Delft University of Technology

Delft Center for Systems and Control

Technical report 08-005

Consistency of fuzzy model-based reinforcement learning*

L. Buşoniu, D. Ernst, B. De Schutter, and R. Babuška

If you want to cite this report, please use the following reference instead:
L. Buşoniu, D. Ernst, B. De Schutter, and R. Babuška, "Consistency of fuzzy model-based reinforcement learning," *Proceedings of the 2008 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2008)*, Hong Kong, pp. 518–524, June 2008.

Delft Center for Systems and Control
Delft University of Technology
Mekelweg 2, 2628 CD Delft
The Netherlands
phone: +31-15-278.24.73 (secretary)
URL: `https://www.dcsc.tudelft.nl`

*This report can also be downloaded via `https://pub.deschutter.info/abs/08_005.html`

Consistency of Fuzzy Model-Based Reinforcement Learning

Lucian Buşoniu, Damien Ernst, Bart De Schutter, and Robert Babuška

Abstract—**Reinforcement learning (RL) is a widely used paradigm for learning control. Computing exact RL solutions is generally only possible when process states and control actions take values in a small discrete set. In practice, approximate algorithms are necessary. In this paper, we propose an approximate, model-based Q-iteration algorithm that relies on a fuzzy partition of the state space, and on a discretization of the action space. Using assumptions on the continuity of the dynamics and of the reward function, we show that the resulting algorithm is consistent, i.e., that the optimal solution is obtained asymptotically as the approximation accuracy increases. An experimental study indicates that a continuous reward function is also important for a predictable improvement in performance as the approximation accuracy increases.**

I. Introduction

REINFORCEMENT learning (RL) is a popular paradigm for learning control, thanks to its mild assumptions on the process (which can be nonlinear and stochastic), and because it can work without an explicit model of the process [1], [2]. A RL controller receives a scalar reward signal as feedback on its immediate performance, and has to maximize the cumulative reward obtained in the long run. Most RL algorithms work by estimating value functions, i.e., cumulative rewards as a function of the process state and possibly of the control action. In general, the classical RL algorithms only work when the state-action space of the problem has a finite (and not too large) number of elements. Therefore, approximate algorithms are necessary in practice, where state-action spaces are usually large or continuous. Two desirable properties of approximate algorithms are convergence to a near-optimal solution and consistency, which in a model-based setting means asymptotic convergence to the optimal solution as the approximation accuracy increases.

Fuzzy Q-iteration [3], [4] is a model-based RL algorithm that represents value functions using a fuzzy partition of the state space, and requires a discrete action space. The approximate value function is a linear combination of the parameters, where the weights are the membership values. Fuzzy Q-iteration was shown in [3], [4] to converge to an approximate value function that lies within a bound from the optimal value function. The practical performance of the algorithm was illustrated on problems with four state variables and two control inputs. Fuzzy Q-iteration was also compared in [4] with Q-iteration with radial basis-function approximation.

In this paper, we study the consistency of fuzzy Q-iteration. Under appropriate assumptions on the process dynamics and on the reward function, it is proven that the approximate solution of fuzzy Q-iteration asymptotically converges to the optimal solution, as the maximum distance between the cores of adjacent fuzzy sets, and the maximum distance between adjacent discrete actions, decrease to 0. A discretization procedure is used to approximate the continuous (or discrete but large) action space. In contrast, the original fuzzy Q-iteration required a small, discrete action space from the outset. The convergence properties of the original fuzzy Q-iteration are not affected by this discretization procedure. Additionally, the influence of discontinuities in the reward function is investigated in a numerical example involving a second-order servo-system.

A number of related fuzzy approximators have been proposed, mostly for model-free RL algorithms such as Q-learning [5]–[7] and actor-critic [8]–[11]. Most of these approaches are heuristic in nature, and their theoretical properties have not been investigated. Notable exceptions are the provably convergent actor-critic algorithms in [9], [10]. In this paper, we use fuzzy approximation with a *model-based* algorithm, and provide a theoretical and empirical analysis of its consistency properties.

A rich body of literature concerns the theoretical analysis of approximate RL algorithms, both in a model-based setting [12]–[17] and in a model-free setting [18]–[22]. However, our consistency analysis for fuzzy Q-iteration is new. Many consistency results for model-based RL can be found for discretization-based approximators [12], [15]. Such discretizations sometimes employ interpolation schemes similar to the approximation formulas used by fuzzy Q-iteration. In model-free RL , consistency is usually understood as the convergence to a well-defined solution as the number of samples increases [16]. Convergence to an optimal solution as the approximation accuracy also increases is proven in [18], [20].

The rest of this paper is structured as follows. Section II introduces the RL problem and reviews some relevant results from dynamic programming. Section III presents fuzzy Q-iteration and recalls its convergence properties. The consistency of fuzzy Q-iteration is proven in Section IV. Section V uses an example to illustrate the impact of a discontinuous reward function on the performance of fuzzy Q-iteration. Section VI outlines several ideas for future work and concludes the paper.

Lucian Buşoniu, Bart De Schutter, and Robert Babuška are with the Delft Center for Systems and Control of the Delft University of Technology, The Netherlands (email: i.l.busoniu@tudelft.nl, b@deschutter.info, r.babuska@tudelft.nl). Bart De Schutter is also with the Marine and Transport Technology Department of the Delft University of Technology.

Damien Ernst is a Research Associate of the Belgian FNRS; he is affiliated with the Systems and Modeling Unit of the University of Liège, Belgium (email: dernst@ulg.ac.be).

II. Reinforcement Learning

In this section, the RL task is briefly introduced and its optimal solution is characterized [1], [2].

Consider a deterministic *Markov decision process (MDP)* with the state space X, the action space U, the transition function $f : X \times U \to X$, and the reward function $\rho : X \times U \to \mathbb{R}$.[1] As a result of the control action u_k in the state x_k at the discrete time step k, the state changes to $x_{k+1} = f(x_k, u_k)$. At the same time, the controller receives the scalar reward signal $r_{k+1} = \rho(x_k, u_k)$, which evaluates the immediate effect of action u_k, but says nothing about its long-term effects. For simplicity, we consider in this paper only the deterministic case. A stochastic formulation is also possible; in that case, expected returns under the probabilistic transitions must be considered.

The controller chooses actions according to its policy $h : X \to U$, using $u_k = h(x_k)$. The goal of the controller is to learn a policy that maximizes, starting from the current moment in time ($k = 0$) and from any initial state x_0, the discounted return:

$$R = \sum_{k=0}^{\infty} \gamma^k r_{k+1} = \sum_{k=0}^{\infty} \gamma^k \rho(x_k, u_k) \tag{1}$$

where $\gamma \in [0, 1)$ and $x_{k+1} = f(x_k, u_k)$ for $k \geq 0$. The discounted return compactly represents the reward accumulated by the controller in the long run. The learning task is therefore to maximize the long-term performance, while only using feedback about the immediate, one-step performance. One way to achieve this is by computing the optimal action-value function.

An action-value function (Q-function), $Q^h : X \times U \to \mathbb{R}$, gives the return of each state-action pair under a policy h:

$$Q^h(x, u) = \rho(x, u) + \sum_{k=1}^{\infty} \gamma^k \rho(x_k, h(x_k)) \tag{2}$$

where $x_1 = f(x, u)$ and $x_{k+1} = f(x_k, h(x_k))$ for $k \geq 1$. The optimal action-value function is defined as $Q^*(x, u) = \max_h Q^h(x, u)$. Any policy that selects for every state an action with the highest optimal Q-value: $h^*(x) = \arg\max_u Q^*(x, u)$, is optimal (it maximizes the return).

A central result in RL , upon which many algorithms rely, is the *Bellman optimality equation*:

$$Q^*(x, u) = \rho(x, u) + \gamma \max_{u' \in U} Q^*(f(x, u), u') \tag{3}$$

This equation can be solved using the Q-value iteration algorithm. Let the set of all Q-functions be denoted by $\mathcal{Q}$. Define the Q-iteration mapping $T : \mathcal{Q} \to \mathcal{Q}$, which computes the right-hand side of the Bellman equation for any Q-function:

$$[T(Q)](x, u) = \rho(x, u) + \gamma \max_{u' \in U} Q(f(x, u), u') \tag{4}$$

[1]Throughout the paper, the standard control-theoretic notation is used: x for state, X for state space, u for control action, U for action space, f for process (environment) dynamics. We denote reward functions by ρ, to distinguish them from the instantaneous rewards r and the return R. We denote policies by h.

Using this notation, the Bellman equation (3) states that Q^* is a fixed point of T, i.e., $Q^* = T(Q^*)$. It is well-known that T is a contraction with factor $\gamma < 1$ in the infinity norm, i.e., for any pair of functions Q and Q', it is true that $\|T(Q) - T(Q')\|_\infty \leq \gamma \|Q - Q'\|_\infty$.

The Q-value iteration (Q-iteration, for short) algorithm uses an *a priori* model of the task, in the form of the transition and reward functions f, ρ. The algorithm starts from an arbitrary Q-function Q_0 and in each iteration ℓ updates the Q-function using the formula $Q_{\ell+1} = T(Q_\ell)$. Because T is a contraction, it has a unique fixed point. From (3), this point is Q^*, so Q-iteration converges to Q^* as $\ell \to \infty$.

III. Fuzzy Q-iteration

In this section, the approximate, fuzzy Q-iteration algorithm is introduced. The state and action spaces of the MDP may be either continuous or discrete, but they are assumed to be subsets of Euclidean spaces, such that the 2-norm of the states and actions is well-defined. For the simplicity of notation, X and U are also assumed to be closed sets.

Fuzzy Q-iteration was introduced in [3], [4] for problems with continuous (or discrete but large) state spaces and small, discrete action spaces. Here, the algorithm is extended to continuous (or discrete but large) action spaces using an explicit action discretization procedure.

The proposed approximation scheme uses a fuzzy partition of the state space. The partition contains N fuzzy sets, each described by a membership function (MF) $\varphi_i : X \to [0, 1]$, $i = 1, \ldots, N$. A state x belongs to each set i with a degree of membership $\varphi_i(x)$. In the sequel, the following requirements are imposed.

Requirement 1 (Normalized partition): The fuzzy partition has been normalized, i.e., $\sum_{i=1}^{N} \varphi_i(x) = 1$, $\forall x \in X$.

Requirement 2 (Normal fuzzy sets): All the fuzzy sets in the partition are normal and have singleton cores, i.e., for every i there exists a unique x_i for which $\varphi_i(x_i) = 1$ (consequently, $\varphi_{i'}(x_i) = 0$ for all $i' \neq i$ by Requirement 1). The state x_i is called the core (center value) of the i-th set.

Requirement 1 is not restrictive, because any fuzzy partition can be normalized as long as for any x, there is some i such that $\varphi_i(x) > 0$. Requirement 2 is imposed here for brevity in the description and analysis of the algorithms; it can be relaxed using results of [14].

Example 1: Triangular fuzzy partitions. A simple type of fuzzy partition that satisfies Requirements 1 and 2 can be obtained as follows. For each state variable x_d with $d = 1, \ldots, D$, a number N_d of triangular MF s are defined as follows:

$$\varphi_{d,1}(x_d) = \max\left(0, \frac{c_{d,2} - x_d}{c_{d,2} - c_{d,1}}\right)$$

$$\varphi_{d,i}(x_d) = \max\left[0, \min\left(\frac{x_d - c_{d,i-1}}{c_{d,i} - c_{d,i-1}}, \frac{c_{d,i+1} - x_d}{c_{d,i+1} - c_{d,i}}\right)\right]$$

$$\text{for } i = 2, \ldots, N_d - 1$$

$$\varphi_{d,N_d}(x_d) = \max\left(0, \frac{x_d - c_{d,N_d-1}}{c_{d,N_d} - c_{d,N_d-1}}\right)$$

where $c_{d,1} < \ldots < c_{d,N_d}$ is the array of cores along dimension d, which completely determines the shape of the MF s, and $x_d \in [c_{d,1}, c_{d,N_d}]$. Adjacent MF s always intersect at a 0.5 level. Then, the product of each combination of (single-dimensional) MF s yields a pyramidal-shaped D-dimensional MF in the fuzzy partition of X. □

A discrete set of actions U_0 is chosen from the larger action space:

$$U_0 = \{u_j | u_j \in U, j = 1, \ldots, M\} \tag{5}$$

The fuzzy approximator stores an $N \times M$ matrix of parameters. Each pair (φ_i, u_j) of a MF and a discrete action is associated to one parameter $\theta_{i,j}$.

Fuzzy Q-iteration uses the classical Q-value iteration mapping T (4), together with an approximation mapping and a projection mapping. The approximation mapping F takes as input a parameter matrix θ and outputs an approximate Q-function. For every state-action pair (x, u), this approximate Q-function is computed as follows:

$$\begin{aligned} \widehat{Q}(x,u) = [F(\theta)](x,u) = \sum_{i=1}^{N} \varphi_i(x)\theta_{i,j} \\ \text{with } j = \underset{j'=1,\ldots,M}{\arg\min} \|u - u_{j'}\|_2 \end{aligned} \tag{6}$$

This is a linear basis functions form. To ensure that $F(\theta)$ is a well-defined function for any θ, the ties in the $\arg\min$ have to be broken consistently (e.g., always in favor of the smallest index that satisfies the condition). For a fixed x, the approximator is constant over the set of actions $\{u \in U \mid \|u - u_j\|_2 < \|u - u_{j'}\|_2 \ \forall j' \neq j\}$. This set is the interior of the Voronoi cell for u_j (a convex polytope).

The projection mapping infers from a Q-function the values of the approximator parameters according to the relation:

$$\theta_{i,j} = [P(Q)]_{i,j} = Q(x_i, u_j) \tag{7}$$

This is the solution θ to the problem:

$$\sum_{i=1,\ldots,N, j=1,\ldots,M} |[F(\theta)](x_i, u_j) - Q(x_i, u_j)|^2 = 0$$

The (synchronous) *fuzzy Q-iteration* algorithm starts with an arbitrary θ_0, and approximately computes the Q-iteration mapping. This is done using the composition of the mappings P, T, and F:

$$\theta_{\ell+1} = PTF(\theta_\ell) \tag{8}$$

This composite mapping is applied iteratively until θ has converged. The convergence criterion is usually approximate: $\max_{i,j} |\theta_{\ell+1,i,j} - \theta_{\ell,i,j}| \leq \varepsilon_{\text{QI}}$. Then, an approximately optimal parameter matrix is $\widehat{\theta}^* = \theta_{\ell+1}$, and an approximately optimal policy can be computed with:

$$\widehat{h}^*(x) = u_{j^*}, \quad j^* = \arg\max_j [F(\widehat{\theta}^*)](x, u_j) \tag{9}$$

Of course, because the approximate Q-function is constant inside every Voronoi cell, any action in the j^*-th cell could be used, but because the algorithm estimates the Q-values $Q^*(x_i, u_j)$, there is intuitively greater confidence that u_{j^*} is optimal, rather than another action in the cell.

An asynchronous version of the algorithm can be given that makes more efficient use of the updates, by using the latest updated values of the parameters θ in each step of the computation [3], [4].

Because all the approximate Q-functions considered by fuzzy Q-iteration are constant inside every Voronoi cell, it suffices to consider only the discrete actions when computing the maximal Q-values in the Q-iteration mapping. This discrete-action version of the Q-iteration mapping is defined as follows:

$$[T_0(Q)](x,u) = \rho(x,u) + \gamma \max_{j=1,\ldots,M} Q(f(x,u), u_j) \tag{10}$$

This result is very useful in the practical implementation of fuzzy Q-iteration. Namely, PTF can be implemented as PT_0F, using the fact that all the Q-functions that are considered by the fuzzy Q-iteration algorithm are of the form $F(\theta)$. The maximization over U in the original T mapping can be replaced with a maximization over the discrete set U_0 (5), which can be solved using enumeration for moderate M. Furthermore, no distances in U need to be computed to implement $T_0F(\theta)$.

The following results are true for the original fuzzy Q-iteration in [3], [4], and can be easily extended to account for the action discretization.

Proposition 1: The following statements are true about fuzzy Q-iteration:

1.1. *(Convergence)* Fuzzy Q-iteration converges to a unique, optimal parameter matrix θ^* (both in its synchronous and asynchronous versions).

1.2. *(Convergence speed)* Asynchronous fuzzy Q-iteration converges at least as fast as synchronous fuzzy Q-iteration.

1.3. *(Suboptimality)* Denote by $\mathcal{F}_{FP} \subset \mathcal{Q}$ the set of fixed points of the mapping FP, and define $\varepsilon = \min_{Q' \in \mathcal{F}_{FP}} \|Q^* - Q'\|_\infty$, the minimum distance between Q^* and any fixed point of FP. The convergence point θ^* satisfies:

$$\|Q^* - F(\theta^*)\|_\infty \leq \frac{2\varepsilon}{1-\gamma} \tag{11}$$

The proofs rely on the fact that because P and F are non-expansions, the composite mapping PTF is a contraction with factor $\gamma < 1$ and with the unique fixed point θ^*.

IV. Consistency Analysis

This section gives our main result: the consistency of (synchronous and asynchronous) fuzzy Q-iteration is established, i.e., it is shown that the approximate solution $F(\theta^*)$ converges to the optimal Q-function Q^*, asymptotically as the maximum distance between the cores of adjacent fuzzy sets, and the maximum distance between adjacent discrete actions, decrease to 0.

The state resolution step δ_x is defined as the largest distance between any point in the state space and the core that is closest

to it. The action resolution step δ_u is defined similarly for the discrete actions. Formally:

$$\delta_x = \max_{x \in X} \min_{i=1,\ldots,N} \|x - x_i\|_2 \tag{12}$$

$$\delta_u = \max_{u \in U} \min_{j=1,\ldots,M} \|u - u_j\|_2 \tag{13}$$

where x_i is the core of the i-th MF , and u_j is the j-th discrete action. Smaller values of δ_x and δ_u indicate a higher resolution. The goal is to show that $\lim_{\delta_x \to 0,\ \delta_u \to 0} F(\theta^*) = Q^*$. First, we require that any state value only activates membership functions locally, in the following sense.

Requirement 3 (Local MF s): There exists a finite $\nu > 0$ such that, regardless of N, the MF s satisfy:

$$\max_{x \in X} \sum_{i=1}^{N} \varphi_i(x) \|x - x_i\|_2 \leq \nu \delta_x$$

This requirement is satisfied in many cases of interest. For instance, it is satisfied by convex fuzzy sets with their cores distributed on an (equidistant or irregular) rectangular grid in the state space, such as the triangular partitions of Example 1. In such cases, every point $x \in X$ falls inside a hyperbox defined by the two adjacent cores that are closest to x_d on each axis d. Some points will fall on the boundary of several hyperboxes, in which case we can just pick any of these hyperboxes. Given Requirement 2 and because the fuzzy sets are convex, only the MF s with the cores in the corners of the hyperbox can take non-zero values in the chosen point. The number of corners is 2^D where D is the dimension of X. By the definition of δ_x, the largest distance between the chosen point and any of the hyperbox corners is at most $2\delta_x$. Therefore, we have:

$$\max_{x \in X} \sum_{i=1}^{N} \varphi_i(x) \|x - x_i\|_2 \leq 2^D \, 2 \, \delta_x$$

and a choice of $\nu = 2^{D+1}$ indeed satisfies Requirement 3.

Assumption 1: The dynamics f and the reward function ρ satisfy are Lipschitz continuous, i.e., there exist finite constants $L_f > 0$ and respectively $L_\rho > 0$ such that:

$$\begin{aligned} \|f(x,u) - f(\bar{x},\bar{u})\|_2 &\leq L_f(\|x - \bar{x}\|_2 + \|u - \bar{u}\|_2) \\ |\rho(x,u) - \rho(\bar{x},\bar{u})| &\leq L_\rho(\|x - \bar{x}\|_2 + \|u - \bar{u}\|_2) \\ &\forall x, \bar{x} \in X, u, \bar{u} \in U \end{aligned}$$

Furthermore, the Lipschitz constant of f satisfies: $L_f < 1/\gamma$.

The Lipschitz continuity of f and ρ is typically needed to prove consistency of approximate RL algorithms. The bound on L_f is more restrictive. For RL problems resulting from the time discretization of continuous-time systems with Lipschitz continuous dynamics, this bound can be ensured by discretizing the continuous-time system with a sufficiently small sampling time.[2]

As a first step to proving consistency, the Lipschitz continuity of Q^* is proven. This will be used later on to show that one of the fixed points of FP can be made arbitrarily close to Q^* by increasing the state and action resolutions of the approximator (decreasing δ_x and δ_u). Consistency will then follow from (11).

Lemma 1 (Lipschitz continuity of Q^*): Under Assumption 1, the optimal Q-function is Lipschitz continuous with the Lipschitz constant $L_Q = \frac{L_\rho}{1 - \gamma L_f}$:

$$\begin{aligned} |Q^*(x,u) - Q^*(\bar{x},\bar{u})| &\leq L_Q(\|x - \bar{x}\|_2 + \|u - \bar{u}\|_2) \\ &\forall x, \bar{x} \in X, u, \bar{u} \in U \end{aligned}$$

Proof: Define the series $\{Q_\ell\}_{\ell \geq 0}$, as follows: $Q_0 = \rho$; $Q_{\ell+1} = T(Q_\ell)$, $\ell \geq 0$. It is well known that $\lim_{\ell \to \infty} Q_\ell = Q^*$ [1]. We show by induction that Q_ℓ is Lipschitz with the Lipschitz constant $L_{Q_\ell} = L_\rho \sum_{k=0}^{\ell} \gamma^k {L_f}^k$. Indeed, $L_{Q_0} = L_\rho$ because $Q_0 = \rho$, and:

$$\begin{aligned} &|[T(Q_\ell)](x,u) - [T(Q_\ell)](\bar{x},\bar{u})| \\ &\quad = \Big|\rho(x,u) + \gamma \max_{u'} Q_\ell(f(x,u),u') - \\ &\qquad \rho(\bar{x},\bar{u}) - \gamma \max_{\bar{u}'} Q_\ell(f(\bar{x},\bar{u}),\bar{u}')\Big| \\ &\quad \leq |\rho(x,u) - \rho(\bar{x},\bar{u})| + \\ &\qquad \gamma \Big|\max_{u'} [Q_\ell(f(x,u),u') - Q_\ell(f(\bar{x},\bar{u}),u')]\Big| \end{aligned}$$

Because ρ is Lipschitz continuous, $|\rho(x,u) - \rho(\bar{x},\bar{u})| \leq L_\rho(\|x - \bar{x}\|_2 + \|u - \bar{u}\|_2)$. For the second term, we have:

$$\begin{aligned} &\gamma \Big|\max_{u'} [Q_\ell(f(x,u),u') - Q_\ell(f(\bar{x},\bar{u}),u')]\Big| \\ &\qquad \leq \gamma \max_{u'} L_{Q_\ell} \|f(x,u) - f(\bar{x},\bar{u})\|_2 \\ &\qquad = \gamma L_{Q_\ell} \|f(x,u) - f(\bar{x},\bar{u})\|_2 \\ &\qquad \leq \gamma L_{Q_\ell} L_f(\|x - \bar{x}\|_2 + \|u - \bar{u}\|_2) \end{aligned}$$

where we used the Lipschitz continuity of Q_ℓ and f. Therefore, $L_{Q_{\ell+1}} = L_\rho + \gamma L_{Q_\ell} L_f = L_\rho + \gamma L_f L_\rho \sum_{k=0}^{\ell} \gamma^k {L_f}^k = L_\rho \sum_{k=0}^{\ell+1} \gamma^k {L_f}^k$ and the induction is complete. Taking the limit as $\ell \to \infty$, it follows that $L_Q = L_\rho \sum_{k=0}^{\infty} \gamma^k {L_f}^k$. Because $L_f < 1/\gamma$, this limit is finite and equal to $\frac{L_\rho}{1 - \gamma L_f}$. ■

Theorem 1 (Consistency): Under Assumption 1 and if Requirement 3 is satisfied, synchronous and asynchronous fuzzy Q-iteration are consistent, i.e., $\lim_{\delta_x \to 0, \delta_u \to 0} F(\theta^*) = Q^*$.

Proof: We will show that $\lim_{\delta_x \to 0, \delta_u \to 0} \varepsilon = 0$, where $\varepsilon = \min_{Q' \in \mathcal{F}_{FP}} \|Q^* - Q'\|_\infty$. Using (11), this implies that $\lim_{\delta_x \to 0, \delta_u \to 0} \|F(\theta^*) - Q^*\|_\infty = 0$, which is equivalent to the desired result.

[2]Consider the general continuous-time system $\dot{x} = g(x,u)$, where g is Lipschitz continuous with the Lipschitz constant L_g. The relationship of L_f to the sample time will depend on the discretization method. Here, we only study the Euler discretization of g: $x_{k+1} \approx x_k + T_\mathrm{s} g(x_k,u_k) = f(x_k,u_k)$, with T_s the sampling time. Then: $\|f(x,u) - f(\bar{x},\bar{u})\|_2 \leq \|x - \bar{x}\|_2 + T_\mathrm{s} \|g(x,u) - g(\bar{x},\bar{u})\|_2 \leq \|x - \bar{x}\|_2 + T_\mathrm{s} L_g(\|x - \bar{x}\|_2 + \|u - \bar{u}\|_2)$. Therefore, $L_f \leq 1 + T_\mathrm{s} L_g$ and can be made arbitrarily close to 1 by choosing a small sampling time.

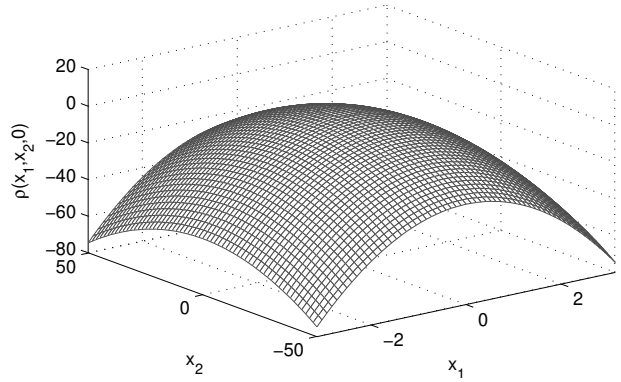


Fig. 1. A projection of the quadratic, continuous reward function (17) on the state space, for $u = 0$.

Define $Q' = FPQ^*$, i.e.,

$$Q'(x,u) = \sum_{i=1}^{N} \varphi_i(x) Q^*(x_i, u_j) \text{ with } j = \arg\min_{j'} \|u - u_{j'}\|_2$$

This Q-function is a fixed point of FP. We now determine an upper bound on $\|Q' - Q^*\|_\infty$. Take an arbitrary pair (x,u) and let $j = \arg\min_{j'} \|u - u_{j'}\|_2$. Then:

$$\begin{aligned} &|Q^*(x,u) - Q'(x,u)| \\ &\quad = \left| Q^*(x,u) - \sum_{i=1}^{N} \varphi_i(x) Q^*(x_i,u_j) \right| \\ &\quad \le |Q^*(x,u) - Q^*(x,u_j)| + \\ &\qquad \left| Q^*(x,u_j) - \sum_{i=1}^{N} \varphi_i(x) Q^*(x_i,u_j) \right| \end{aligned} \tag{14}$$

Because $\sum_{i=1}^{N} \varphi_i(x) = 1$, the second term can be written as:

$$\begin{aligned} &\left| \sum_{i=1}^{N} \varphi_i(x) \left[Q^*(x,u_j) - Q^*(x_i,u_j) \right] \right| \\ &\quad \le \sum_{i=1}^{N} \varphi_i(x) \left| Q^*(x,u_j) - Q^*(x_i,u_j) \right| \\ &\quad \le \sum_{i=1}^{N} \varphi_i(x) L_Q \|x - x_i\|_2 \le L_Q \nu \delta_x \end{aligned} \tag{15}$$

where the last step follows from Requirement 3, and from the Lipschitz continuity of Q^*. Using again the Lipschitz continuity of Q^*, and by the definition of δ_u, we get $|Q^*(x,u) - Q^*(x,u_j)| \le L_Q \|u - u_j\|_2 \le L_Q \delta_u$. Replacing this and (15) in (14), we find:

$$|Q^*(x,u) - Q'(x,u)| \le L_Q(\delta_u + \nu\delta_x)$$

Therefore, $\|Q^* - Q'\|_\infty \le L_Q(\delta_u + \nu\delta_x)$, and because L_Q and ν are finite, $\lim_{\delta_x \to 0, \delta_u \to 0} \|Q^* - Q'\|_\infty = 0$. Since $\varepsilon \le \|Q^* - Q'\|_\infty$, $\lim_{\delta_x \to 0, \delta_u \to 0} \varepsilon = 0$, which completes the proof. ■

V. Experimental Study: A Servo-System

In this section, a numerical example is used to illustrate the practical impact of discontinuities in the reward function on the consistency of the fuzzy Q-iteration algorithm. Consider the second-order discrete-time model of a servo-system:

$$\begin{aligned} x_{k+1} &= f(x_k, u_k) = Ax_k + Bu_k \\ A &= \begin{bmatrix} 1 & 0.0049 \\ 0 & 0.9540 \end{bmatrix}, \quad B = \begin{bmatrix} 0.0021 \\ 0.8505 \end{bmatrix} \end{aligned} \tag{16}$$

The sample time is $T_s = 0.005$ seconds. The position $x_{1,k}$ is bounded to $[-\pi, \pi]$, and the velocity $x_{2,k} \in [-16\pi, 16\pi]$. The control input $u_k \in [-10, 10]$. A linear system was chosen because it allows for extensive simulations to be performed with reasonable computational cost.

In the first part of our experiment, RL control was used to solve a discounted, quadratic regulation problem, described by the reward function (also shown in Figure 1):

$$\begin{aligned} r_{k+1} &= \rho(x_k, u_k) = -x_k^{\mathrm{T}} W x_k - w u_k^2 \\ W &= \begin{bmatrix} 5 & 0 \\ 0 & 0.01 \end{bmatrix}, \quad w = 0.01 \end{aligned} \tag{17}$$

The discount factor was chosen $\gamma = 0.95$. This reward function is smooth and has bounded support; therefore, it is Lipschitz. The transition function is Lipschitz with constant $L_f \le \max\{\|A\|_2, \|B\|_2\} = 1.0001 < 1/\gamma$. Therefore, the problem satisfies Assumption 1.

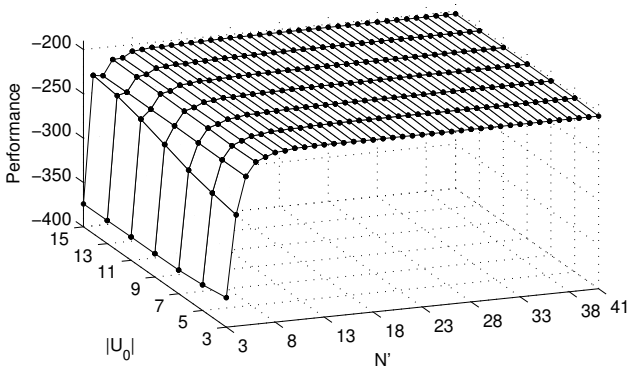
The aim of the second part of our experiment was to study the practical effect of a discontinuous reward function, which does not satisfy Assumption 1. To this end, a discontinuous reward function is required, but its choice cannot be arbitrary. Instead, to ensure a meaningful comparison between the solutions obtained with the original reward function (17) and those obtained with the new reward function, the quality of the policies must be preserved. One way to preserve it is to add a term of the form $\gamma\psi(f(x_k,u_k)) - \psi(x_k)$ to each reward $\rho(x_k,u_k)$, where $\psi(x)$ is an arbitrary bounded function [23]. A discontinuous function ψ is chosen, leading to the reward function (also shown in Figure 3):

$$\begin{aligned} \rho'(x_k,u_k) &= \rho(x_k,u_k) + \gamma\psi(f(x_k,u_k)) - \psi(x_k) \\ \psi(x) &= \begin{cases} 10 & \text{if } |x_1| \le \pi/4 \text{ and } |x_2| \le 4\pi \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{18}$$

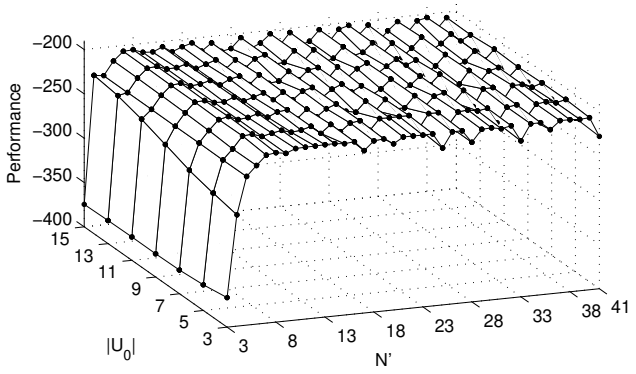
The quality of the policies is preserved by this modification, in the sense that for any policy h, $Q^h_{\rho'} - Q^*_{\rho'} = Q^h_\rho - Q^*_\rho$, where Q_ρ is a Q-function under the reward ρ. Indeed, it is easy to show by replacing ρ' in the expression (2) for the Q-function, that for any policy h, including any optimal policy, $Q^h_{\rho'}(x,u) = Q^h_\rho(x,u) - \psi(x)\ \forall x,u$ [23]. In particular, a policy is optimal for ρ' if and only if it is optimal for ρ.

The function ψ is positive in a rectangular region around the origin. Therefore, the newly added term rewards transitions that take the state inside this rectangular region, and penalizes transitions that take it outside.

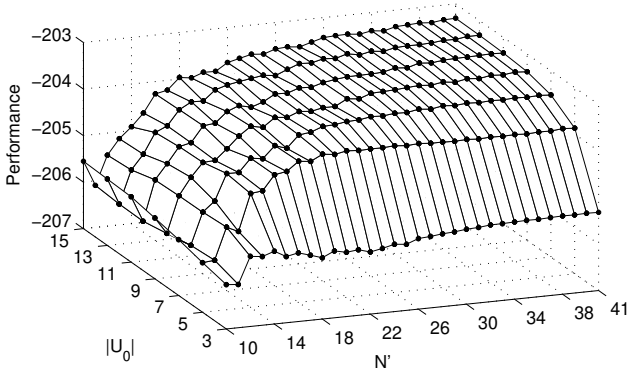
In order to study the consistency of fuzzy Q-iteration, a triangular fuzzy partition with N' equidistant cores for



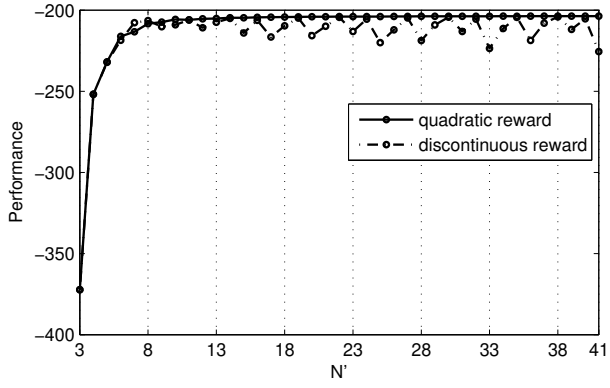
(a) Quadratic reward (17).



(b) Discontinuous reward (18); evaluation with quadratic reward.



(c) Quadratic reward, detail.



(d) Average performance over M, for varying N'.

Fig. 2. The performance of fuzzy Q-iteration as a function of N and M, for quadratic and discontinuous reward.

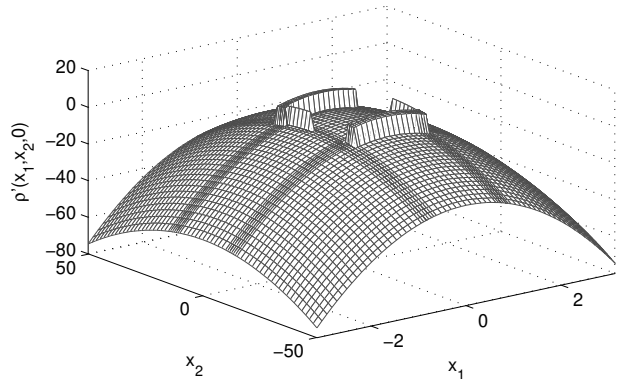


Fig. 3. A projection of the discontinuous reward function (18) on the state space, for $u = 0$.

each state variable was defined, leading to a total number of $N = N'^2$ fuzzy sets. The value of N' was gradually increased from 3 to 41. Similarly, the action was discretized into M equidistant values, with M ranging in $\{3, 5, \ldots, 15\}$ (only odd values were used because the 0 action is necessary for a good policy). Fuzzy Q-iteration was run for each combination of N' and M, and with both reward functions (17) and (18). The convergence threshold was set to $\varepsilon_{\text{QI}} = 10^{-5}$ to ensure the obtained parameter matrix is close to θ^*.

The performance of the policies obtained with fuzzy Q-iteration is given in Figure 2. Each point in these graphs corresponds to the return of the policy, averaged over the grid of initial states $X_0 = \{-\pi, -5\pi/6, -4\pi/6, \ldots, \pi\} \times \{-16\pi, -14\pi, \ldots, 16\pi\}$. The returns are evaluated using simulation, with a precision of $\varepsilon_R = 0.1$. The discounted infinite-horizon return of any state can be approximated in finite time by simulating only the first K steps and assuming that all the subsequent rewards are 0. To guarantee that an error of at most ε_R is introduced, $K = \left\lceil \log_\gamma \frac{\varepsilon_R(1-\gamma)}{\|\rho\|_\infty} \right\rceil$ where $\|\rho\|_\infty = \max_{x,u} |\rho(x,u)|$ is finite, and $\lceil \cdot \rceil$ produces the first integer larger than or equal to the argument.

Whereas the reward functions used for Q-iteration are different, the performance evaluation is always done with the reward (17). As explained, the change in the reward function preserves the quality of the policies, so comparing policies in this way is meaningful. The qualitative evolution of the performance is similar when evaluated with (18).

Discussion

When the continuous reward is used, the performance of fuzzy Q-iteration is close to optimal for $N' = 10$ and remains relatively smooth thereafter – see Figures 2(a) and 2(c). Also, the influence of the number of discrete actions is small for $N' \neq 4$. However, when the reward is changed to the discontinuous (18), the performance varies significantly as N' increases – see Figure 2(b). For many values of N',

the influence of M also becomes significant. Additionally, for many values of N' the performance is worse than with the continuous reward function – see Figure 2(d).

An interesting and somewhat counterintuitive fact is that the performance is not monotonous in N' and M. For a given value of N', the performance sometimes *decreases* as M increases. Similar situations occur as M is kept fixed and N' varies. This effect is present with both reward functions, but is much more significant in Figure 2(b) than in Figure 2(a) (see also Figure 2(c)). The magnitude of the changes decreases significantly as N' and M become large in Figures 2(a) and 2(c); this is not the case in Figure 2(b).

VI. Conclusions and Future Work

In this paper, we have proposed fuzzy Q-iteration, an approximate Q-iteration algorithm that relies on a fuzzy partition of the state space, and on a discretization of the action space. The consistency of fuzzy Q-iteration was proven under Lipschitz continuity assumptions on the system dynamics and reward function. In an example, a continuous reward function resulted in a more predictable performance improvement than a discontinuous reward, as the approximation accuracy increased. This shows that discontinuous rewards can harm RL performance in continuous-variable control tasks. Discontinuous rewards are common practice due to the origins of RL in artificial intelligence, where discrete-valued tasks are often considered.

Consistency can be proven also without imposing the restrictive bound $L_f \leq 1/\gamma$ on the Lipschitz constant of the dynamics. In that case, the conditions the MF s have to satisfy are slightly different from those in Requirement 3. This ongoing work will be reported in a subsequent publication.

The fuzzy approximator is pre-designed in our approach, and determines the computational complexity of fuzzy Q-iteration, as well as the accuracy of the solution. While we considered in this paper that the MF s were given *a priori*, we suggest as a future research direction to develop techniques that determine for a given accuracy an approximator with a small number of MF s. Another useful research direction is an extensive comparison of the performance (convergence, sub-optimality, consistency) of the various types of linear approximators that can be combined with the Q-value iteration algorithm (e.g., radial basis functions, Kuhn triangulations, polynomial approximation, etc.). Finally, action-space approximators more powerful than Voronoi partitions could be studied (e.g., approximators based on fuzzy partitions of the action space).

Acknowledgment: This research is financially supported by the BSIK-ICIS project "Interactive Collaborative Information Systems" (grant no. BSIK03024), by the NWO Van Gogh grant VGP 79-99, and by the STW-VIDI project DWV.6188.

References

[1] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, 3rd ed. Athena Scientific, 2007, vol. 2.

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

[3] L. Buşoniu, D. Ernst, B. De Schutter, and R. Babuška, "Fuzzy approximation for convergent model-based reinforcement learning," in *Proceedings 2007 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE-07)*, London, UK, 23–26 July 2007, pp. 968–973.

[4] ——, "Continuous-state reinforcement learning with fuzzy approximation," in *Adaptive Agents and Multi-Agent Systems III*, ser. Lecture Notes in Computer Science, K. Tuyls, A. Nowé, Z. Guessoum, and D. Kudenko, Eds. Springer, 2008, vol. 4865, pp. 27–43.

[5] P. Y. Glorennec, "Reinforcement learning: An overview," in *Proceedings European Symposium on Intelligent Techniques (ESIT-00)*, Aachen, Germany, 14–15 September 2000, pp. 17–35.

[6] T. Horiuchi, A. Fujino, O. Katai, and T. Sawaragi, "Fuzzy interpolation-based Q-learning with continuous states and actions," in *Proceedings 5th IEEE International Conference on Fuzzy Systems (FUZZ-IEEE-96)*, New Orleans, US, 8-11 September 1996, pp. 594–600.

[7] L. Jouffe, "Fuzzy inference system learning by reinforcement methods," *IEEE Transactions on Systems, Man, and Cybernetics—Part C: Applications and Reviews*, vol. 28, no. 3, pp. 338–355, 1998.

[8] H. R. Berenji and P. Khedkar, "Learning and tuning fuzzy logic controllers through reinforcements," *IEEE Transactions on Neural Networks*, vol. 3, no. 5, pp. 724–740, 1992.

[9] H. R. Berenji and D. Vengerov, "A convergent actor-critic-based FRL algorithm with application to power management of wireless transmitters," *IEEE Transactions on Fuzzy Systems*, vol. 11, no. 4, pp. 478–485, 2003.

[10] D. Vengerov, N. Bambos, and H. R. Berenji, "A fuzzy reinforcement learning approach to power control in wireless transmitters," *IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics*, vol. 35, no. 4, pp. 768–778, 2005.

[11] C.-K. Lin, "A reinforcement learning adaptive fuzzy controller for robots," *Fuzzy Sets and Systems*, vol. 137, no. 3, pp. 339–352, 2003.

[12] C.-S. Chow and J. Tsitsiklis, "An optimal one-way multigrid algorithm for discrete-time stochastic control," *IEEE Transactions on Automatic Control*, vol. 36, no. 8, pp. 898–914, 1991.

[13] G. Gordon, "Stable function approximation in dynamic programming," in *Proceedings Twelfth International Conference on Machine Learning (ICML-95)*, Tahoe City, US, 9–12 July 1995, pp. 261–268.

[14] J. N. Tsitsiklis and B. Van Roy, "Feature-based methods for large scale dynamic programming," *Machine Learning*, vol. 22, no. 1–3, pp. 59–94, 1996.

[15] M. S. Santos and J. Vigo-Aguiar, "Analysis of a numerical dynamic programming algorithm applied to economic models," *Econometrica*, vol. 66, no. 2, pp. 409–426, 1998.

[16] C. Szepesvári and R. Munos, "Finite time bounds for sampling based fitted value iteration," in *Proceedings Twenty-Second International Conference on Machine Learning (ICML-05)*, Bonn, Germany, 7–11 August 2005, pp. 880–887.

[17] M. Wiering, "Convergence and divergence in standard and averaging reinforcement learning," in *Proceedings 15th European Conference on Machine Learning (ECML'04)*, Pisa, Italy, 20–24 September 2004, pp. 477–488.

[18] D. Ormoneit and S. Sen, "Kernel-based reinforcement learning," *Machine Learning*, vol. 49, no. 2–3, pp. 161–178, 2002.

[19] D. Ernst, P. Geurts, and L. Wehenkel, "Tree-based batch mode reinforcement learning," *Journal of Machine Learning Research*, vol. 6, pp. 503–556, 2005.

[20] C. Szepesvári and W. D. Smart, "Interpolation-based Q-learning," in *Proceedings Twenty-First International Conference on Machine Learning (ICML-04)*, Bannf, Canada, 4–8 July 2004, pp. 791–798.

[21] S. P. Singh, T. Jaakkola, and M. I. Jordan, "Reinforcement learning with soft state aggregation," in *Advances in Neural Information Processing Systems 7*, G. Tesauro, D. S. Touretzky, and T. K. Leen, Eds., 1995, pp. 361–368.

[22] D. Ernst, "Near optimal closed-loop control. Application to electric power systems," Ph.D. dissertation, University of Liège, Belgium, March 2003.

[23] A. Y. Ng, D. Harada, and S. Russell, "Policy invariance under reward transformations: Theory and application to reward shaping," in *Proceedings Sixteenth International Conference on Machine Learning (ICML'99)*, Bled, Slovenia, 27–30 June 1999, pp. 278–287.