ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

Learning Control Under Uncertainty: A Probabilistic Value-Iteration Approach

B. Bischoff[1], D. Nguyen-Tuong[1], H. Markert[1] and A. Knoll[2]

1- Robert Bosch GmbH - Corporate Research
Robert-Bosch-Str. 2, 71701 Schwieberdingen - Germany

2- TU Munich - Robotics and Embedded Systems
Boltzmannstr. 3, 85748 Garching at Munich - Germany

Abstract. In this paper, we introduce a probabilistic version of the well-studied Value-Iteration approach, i.e. Probabilistic Value-Iteration (PVI). The PVI approach can handle continuous states and actions in an episodic Reinforcement Learning (RL) setting, while using Gaussian Processes to model the state uncertainties. We further show, how the approach can be efficiently realized making it suitable for learning with large data. The proposed PVI is evaluated on a benchmark problem, as well as on a real robot for learning a control task. A comparison of PVI with two state-of-the-art RL algorithms shows that the proposed approach is competitive in performance while being efficient in learning.

1 Introduction

Today, Reinforcement Learning (RL) has become an established method for policy learning and has found its way to various fields of application. Machine learning techniques, especially, RL can help to master problems which often occur in technical systems, such as uncertainties. In robot control, for example, it can be difficult to obtain an accurate, analytical controller due to increasing uncertainties in complex robot systems [1]. These uncertainties can be caused by complex mechanical components, as well as by intricate measurement systems.

While classical RL provides efficient methods for learning control policies [2, 3], including uncertainties into RL formalisms has found increasing interests in the last decade [4, 5]. A well-studied approach in the classical field of RL is the Value-Iteration approach [2, 3]. In [5], the authors provide a theoretical framework to include uncertainties into the Value-Iteration learning process with continuous states using Gaussian Processes (GP). However, their approach is not appropriate for real-world problems, as the system dynamics must be known beforehand and the support points for the Value-Iteration are fixed. Based on the idea introduced in [5], [6] presents an extended approach, where the dynamics is learned online and the support points can be flexibly set during iterations of the value function. In this work, they also show that probabilistic Value-Iteration can be employed to learn altitude control of a robotic blimp [6]. However, their approach requires that the action space is discretized, while the state space can be continuous. This is a major restriction for many control tasks, since the accuracy is limited here (due to discretized actions). Furthermore, prior knowledge is essential to obtain a reasonable discretization of the actions. In practice, the

ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

size of the discrete action sets grows exponentially with the number of action dimensions and, hence, making it inefficient for complex learning problems.

In this paper, we take a step forward and formulate a probabilistic Value-Iteration approach (PVI), where the action and state spaces can be both continuous. Thus, the optimal action can now be found efficiently using gradient-based search, where analytical gradients for the PVI are provided. Furthermore, the dynamics model can be learned online and the support points of the value function are selected using a kernel-based criterion. We evaluate our PVI algorithm on the simulated mountain-car benchmark problem first. In more advanced experiments, we test the algorithm on a simulated robot, as well as on the mobile robot Festo Robotino, see Figure 4. Here, the control task for the mobile robot is moving to a desired goal position from a given starting point. We also compare our PVI approach with two other RL algorithms, e.g. Pilco [4] and Neural fitted Q-Iteration (NFQ) [7]. The remainder of the paper will be organized as follows: first, we provide a brief review on classical RL. In Section 3, we describe our PVI algorithm in details. Section 4 provides evaluations and robot experiments using the proposed method. A conclusion will be given in Section 5.

2 Reinforcement Learning and Value-Iteration

In this section, we provide a short overview on the basic concepts of RL with focus on Value-Iteration [2, 3]. RL describes the concept of goal directed learning of an agent by interacting with a system. The state of the agent is given as $s \in S$, the agent can apply actions $a \in A$ to move to new states $s' \in S$. The transition probability is given by $p(s, a, s')$ representing the *dynamics*. A policy $\pi : S \to A$ determines in every state the action to be used. The user-defined reward function $R : S \times A \times S \to \mathbb{R}$ rates transitions and, thus, encodes the desired goal. The latent long term reward for a given policy π is the sum of expected rewards, and can be written as given in equation (1). Here, V_π denotes the *value function* of π, γ is a discount factor, with $0 < \gamma < 1$. The goal of RL is to obtain a policy π^* that maximizes the long term reward, i.e. $V_{\pi^*}(s) = \max_{\pi \in \Pi} \{V_\pi(s)\}$ for all states $s \in S$. The optimal policy π^* can be obtained from V_{π^*} by solving equation (2). A well-studied approach to estimate V_{π^*} is *Value-Iteration* [2]. Here, the functions V^k are calculated iteratively using equation (3), which is guaranteed to converge to V_{π^*} under some mild assumptions [8].

$$V_\pi(s) = \sum_{s' \in S} p(s, \pi(s), s') \left[R(s, \pi(s), s') + \gamma V_\pi(s') \right] \tag{1}$$

$$\pi^*(s) = \arg\max_{a \in A} \sum_{s' \in S} p(s, a, s') \left[R(s, a, s') + \gamma V_{\pi^*}(s') \right] \tag{2}$$

$$\forall s \in S : V^{k+1}(s) = \max_{a \in A} \sum_{s' \in S} p(s, a, s') \left[R(s, a, s') + \gamma V^k(s') \right] \tag{3}$$

However, the suitability of Value-Iteration for practical control tasks, e.g. in robotics, is limited. Classical Value-Iteration considers in most cases only dis-

ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

crete states and actions. Rasmussen and Kuss [5] proposed an approach to generalize Value-Iteration for continuous state spaces. Furthermore, they extend Value-Iteration learning for uncertain cases, while using GP to model the value function and dynamics. However, their value function needs to be estimated for a fixed set of support points. Following the work by [5], Rottmann proposed a further extension, where support points and dynamics can be learned [6]. However, the action space is still discretized, which implies some disadvantages, such as prior knowledge for action discretization, limitation in control accuracy, and exponential grow of actions with the number of dimensions. In this work, we present a further development for Value-Iteration allowing continuous actions.

3 Probabilistic Value-Iteration (PVI)

Value-Iteration can be extended to continuous states and actions by replacing the sums in equations (1)–(3) with integrals, see e.g. (4). To model the uncertainty in the dynamics and states, we employ GP, as previously proposed in [5]. Given the dynamics model, the distribution of the next state $p(s')$ given action a is estimated as Gaussian distribution with mean $\mu_{s'}$ and covariance $\Sigma_{s'}$. For a given set of support points $s \in S_{sp}$, the value function $V^k(s)$ can be learned, as given in equation (5). A GP is employed to generalize from the values $\left\{V^k(s) \,|\, s \in S_{sp}\right\}$ of the support points to the entire state space.

$$\forall s \in S_{sp} : V^{k+1}(s) = \max_{a \in A} \int p(s,a,s') \left(R(s,a,s') + V^k(s')\right) ds' \tag{4}$$

$$\forall s \in S_{sp} : V^{k+1}(s) = \max_{a \in A} \left(R(s,a,s' \,|\, \mu_{s'}, \Sigma_{s'}) + \gamma V^k(s' \mid \mu_{s'}, \Sigma_{s'})\right) \tag{5}$$

$$\pi(s) = \arg\max_{a \in A} \left(R(s,a,s' \,|\, \mu_{s'}, \Sigma_{s'}) + \gamma V(s' \mid \mu_{s'}, \Sigma_{s'})\right) . \tag{6}$$

To determine S_{sp}, we generate points by policy application and thereof select the set of support points incrementally and online while employing a kernel-based criterion [1]. Here, the basic idea is to select points being "far away" from a linear span in the feature space defined by the support set. Thus, given the support set $S_{sp} = \{s_i\}_{i=1}^n$, the measure $\delta(s_{new}) = \| \sum_{i=1}^n a_i \phi(s_i) - \phi(s_{new}) \|$ is computed, where ϕ is the feature vector and a_i represent the linear coefficients. The new point s_{new} is included into the support set, when δ is larger than a given threshold value. The measure δ can be written in terms of kernels incorporating the feature representation ϕ. More details can be found in [1].

To sample the dynamics in the relevant state space region, we employ episodic RL. First, we sample dynamics data using the learned policy. Subsequently, we learn the dynamics model based on sampled data and improve the policy based on the learned dynamics model. These steps are iterated till convergence. The basic concept of PVI is showed in Algorithm 1. As the action space is continuous, we solve equation (6) using gradient-based search for finding an optimal policy π. In the following, we provide the required analytical gradients. Due to space limitations, we only give the gradients of the value-function $\partial V(s')/\partial a$, i.e.

$$\frac{\partial V(s'|\mu_{s'}, \Sigma_{s'})}{\partial a^{(i)}} = \frac{\partial(\lambda q^T \beta)}{\partial a^{(i)}} = \frac{\partial \lambda}{\partial a^{(i)}} q^T \beta + \lambda \frac{\partial q^T}{\partial a^{(i)}} \beta$$

ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0. Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

Algorithm 1 Probabilistic Value-Iteration (PVI)

1: Apply random actions on system to obtain initial dynamics data D
2: **for** $e = 1$ **to** *Episodes* **do**
3: Train GP with D to estimate system dynamics p
4: **repeat**
5: Generate set of support points S_{sp}
6: $\forall s \in S_{sp} : V^{k+1}(s) = \max_{a \in A} \left(R(s, a, s' \mid \mu_{s'}, \Sigma_{s'}) + \gamma V^k(s' \mid \mu_{s'}, \Sigma_{s'}) \right)$
7: Train GP V^{k+1} with targets $V^{k+1}(s)$
8: **until** convergence
9: Apply current policy π on system, increment D with new dynamics data
10: **end for**

where we employ the parametrization $V(s'|\mu_{s'}, \Sigma_{s'}) = \lambda q^T \beta$, as proposed in [9]. Here, $q = \exp(-\frac{1}{2}(\mu_{s'} - S_{sp})^T(\Lambda_V + \Sigma_{s'})^{-1}(\mu_{s'} - S_{sp}))$, $\beta = K(S_{sp}, S_{sp})V(S_{sp})$ and $\lambda = |\Lambda_V^{-1}\Sigma_{s'} + I|^{-\frac{1}{2}}$. The other gradient $\partial R(s, a, s')/\partial a$, see equation (6), can be computed accordingly and will be provided in a longer version of the paper. The derivative $\partial V(s')/\partial a$ is given for the squared exponential kernel, i.e. $k_{se}(x, x') = \sigma_f^2 \exp\left(-\frac{1}{2}(x - x')^T \Lambda^{-1}(x - x')\right)$, resulting in

$$\frac{\partial \lambda}{\partial a^{(i)}} = -\frac{\lambda}{2} Tr\left[(\Lambda_V^{-1}\Sigma_{s'} + I)^{-1}\Lambda_V^{-1}\frac{\partial \Sigma_{s'}}{\partial a^{(i)}}\right]; \frac{\partial q}{\partial a^{(i)}} = q\left(-A^T\frac{\partial \mu_{s'}}{\partial a^{(i)}} + \frac{1}{2}A^T\frac{\partial \Sigma_{s'}}{\partial a^{(i)}}A\right)$$

with $A = (\Lambda_V + \Sigma_{s'})^{-1}(\mu_{s'} - S_{sp})$ and $\partial\Sigma_{s'}/\partial a^{(i)}$, $\partial\mu_{s'}/\partial a^{(i)}$ are given as

$$\frac{\partial \mu_{s'}}{\partial a^{(i)}} = \frac{\partial k([s,a], X_{\text{dyn}})}{\partial a^{(i)}} K_{\text{dyn}} y_{\text{dyn}}\ ;\ \frac{\partial k([s,a], X_{\text{dyn}})}{\partial a^{(i)}} = \exp\left(-\frac{1}{2}CB\right)(-C\alpha_i)$$

$$\frac{\partial \Sigma_{s'}}{\partial a^{(i)}} = \frac{\partial k([s,a], X_{\text{dyn}})}{\partial a^{(i)}} K_{\text{dyn}} k(X_{\text{dyn}}, [s,a]) + k([s,a], X_{\text{dyn}}) K_{\text{dyn}} \frac{\partial k(X_{\text{dyn}}, [s,a])}{\partial a^{(i)}}$$

with $B = ([s,a] - X_{\text{dyn}})$, $C = B^T\Lambda_{\text{dyn}}^{-1}$ and α_i equals 1 at the entry corresponding to action-dimension i and 0 else. Further, Λ_{dyn} and Λ_V are hyper-parameters of the dynamics respectively value-function GP, X_{dyn}, y_{dyn} are sampled dynamics data.

4 Evaluation

In this section, we evaluate PVI in simulation, as well as on a mobile robot, e.g. the Robotino in Figure 4. We compare the results of PVI to Pilco [4] and NFQ [7]. For all evaluations, we use a saturated immediate reward (see [4]) defined by $R(s) = \exp\left(-\frac{1}{2c^2}|s - s_{\text{goal}}|\right)$ with Euclidean norm $|\cdot|$.

First, we compare the algorithms on the well-known mountain-car task. The state space consists of positions and velocities of the car. The starting position is in the valley, the goal is to drive uphill and stay at the slope. As can be seen in Figure 1, PVI, Pilco and NFQ all converge to a similar solution (which is nearly optimal). However, PVI and Pilco are more efficient and find the solution already after 2 respectively 4 episodes.

ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

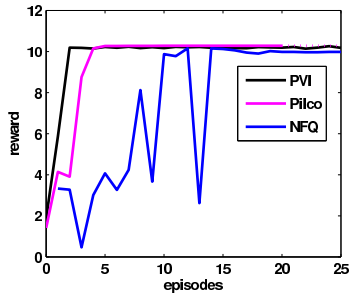

Figure 1: Comparison of PVI, Pilco and NFQ on the mountain-car task.

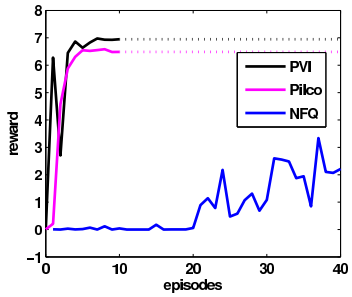

Figure 2: Comparison of PVI, Pilco and NFQ on a 1m robot translation task simulated in Gazebo.

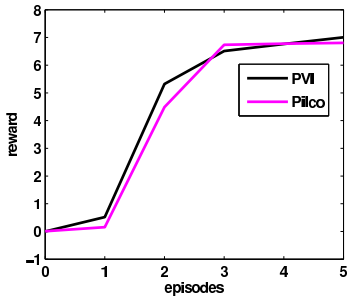

Figure 3: Learning results for translation tasks on the Robotino, exemplary 1m forward translation.

Task	dist. error in cm at end position	
	PVI	**Pilco**
1m forward	0.5	1.7
1m backward	1.1	1.48
1m right	2.6	2.97
1m left	0.8	2.28

Table 1: Robotino translation task, accuracy at the end position for all four directions.

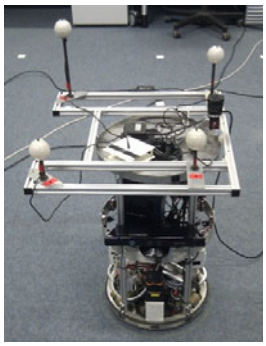

Figure 4: Robot platform Festo Robotino used for evaluation.

Before we employ the learning algorithms on the Robotino, we first compare the algorithms in a robot simulation. Here, the robot is modelled to be similar to the real Robotino. The robot state $s \in \mathbb{R}^3$ corresponds to x, y-position and yaw θ of the mobile robot. The task is to control the rotational speed of the wheels to achieve 1m forward translation. For all learning and evaluation, the cost width was set to $c = 0.25$. Figure 2 shows, that PVI and Pilco converge successfully to policies after few learning episodes. It turns out that PVI gains slightly more reward than Pilco, as Pilco's policy does not employ the full engine power to accelerate the robot. NFQ is able to drive the robot near the goal, but not accurately enough to gain significant reward.

Finally, we evaluate PVI on the real Robotino for a translation task in 4 directions. The task settings are similar to the simulation, the rotational speed, i.e. the action $a \in \mathbb{R}^3$, of the robot wheels need to be adjusted. For real robot experiments, we compare our PVI with Pilco. The resulting reward over 5 episodes is shown in Figure 3. After only 3 episodes, PVI and Pilco converge to successful policies. In these experiments, the policies are learned based on 30 points of dynamics data, i.e. 30 tuples s, a, s'. For PVI, the final set of support

ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

points to estimate the value function included 239 points, choosen by the kernel-based criterion with a threshold of 0.025. While Figure 3 exemplary shows the reward for 1m forward-translation, the learning results for backward, left and right translation are similar. The resulting accuracy of the final robot position is shown in Table 1. As can be seen, all translation tasks can be solved with high accuracy in the final position.

5 Conclusion

For complex systems, it is difficult to obtain accurate analytical controllers due to unknown nonlinearities, as well as increasing uncertainty. While RL provides efficient methods for learning control policy, incorporating uncertainties into RL formalisms is still a challenging issue. Inspired by the work [5], we presented an extended version of the well-known Value-Iteration, i.e. PVI, while including uncertainty into the learning process. The proposed approach is appropriate for RL learning with continuous states and actions, where the uncertainties are taking into account by using GP for modelling states and actions. Using PVI, the dynamics and support points of the value function can be learned from sampled data. As PVI is formulated for continuous states and actions, efficient gradient-search can be employed to solve the optimization problem for finding the optimal action. The approach is evaluated on the mountain-car benchmark problem, as well as on a real robot control task. A comparison with the state-of-the-art algorithms, e.g. Pilco and NFQ, shows that our approach is competitive in performance while efficient in learning.

References

[1] Duy Nguyen-Tuong and Jan Peters. Incremental online sparsification for model learning in real-time robot control. *Neurocomputing*, 74(11):1859–1867, 2011.

[2] Richard S. Sutton and Andrew G. Barto. Reinforcement learning: An introduction. *IEEE Transactions on Neural Networks*, 9(5):1054–1054, 1998.

[3] M. Wiering and M. van Otterlo. *Reinforcement Learning: State-of-the-Art*. Adaptation, Learning, and Optimization. Springer, 2012.

[4] Marc Peter Deisenroth and Carl Edward Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *ICML*, pages 465–472, 2011.

[5] Carl Edward Rasmussen and Malte Kuss. Gaussian processes in reinforcement learning. In *NIPS*, 2003.

[6] Axel Rottmann and Wolfram Burgard. Adaptive autonomous control using online value iteration with gaussian processes. In *ICRA*, pages 2106–2111, 2009.

[7] Martin Riedmiller. Neural fitted q iteration - first experiences with a data efficient neural reinforcement learning method. In *ECML*, pages 317–328, 2005.

[8] Eugenio Della Vecchia, Silvia Di Marco, and Alain Jean-Marie. Illustrated review of convergence conditions of the value iteration algorithm and the rolling horizon procedure for average-cost mdps. *Annals OR*, 199:193–214, 2012.

[9] A. Girard, J. Q. Candela, R. Murray-Smith, and C. E. Rasmussen. Gaussian Process Priors with Uncertain Inputs - Application to Multiple-Step Ahead Time Series Forecasting. 2003.