two cases of alignment scenarios.

In the first case, a direct correspondence between the data points is known. By direct correspondence we mean that $\mathbf{D}^s$ and $\mathbf{D}^t$ contain the same number of points and the points come in pairs. This setup can be used, for example, when the same task has been performed both in the source space and in the target space.

In the second case, there is no direct correspondence between the points of $\mathbf{D}^s$ and $\mathbf{D}^t$. They may even contain different number of points, *i.e.*, $|\mathbf{D}^s| \neq |\mathbf{D}^t|$.

- In the first case – called **direct correspondence alignment** –, we assume a linear function and minimize the expected loss of the transformation. We want to find the parameter $\mathbf{A}$ from Equation (6) such that the expectation of the error of the transformation is minimized, *i.e.*,

  $$\mathbf{A} = \underset{\mathbf{A}}{\operatorname{argmin}} \, \mathbf{E}\left\{(\mathbf{t} - \mathbf{A}\mathbf{s})^\top (\mathbf{t} - \mathbf{A}\mathbf{s})\right\},$$

  where $\mathbf{E}\{\cdot\}$ denotes the expectation operator. The solution for A has the following form:

  $$\mathbf{A} = \mathbf{\Sigma}_{ss}^{-1}\mathbf{\Sigma}_{ts}, \tag{7}$$

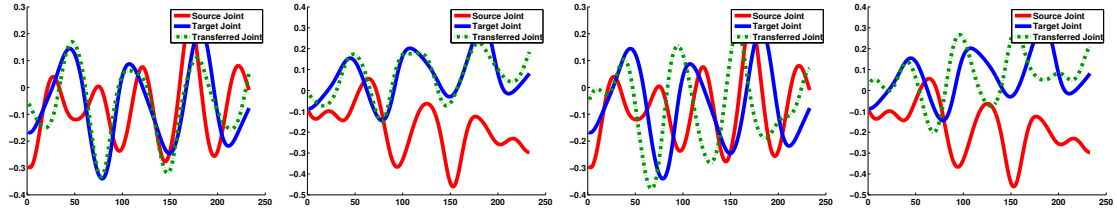
  where $\mathbf{\Sigma}_{ss}$, $\mathbf{\Sigma}_{tt}$, and $\mathbf{\Sigma}_{ts}$ are covariance matrices.
- In the second case – called **rough alignment** –, the distance between the distributions defined by the points of $\mathbf{M}^s$ and $\mathbf{M}^t$ is minimized. In what follows we assume that both data-sets are Gaussian distributed and we minimize the distance between the two Gaussians $p(\mathbf{M}^s)$ and $p(\mathbf{M}^t)$. Defined as the Kullback-Leibler divergence [Kullback, 1959], the divergence has an analytical form when the distributions are Gaussians. As a result, the $\mathbf{A}$ that minimizes the distance is the solution of the following equation:

  $$\mathbf{\Sigma}_{tt} = \mathbf{A}\mathbf{\Sigma}_{ss}\mathbf{A}^\top.$$

  This expression is quadratic in $\mathbf{A}$ and does not have a unique solution. A matrix $\mathbf{A}$ that solves the equation above can be obtained using the eigenvalue decomposition of the covariance matrices [Trefethen and Bau, 1997]. The proposed solution looks as follows:

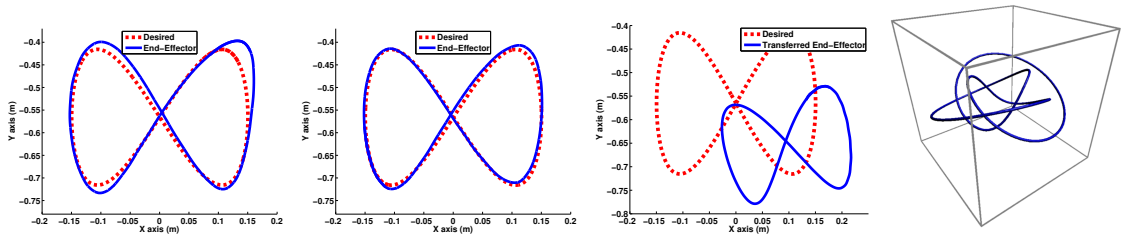
  $$\mathbf{A} = \mathbf{U}_t\mathbf{\Lambda}_t^{1/2}\mathbf{\Lambda}_s^{-1/2}\mathbf{U}_s^\top,$$

  where $\mathbf{U}_s$ and $\mathbf{U}_t$ are rotation matrices (*i.e.*, $\mathbf{U}_s\mathbf{U}_s^\top = \mathbf{I}$) with the eigenvector of $\mathbf{\Sigma}_{ss}$ and $\mathbf{\Sigma}_{tt}$, and $\mathbf{\Lambda}_s$ and $\mathbf{\Lambda}_t$ are diagonal matrices with the eigenvalues of $\mathbf{\Sigma}_{ss}$ and $\mathbf{\Sigma}_{tt}$, respectively.



(a) Joint 1. (b) Joint 3. (c) Joint 1. (d) Joint 3.

Figure 5: Results for direct correspondence alignment and for rough alignment.



(a) After data transfer *with-out* further learning. (b) After data transfer *with* further learning. (c) Transfer from the Master arm to the Barrett arm. (d) Trefoil knot.

Figure 6: Tracking control results after three seconds of online learning, then, using the transfer learning algorithm

Experiments

We conducted experiments on robots with different architectures to emphasize two features of our method: (1) the information loss induced by dimensionality reduction is not significant and (2) the expressive power of the linear function used for manifold alignment is sufficiently good to achieve good performance.

The experiment was performed on a simulated Sarcos Master arm [Nguyen-Tuong et al., 2009b] with eight DoFs and a simulated Barrett WAM arm [Nguyen-Tuong et al., 2009b] with seven DoFs. Both source and target tasks were to do task-space tracking of a trefoil knot (Figure 6d). We used the analytical controllers of the robots to collect data for both tasks. After running each task with the same speed for one minute, we had data points with direct correspondence. We applied the direct correspondence method on this data-set. Figure 5a and Figure 5b shows that after estimating $\mathbf{A}$ with Equation (7), we could transfer the data points from the source data-set (red) to the target data-set (blue) with good efficiency (dashed green). To see how much information can be caught if we do not assume direct correspondence, but only distribution minimization, we applied the rough alignment method to this data-set. Result are presented in Figure 5c and Figure 5d. It can be seen that the transformation is not accurate, but it captures the correct mean and variance of the transferred data, as expected.

In a next experiment, we directly transferred a task from the Master arm to the Barret arm using the transition matrix obtained from the previous data-sets. We draw a figure

eight with the Master arm (using the analytical controller) that was placed inside the space defined by the trefoil knot. After transforming the joint-space trajectories and following the transformed trajectories with the Barrett arm, the figure eight presented in Figure 6c has been obtained. Note that showing a desired figure eight in Figure 6c may be misleading since there is no ground truth of task transformation. We defined the desired figure eight as the figure tracked by the analytical controller of the Barrett arm with the same initial posture as the trefoil knot. This intuitive definition is based on the principle that the data transfer must incorporate only translation, rotation and scaling, and no deformation of the tracked figures.

In the last experiment, we used the same robot two architecture for the source task and for the target task as in the previous experiment. We used the source data-set from the previous experiment (the trefoil knot). The target task was to speed-up the forward kinematics model learning of the Barret arm. The forward kinematics model has been approximated with sparse online Gaussian processes and used to perform task-space control. Without transfer learning it takes from 20 seconds to four minutes to learn this model online. After performing quasi-random movements for three seconds with the Barrett arm, we applied the distribution alignment approach. We stopped the learning process after the three seconds of burn-in period and used the transferred points to further train the forward kinematics model. Figure 6a shows the figure eight as a result. The shape eight is not perfect, however, we needed only three seconds of learning and the transfer algorithm. We repeated the experiment but now the learning process was not stopped after three seconds, only the Master arm data-set has been used to gain additional training samples. Figure 6b shows that if learning is not stopped an accurate figure eight tracking is achievable after three seconds of learning and the transfer algorithm.

6 Conclusions and Further Research

In this thesis we addressed the problem of robot model learning in different contexts. The motivation of learning robot models, instead of using its physical parameters, comes from several sources: (1) the physical parameters may be unknown, (2) the physical parameters may be inaccurate, (3) the control model is too complex based on the physical parameters, and (4) the observations may be corrupted by noise, thus, noise modeling is required.

First, we investigated RL algorithms in the context of robot model learning. The goal was to see how RL methods work in this domain where typical features of the problem are continuous and high dimensional state spaces. We presented the prevailing RL approaches (value function based methods, policy gradient methods, and evolutionary algorithms) and compared them from both theoretical and practical point of view. The experiments have been conducted on a simulated pole balancing robot as a standard RL benchmark problem. Results showed that direct policy approximation based methods such as policy gradient and evolutionary algorithms produced better results in continuous high dimensional domains such as robot control.

Next, an extension of the Q-learning algorithm was presented. The state-action value function was modeled with GPs. Experiments on a mountain car problem showed that the GP extension converged to an overall better solution than the standard Q-learning algorithm.

Then, we addressed more specific robot model learning applications. The inverse kinematics model of arbitrary robot architectures has been approximated from sampled data. The novelty of the proposed algorithm is that an indirect way of modeling the inverse kinematics mapping was used. A joint energy function of task-space coordinates and joint-space coordinates was modeled. Then, applying local optimization techniques, the solution for the desired task-space position has been obtained. The indirect modeling provides a natural way for tackling the problem of non-uniqueness of inverse kinematics, *i.e.*, one model can contain more than one solution for a given task-space position. The *right* solution is selected based on the natural principal that smooth robot movement is a requirement.

We proposed three ways of modeling the joint energy function: (1) JKSE, (2) SOGP, and (3) a method based on the forward kinematics model of the robot called FWGP. We conducted experiments on a Barrett WAM robot arm with different settings. Results showed that the best accuracy was obtained with FWGP that converged to the control model provided by the analytical solution. FWGP could also be used online and outperformed LWPR, one of the best inverse kinematics learning methods. JKSE and SOGP appeared to be too slow to be used online. Furthermore, FWGP could learn the inverse kinematics model of a non-rigid robot arm where all the other methods failed.

We proposed a transfer learning based paradigm that can improve arbitrary robot model learning algorithms. The key idea was that knowledge gained from other robot model learning experiments can be used to improve the current learning problem. We transferred the sampled from the other experiments such that it provided useful information for the current task. Experiments showed that faster model learning and better accuracy can be gained using the additional data.

Further Research

We believe that the proposed algorithms can be improved in several ways. The GP extension of the state-action value function has been investigated extensively in the literature and several drawbacks have been pointed out: inefficient handling of discontinuities, problem of correlated samples, high computational requirements. The presented method also has these drawbacks, and it is desired to alleviate them. Another possible improvement can be gained by using the probabilistic modeling of the state-action value function. Beside the posterior mean, the variance of the prediction may also provide useful information. For example, it can guide the robot to less explored regions with high posterior variance [Jakab and Csató, 2012].

For the inverse kinematics modeling, we proposed three different ways of modeling the joint energy function. Other approximations (or using the probabilistic modeling as in the case of the state-action value function) may lead to more accurate models. An important

property of these models must be fast evaluation since a minimization must be performed over them at each inverse kinematics prediction. Finding more suitable search algorithms specific for this problem may also improve the prediction. The online application of the learning method is also an appealing feature, thus, one must choose the models such that the online update would be possible. To use the algorithm for humanoid robots, it must be investigated how well it scales for robots with many DoFs where the search space is higher dimensional.

The presented idea of using a joint energy function for inputs and outputs, then, minimize it for a given input, can also be used for other types of robot models. It would be interesting to try it for inverse dynamics modeling or operational space control.

The transfer learning extension of robot model learning can be improved by applying other techniques borrowed from the transfer learning framework. Other approaches may lead to more efficient information transfer from other experiments. A specially appealing transfer learning approach seems to the Bayesian knowledge transfer, where information gained from other experiments serves as a prior for the current learning task.

Another interesting future direction may be to try to combine how information between different robot model learning algorithms can be transferred, *e.g.*, how inverse kinematics learning can be improved when the inverse dynamics model is already learned or vice versa.

Bibliography

M. Alvarez and N. D. Lawrence. Sparse convolved Gaussian processes for multi-output regression. In *Neural Information Processing Systems*, pages 57–64, 2008.

S.-I. Amari and H. Nagaoka. *Methods of Information Geometry (Translations of Mathematical Monographs)*. American Mathematical Society, 2001.

V. R. d. Angulo and C. Torras. Learning inverse kinematics via cross-point function decomposition. In *Proceedings of the International Conference on Artificial Neural Networks*, (ICANN 2002), pages 856–864, London, UK, UK, 2002. Springer-Verlag.

B. D. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57:469–483, 2009.

A. Arnold, R. Nallapati, and W. W. Cohen. A comparative study of methods for transductive transfer learning. In *Proc. IEEE Int. Conf. on Data Mining Workshops*, pages 77–82, 2007.

G. H. Bakir, T. Hofmann, B. Schölkopf, A. J. Smola, B. Taskar, and S. V. N. Vishwanathan, editors. *Predicting Structured Data*. Neural Information Processing. The MIT Press, 2007.

D. Barber. Bayesian methods for supervised neural networks. In *Handbook of Brain Theory and Neural Networks*. MIT Press, 2002.

D. Barber and C. M. Bishop. *Ensemble learning in Bayesian neural networks.*, pages 215–237. Springer-Verlag, Berlin, 1998.

H. Benbrahim, J. Doleac, J. Franklin, , and O. Selfridge. Real-time learning: A ball on a beam. In *Proceedings of the International Joint Conference on Neural Networks*, volume 1, pages 98–103, 1992.

D. P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 1995.

C. M. Bishop. *Pattern recognition and machine learning*. Springer, 1st edition, 2006.

B. Bócsi and L. Csató. Reinforcement learning algorithms in robotics. In M. Frentiu, H. F. Pop, and S. Motogna, editors, *KEPT-2011:Knowledge Engineering Principles and Techniques International Conference, Selected Papers.*, pages 131–143. Presa Universitara Clujeana, 2011a.

B. Bócsi, L. Csató, and Jan Peters. Structured output Gaussian processes. Technical report, Babes-Bolyai University, 2011a. URL http://www.cs.ubbcluj.ro/~bboti/pubs/sogp_2011.pdf.

B. Bócsi, D. Nguyen-Tuong, L. Csató, B. Schoelkopf, and J. Peters. Learning inverse kinematics with structured prediction. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 698–703, San Francisco, USA, 2011b.

B. Bócsi, L. Csató, B. Schölkopf, and J. Peters. Indirect robot model learning for tracking control. *Robotics and Autonomous Systems (submitted on 10 September 2012)*, 2012a.

B. Bócsi, P. Hennig, L. Csató, and J. Peters. Learning tracking control with forward models. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 259–264, St. Paul, MN, USA, 2012b.

B. A. Bócsi and L. Csató. Reinforcement learning algorithms in robotics. *Studia Universitatis Babes-Bolyai Series Informatica*, LVI(2):61–67, 2011b.

B. A. Bócsi, H. Jakab, and L. Csató. Nonparametric methods in robotics. In *Proceedings of the 8th Joint Conference on Mathematics and Computer Science (Abstract)*, page 8, Komarno, Slovakia, 2010.

E. V. Bonilla, K. M. Chai, and C. K. I. Williams. Multi-task Gaussian process prediction. In J. C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20 (NIPS)*. MIT Press, Cambridge, MA, 2008.

B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, COLT '92, pages 144–152, New York, NY, USA, 1992. ACM.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

G. Cauwenberghs and T. Poggio. Incremental and decremental support vector machine learning. In *Advances in Neural Information Processing Systems*, volume 13, 2001.

K. M. Chai, C. Williams, S. Klanke, and S. Vijayakumar. Multi-task Gaussian process learning of robot inverse dynamics. In *Advances in Neural Information Processing Systems (NIPS)*, pages 265–272, 2009.

C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Y.-S. Choi. Least squares one-class support vector machine. *Pattern Recognition Letters*, 30: 1236–1240, 2009.

J. J. Craig. *Introduction to Robotics: Mechanics and Control*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2nd edition, 1989.

R. Crites and A. Barto. Improving elevator performance using reinforcement learning. In *Neural Information Processing Systems*, pages 1017–1023. MIT Press, 1996.

L. Csató and M. Opper. Sparse on-line Gaussian processes. *Neural Computation*, 14(3):641–668, 2002.

W. Dai, G.-R. Xue, Q. Yang, and Y. Yu. Transferring naive Bayes classifiers for text classification. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*, pages 540–545, 2007a.

W. Dai, Q. Yang, G. R. Xue, and Y. Yu. Boosting for transfer learning. In *Proceedings of the 24th international conference on Machine learning (ICML)*, pages 193–200. ACM, 2007b.

V. R. de Angulo and C. Torras. Learning inverse kinematics: reduced sampling through decomposition into virtual robots. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 38(6):1571–1577, 2008.

J. F. G. de Freitas. *Bayesian methods for neural networks*. PhD thesis, Trinity College. University of Cambridge, 1999.

B. de Kruif and T. de Vries. Pruning error minimization in least squares support vector machines. *IEEE Transactions on Neural Networks*, 14(3):696–702, 2003.

M. P. Deisenroth and C. E. Rasmussen. PILCO: A Model-Based and Data-Efficient Approach to Policy Search. In L. Getoor and T. Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning*, Bellevue, WA, USA, 2011.

M. P. Deisenroth, C. E. Rasmussen, and J. Peters. Gaussian process dynamic programming. *Neurocomputing*, 72(7-9):1508–1524, 2009.

D. Demers and K. Kreutz-Delgado. Learning global direct inverse kinematics. In *Advances in Neural Information Processing Systems*, pages 589–595. Morgan Kaufmann, 1992.

R. Detry, D. Kraft, O. Kroemer, L. Bodenhagen, J. Peters, N. Krüger, and J. Piater. Learning Grasp Affordance Densities. *Paladyn Journal of Behavioral Robotics*, 2(1):1–17, 2011.

L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Applications of Mathematics. Springer, 1996.

A. D'Souza, S. Vijayakumar, and S. Schaal. Learning inverse kinematics. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*. Piscataway, NJ: IEEE, 2001.

A. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing (Natural Computing Series)*. Springer, 2008.

Y. Engel, S. Mannor, and R. Meir. Bayes meets Bellman: the Gaussian process approach to temporal difference learning. In *Proceedings of the 20th International Conference on Machine Learning (ICML)*, pages 154–161, 2003.

Y. Engel, S. Mannor, and R. Meir. Reinforcement learning with Gaussian processes. In *Proceedings of the 20th International Conference on Machine Learning*, pages 201–208, 2005.

L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:594–611, 2006.

T. S. Ferguson. A Bayesian analysis of some nonparametric problems. *The Annals of Statistics*, 1(2):209–230, 1973.

M. Galassi, J. Davies, J. Theiler, B. Gough, G. Jungman, M. Booth, and F. Rossi. *Gnu Scientific Library: Reference Manual*, 2003. Software `http://www.gnu.org/software/gsl/manual/html_node/index.html`.

M. Ghavamzadeh and Y. Engel. Bayesian policy gradient algorithms. In *Advances in Neural Information Processing Systems 19*. MIT Press, 2007.

J. Ghosh and R. Ramamoorthi. *Bayesian Nonparametrics*. Springer Series in Statistics. Springer-Verlag, 2003.

J. Gittins, K. Glazebrook, and R. Weber. *Multi-Armed Bandit Allocation Indices*. John Wiley & Sons, 2011.

F. Gomez and R. Miikkulainen. Active guidance for a finless rocket using neuroevolution. *Genetic and Evolutionary Computation - GECCO 2003*, pages 213–213, 2003.

F. Gomez, J. Schmidhuber, and R. Miikkulainen. Accelerated neural evolution through cooperatively coevolved synapses. *Journal of Machine Learning Research*, 9:937–965, 2009.

D. Grollman. *Sparse Online Gaussian Process C++ Library*, 2012. Software http://cs.brown.edu/people/dang/code.shtml.

P. Hennig. Optimal reinforcement learning for Gaussian systems. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 325–333, 2011.

J. Huang, A. Gretton, B. Schölkopf, A. J. Smola, and K. M. Borgwardt. Correcting sample selection bias by unlabeled data. In *Advances in Neural Information Processing Systems (NIPS)*. MIT Press, 2007.

H. Jakab and L. Csató. Reinforcement learning with guided policy search using Gaussian processes. In *International Joint Conference on Neural Networks (IJCNN)*, Brisbane, QLD, June 10-15 2012.

H. Jakab, B. A. Bócsi, and L. Csató. Non-parametric value function approximation in robotics. In H. F. Pop, editor, *MACS2010: The 8th Joint Conference on Mathematics and Computer Science*, volume Selected Papers, pages 235–248, Komarno, Slovakia, 2011. Györ-:NOVADAT.

E. Jaynes. *Probability Theory: The Logic of Science*. Cambridge University Press, 2003.

M. Johnson. PCFG models of linguistic tree representations. *Computatoinal Linguistics*, 24: 613–632, 1998.

M. I. Jordan and D. E. Rumelhart. Forward models: Supervised learning with a distal teacher. *Cognitive Science*, 16:307–354, 1992.

S. Kakade. A natural policy gradient. In *Neural Information Processing Systems*, pages 1531–1538, 2001.

M. Kalakrishnan, J. Buchli, P. Pastor, and S. Schaal. Learning locomotion over rough terrain using terrain templates. In *intelligent robots and systems, 2009. iros 2009. ieee/rsj international conference on*, pages 167–172, 2009.

R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

R. E. Kass and A. E. Raftery. Bayes factors. *Journal of the American Statistical Association*, 90 (430):773–795, 1995.

O. Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *Robotics and Automation, IEEE Journal of*, 3(1):43–53, 1987.

Z. Kira. Transferring embodied concepts between perceptually heterogeneous robots. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, pages 4650–4656, 2009.

J. Kober, B. J. Mohler, and J. Peters. Imitation and reinforcement learning for motor primitives with perceptual coupling. In *From Motor Learning to Interaction Learning in Robots*, pages 209–225. Springer, 2010.

J. R. Koza and J. P. Rice. Automatic programming of robots using genetic programming. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 194–201. The MIT Press, 1992.

S. Kullback. *Information Theory and Statistics*. Wiley, New York, 1959.

C. H. Lampert and M. B. Blaschko. Structured prediction by joint kernel support estimation. *Machine Learning*, 77:249–269, 2009.

P. Laskov, C. Gehl, S. Krüger, and K.-R. Müller. Incremental support vector learning: Analysis, implementation and applications. *J. Mach. Learn. Res.*, 7:1909–1936, 2006.

N. D. Lawrence, M. Seeger, and R. Herbrich. Fast sparse Gaussian process methods: The informative vector machine. In *Neural Information Processing Systems*, pages 609–616. MIT Press, 2002.

M. Lázaro-Gredilla, J. Quiñonero Candela, C. E. Rasmussen, and A. R. Figueiras-Vidal. Sparse spectrum Gaussian process regression. *Journal of Machine Learning Research*, 99: 1865–1881, 2010.

Y. Lecun, S. Chopra, R. Hadsell, F. J. Huang, G. Bakir, T. Hofman, B. Schoelkopf, A. Smola, and B. T. (eds). A tutorial on energy-based learning. In *Predicting Structured Data*. MIT Press, 2006.

J. Lee and M. Verleysen. *Nonlinear Dimensionality Reduction*. Springer, 2007.

D. Liberzon. *Calculus of Variations and Optimal Control Theory: a Concise Introduction*. Princeton University Press, 2012.

A. Liegeois. Automatic supervisory control of the configuration and behavior of multibody mechanisms. *IEEE Transactions on Systems, Man, and Cybernetics*, 7(12):842–868, 1977.

D. J. C. Mackay. Bayesian methods for backpropagation networks. In E. Domany, J. L. van Hemmen, and K. Schulten, editors, *Models of Neural Networks III*, chapter 6, pages 211–254. Springer, 1994.

C. D. Manning and H. Schütze. *Foundations of statistical natural language processing*. MIT Press, Cambridge, MA, USA, 1999.

Z. Marx, M. T. Rosenstein, L. P. Kaelbling, and T. G. Dietterich. Transfer learning with an ensemble of background tasks. In *Advances in Neural Information Processing Systems*. MIT Press, 2005.

A. McCallum and C. Sutton. An introduction to conditional random fields for relational learning. In L. Getoor and B. Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press, 2006.

F. S. Melo and M. I. Ribeiro. Q-learning with linear function approximation. In *Proceedings of the 20th Annual Conference on Learning Theory*, pages 308–322. Springer-Verlag, 2007.

D. E. Moriarty, A. C. Schultz, and J. J. Grefenstette. Evolutionary algorithms for reinforcement learning. *Journal of Artificial Intelligence Research*, 11:241–276, 1999.

J. Nakanishi, R. Cory, M. Mistry, J. Peters, and S. Schaal. Operational Space Control: A Theoretical and Empirical Comparison. *Int. J. Rob. Res.*, 27(6):737–757, 2008.

R. Neal. Regression and classification using Gaussian process priors (with discussion). *Bayesian Statistics*, 6:475–501, 1999.

K. Neumann, M. Rolf, J. J. Steil, and M. Gienger. Learning inverse kinematics for pose-constraint bi-manual movements. In *Proceedings of the 11th international conference on Simulation of adaptive behavior: from animals to animats*, SAB'10, pages 478–488, 2010.

D. Nguyen-Tuong and J. Peters. Using model knowledge for learning inverse dynamics. In

Proceedings of the IEEE International Conference on Robotics and Automation, pages 2677–2682, 2010.

D. Nguyen-Tuong, M. W. Seeger, and J. Peters. Model learning with local Gaussian process regression. *Advanced Robotics*, 23(15):2015–2034, 2009a.

D. Nguyen-Tuong, M. W. Seeger, and J. Peters. Model learning with local Gaussian process regression. *Advanced Robotics*, 23(15):2015–2034, 2009b.

D. Nguyen-Tuong, M. W. Seeger, and J. Peters. Real-time local GP model learning. In *From Motor Learning to Interaction Learning in Robots*, pages 193–207. Springer, 2010.

S. Nolfi and D. Parisi. Learning to adapt to changing environments in evolving neural networks. In *Adaptive Behavior*, pages 75–98, 1997.

M. Opper. *A Bayesian approach to on-line learning*, pages 363–378. Cambridge University Press, 1998.

E. Oyama and S. Tachi. Modular neural net system for inverse kinematics learning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3239 – 3246, 2000.

E. Oyama, N. Y. Chong, A. Agah, T. Maeda, and S. Tachi. Inverse kinematics learning by modular architecture neural networks with performance prediction networks. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1006–1012, 2001.

S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

S. J. Pan, J. T. Kwok, and Q. Yang. Transfer learning via dimensionality reduction. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*, 2008.

E. Parzen. On Estimation of a Probability Density Function and Mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076, 1962.

J. Peters and S. Schaal. Learning to control in operational space. *International Journal of Robotics Research*, 27(2):197–212, 2008a.

J. Peters and S. Schaal. Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 21(4):682–697, 2008b.

J. Peters, S. Vijayakumar, and S. Schaal. Reinforcement Learning for Humanoid Robotics. In *Conference on Humanoid Robots*, 2003.

F. M. Phelps and J. H. Hunter. An analytical solution of the inverted pendulum. *American Journal of Physics*, 33, Issue 4:285, 1965.

F. Pourboghrat. Neural networks for learning inverse-kinematics of redundant manipulators. In *Proceedings of the 32nd Midwest Symposium on Circuits and Systems*, volume 2, pages 760–762, 1989.

M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley-Interscience, 1994.

J. Quiñonero Candela and C. E. Rasmussen. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6:1939–1959, 2005.

L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, pages 257–286, 1989.

R. Raina, A. Y. Ng, and D. Koller. Constructing informative priors using transfer learning. In *Proceedings of the 23rd international conference on Machine learning*, ICML '06, pages 713–720, New York, NY, USA, 2006. ACM.

A. Ranganathan, M.-H. Yang, and J. Ho. Online sparse Gaussian process regression and its applications. *IEEE Transactions on Image Processing*, 20(2):391–404, 2011.

C. E. Rasmussen and M. Kuss. Gaussian processes in reinforcement learning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 751–759. MIT Press, 2004.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

R. F. Reinhart and J. J. Steil. Recurrent neural associative learning of forward and inverse kinematics for movement generation of the redundant pa-10 robot. In *Proceedings of the 2008 ECSIS Symposium on Learning and Adaptive Behaviors for Robotic Systems*, LAB-RS '08, pages 35–40, Washington, DC, USA, 2008. IEEE Computer Society.

R. F. Reinhart and J. J. Steil. Reaching movement generation with a recurrent neural network based on learning inverse kinematics for the humanoid robot icub. In *IEEE CONF. HUMANOID ROBOTICS*, 2009.

C. P. Robert and G. Casella. *Monte Carlo Methods*. Springer, second edition, 2004.

M. Rolf, J. J. Steil, and M. Gienger. Goal babbling permits direct learning of inverse kinematics. *IEEE Trans. Autonomous Mental Development*, 2(3):216 – 229, 2010.

S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Ed., 2003.

C. Salaun, V. Padois, and O. Sigaud. Learning forward models for the operational space control of redundant robots. In O. Sigaud and J. Peters, editors, *From Motor Learning to Interaction Learning in Robots*, volume 264, pages 169–192. Springer, 2010.

A. L. Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3):210–229, 1959.

S. Schaal, D. Sternad, and C. G. Atkeson. One-handed juggling: A dynamical approach to a rhythmic movement task. *Journal of Motor Behavior*, pages 165–183, 1996.

B. Schlkopf, A. J. Smola, and K. R. Müller. Kernel principal component analysis. *Advances in kernel methods: support vector learning*, pages 327–352, 1999.

B. Schölkopf and A. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond*. MIT-Press, Cambridge, MA, 2002.

B. Schölkopf, J. C. Platt, J. C. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computations*, 13:1443–1471, 2001.

L. Sciavicco and B. Siciliano. *Modelling and Control of Robot Manipulators (Advanced Textbooks in Control and Signal Processing)*. Advanced textbooks in control and signal processing. Springer, 2nd edition, 2005.

J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, New York, NY, USA, 2004.

B. Siciliano. Kinematic control of redundant robot manipulators: A tutorial. *Journal of Intelligent and Robotic Systems*, 3(3):201–212, 1990.

S. P. Singh. Transfer of learning by composing solutions of elemental sequential tasks. *Mach.*

Learn., 8(3-4):323–339, 1992.

B. Skinner. *About Behaviorism*. Knopf Doubleday Publishing Group, 2011.

V. Smidl and A. Quinn. On Bayesian principal component analysis. *Computational Statistics and Data Analysis*, 51(9):4101–4123, 2007.

A. J. Smola and P. Bartlett. Sparse greedy Gaussian process regression. In *Neural Information Processing Systems*, pages 619–625. MIT Press, 2001.

E. Snelson. Local and global sparse Gaussian process approximations. *Artificial Intelligence and Statistics*, 11, 2006.

E. Snelson and Z. Ghahramani. Sparse Gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems*, pages 1257–1264. MIT press, 2006.

J. A. Snyman. *Practical Mathematical Optimization: An Introduction to Basic Optimization Theory and Classical and New Gradient-Based Algorithms*, volume 97 of *Applied Optimization*. Springer-Verlag New York, 2005.

E. Sontag. *Mathematical Control Theory: Deterministic Finite Dimensional Systems*. Texts in Applied Mathematics. Springer, 1998.

M. Spong and M. Vidyasagar. *Robot Dynamics And Control*. Wiley India Pvt. Ltd., 2008.

L. Steels and M. Hild, editors. *Language Grounding in Robots*. Springer, New York, 2012.

G. Sun and B. Scassellati. Reaching through learned forward model. In *Proceedings of the IEEE-RAS/RSJ International Conference on Humanoid Robots*, Santa Monica, 2004.

R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 1998.

J. A. K. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural Processing Letters*, 9:293–300, 1999.

B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

M. E. Taylor and P. Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(1):1633–1685, 2009.

G. Tesauro. Temporal difference learning and TD-gammon. *Commun. ACM*, 38(3):58–68, 1995.

G. Tevatia and S. Schaal. Inverse kinematics for humanoid robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 294–299, 2000.

S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents series)*. Intelligent robotics and autonomous agents. The MIT Press, 2005.

S. B. Thrun and T. M. Mitchell. Lifelong robot learning. Technical Report IAI-TR-93-7, Robotics and Autonomous Systems, 1993.

M. E. Tipping. *Bayesian Inference: An Introduction to Principles and Practice in Machine Learning*, volume 3176, chapter 3, pages 41–62. Springer Berlin Heidelberg, 2004.

M. Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *the 12th International Conference on Artificial Intelligence and Statistics*, volume 5, 2009.

L. Trefethen and D. Bau. *Numerical linear algebra*. Miscellaneous Books. Society for Industrial and Applied Mathematics, 1997.

V. Tresp. A Bayesian committee machine. *Neural Comput.*, 12:2719–2741, 2000.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

V. Vapnik. *The Nature of Statistical Learning Theory*. Statistics for Engineering and Information Science. Springer, 1999.

S. Vijayakumar, A. D'Souza, and S. Schaal. Incremental online learning in high dimensions. *Neural Computation*, 17:2602–2634, 2005.

A. von Twickel, M. Hild, T. Siedel, V. Patel, and F. Pasemann. Neural control of a modular multi-legged walking machine: Simulation and hardware. *Robotics and Autonomous Systems*, 60(2):227 – 241, 2012.

G. Wahba. *Spline Models for Observational Data*. CBMS-NSF Regional Conference Series in Applied Mathematics. Society for Industrial and Applied Mathematics, 1990.

C. Wang and S. Mahadevan. Manifold alignment using Procrustes analysis. In *Proceedings of the 25th international conference on Machine learning*, pages 1120–1127. ACM New York, NY, USA, 2008.

J. Weston, O. Chapelle, A. Elisseeff, B. Schölkopf, and V. Vapnik. Kernel dependency estimation. In *Neural Information Processing Systems*, pages 873–880, 2002.

C. Williams and M. Seeger. Using the nyström method to speed up kernel machines. In *Neural Information Processing Systems*, pages 682–688. MIT Press, 2001.

R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. In *Machine Learning*, pages 229–256, 1992.