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 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]. An 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 means asymptotical convergence to the optimal solution as the approximation accuracy increases.

Fuzzy Q-iteration [3], [4] is a model-based 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 fuzzy membership values. Using results from the rich literature on RL with linear approximation [1], [5]–[8], fuzzy Q-iteration was shown in [3], [4] to converge to an approximate value function that is within a bound from the optimal value function.

A number of related fuzzy approximators have been proposed and applied to model-free RL, e.g., Q-learning [9], [10] or actor-critic algorithms [11]–[13]. However, most of these approaches are heuristic in nature, and their theoretical properties have not been investigated (notable exceptions where convergence is studied are the actor-critic algorithms in [12], [13]).

In this paper, we show that fuzzy Q-iteration is consistent. Namely, under appropriate assumptions on the dynamics and on the reward function, it is shown that the approximate solution converges to the optimal solution, asymptotically 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. The convergence properties of 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 motion system.

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 studied in Section IV. Section V uses an example to illustrate the performance impact of discontinuities in the reward function. Section VI outlines ideas for future work and concludes the paper.

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. Our algorithm and results can be easily extended to stochastic MDPs if the expectations can be evaluated exactly.

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

Lucian Buşoniu, Bart De Schutter, and Robert Babuška are with the 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 TU Delft. Damien Ernst is with Supélec, Rennes, France (email: damien.ernst@supelec.fr).

[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.

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 given 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 picks 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:

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

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. When the state (and / or the action space) is discrete, fuzzy approximation is useful when the number of discrete values is large, making the application of exact Q-iteration impractical.

Fuzzy Q-iteration was introduced in [3], [4] for discrete (or already discretized) action spaces. Here, the algorithm is extended to continuous action spaces using an explicit action discretization procedure.

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

Assumption 1: The fuzzy partition satisfies the following:

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

1.2. *(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, $\varphi_{\underline{i}}(x_i) = 0$ for all $\underline{i} \neq i$ by Assumption 1.1). The state x_i is called the core (center value) of the i-th set.

Assumption 1.1 is not restrictive, because any fuzzy partition can be normalized as long as for any x, there is some i such that $\varphi_i(x) > 0$. Assumption 1.2 is required here for brevity in the description and analysis of the algorithms; it can be relaxed using results of [5].

A discrete set of actions $U_0 = \{u_j | u_j \in U, j = 1, \ldots, M\}$ is chosen from the action space. The fuzzy approximator stores an $N \times M$ matrix of parameters, with one component $\theta_{i,j}$ corresponding to each pair of fuzzy core and discrete action (x_i, u_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 θ and outputs an approximate Q-function. For every state-action pair (x, u), this approximate Q-function is computed as follows:

$$\begin{gathered}\widehat{Q}(x, u) = [F(\theta)](x, u) = \sum_{i=1}^{N} \varphi_i(x) \theta_{i,j} \\ \text{with } j = \underset{\underline{j}=1,\ldots,M}{\arg\min} \|u - u_{\underline{j}}\|\end{gathered} \tag{5}$$

where $\|\cdot\|$, here as well as in the sequel, is the Euclidean norm of the argument. This is a linear basis functions form. For a fixed x, the approximator is piecewise constant over each cell (set) in the Voronoi partition of U corresponding to the discrete set of points U_0. Ties in the $\arg\min$ over $\underline{j}$ have to be broken consistently (e.g., always in favor of the smallest index that satisfies the condition).

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

$$\theta_{i,j} = [P(Q)]_{i,j} = Q(x_i, u_j) \tag{6}$$

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:

$$\theta_{\ell+1} = PTF(\theta_\ell) \tag{7}$$

This composite mapping is applied iteratively until θ has converged. The convergence criterion is usually approximate: $\max_{i,j} |\theta_{\ell+1,i,j} - \theta_{\ell,i,j}| \leq \varepsilon$. An approximately optimal policy can then be computed with:

$$h(x) = u_{j^*}, \quad j^* = \arg\max_j [F(\theta^*)](x, u_j) \tag{8}$$

Of course, any action in the j^*-th Voronoi cell of U 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].

It is easy to show that in order to compute the maximizations in the approximate Bellman updates of fuzzy Q-iteration, it suffices to consider only the discrete actions. This discrete-action version of Q-iteration is defined as follows:

$$[T_0(Q)](x,u) = \rho(x,u) + \gamma \max_{j=1,\dots,U} Q(f(x,u), u_j) \tag{9}$$

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, 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 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)* Define $\varepsilon = \min_{\underline{Q}} \left\| Q^* - \underline{Q} \right\|_\infty$ where $\underline{Q}$ is any fixed point of the composite mapping $FP : \mathcal{Q} \to \mathcal{Q}$. The convergence point θ^* satisfies:

$$\|Q^* - F(\theta^*)\|_\infty \leq \frac{2\varepsilon}{1-\gamma} \tag{10}$$

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.

Define the resolutions of the fuzzy approximator over the state space, and respectively over the action space:

$$\delta_x = \max_{x \in X} \min_{i=1,\dots,N} \|x - x_i\| \tag{11}$$

$$\delta_u = \max_{u \in U} \min_{j=1,\dots,M} \|u - u_j\| \tag{12}$$

where x_i is the core of the i-th membership function, and u_j is the j-th discrete action. The goal is to show that $\lim_{\delta_x \to 0,\ \delta_u \to 0} F(\theta^*) = Q^*$. The following assumptions are made.

Assumption 2: There exists a finite $\nu > 0$ such that, regardless of N, the fuzzy membership functions satisfy:

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

In the end of Section IV, we show that triangular fuzzy partitions (of the type used in the example of Section V) satisfy this assumption.

Assumption 3: The dynamics f and the reward function ρ satisfy the following conditions:

3.1. *(Lipschitz continuity)* They are Lipschitz continuous with Lipschitz constants L_f and respectively L_ρ:

$$\|f(x,u) - f(\underline{x},\underline{u})\| \leq L_f(\|x - \underline{x}\| + \|u - \underline{u}\|)$$
$$|\rho(x,u) - \rho(\underline{x},\underline{u})| \leq L_\rho(\|x - \underline{x}\| + \|u - \underline{u}\|)$$

3.2. The Lipschitz constant of f satisfies: $L_f < 1/\gamma$.

Lipschitz conditions similar to Assumption 3.1 are typically needed to prove consistency of approximate RL algorithms.

As a first step to proving consistency, the Lipschitz continuity of Q^* is established. This will help later on in proving that a fixed-point of FP can be made arbitrarily close to Q^* by increasing the resolution of the approximator. Consistency will then follow from item 1.3 of Proposition 1.

Lemma 1 (Lipschitz continuity of Q^*): There exists a finite L_Q such that under Assumption 3:

$$|Q^*(x,u) - Q^*(\underline{x},\underline{u})| \leq L_Q(\|x - \underline{x}\| + \|u - \underline{u}\|)$$

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_ρ, and:

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

Due to Assumption 3.1, $|\rho(x,u) - \rho(\underline{x},\underline{u})| \le L_\rho(\|x - \underline{x}\| + \|u - \underline{u}\|)$. For the second term:

$$\begin{aligned}
&\gamma \left| \max_{u'} [Q_\ell(f(x,u),u') - Q_\ell(f(\underline{x},\underline{u}),u')] \right| \\
&\qquad \le \gamma \max_{u'} L_{Q_\ell} \|f(x,u) - f(\underline{x},\underline{u})\| \\
&\qquad = \gamma L_{Q_\ell} \|f(x,u) - f(\underline{x},\underline{u})\| \\
&\qquad \le \gamma L_{Q_\ell} L_f (\|x - \underline{x}\| + \|u - \underline{u}\|)
\end{aligned}$$

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 $L_Q = L_\rho \sum_{k=0}^{\infty} \gamma^k {L_f}^k$ which under Assumption 3.2 is finite and equal to $L_Q = \frac{L_\rho}{1 - \gamma L_f}$. ■

Proposition 2 (Consistency): Under Assumption 2 and Assumption 3, 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 $\varepsilon = \min_{\underline{Q}} \left\| Q^* - \underline{Q} \right\|_\infty$ with $\underline{Q}$ any fixed point of FP. Using Proposition 1.3, this implies that $\lim_{\delta_x \to 0, \delta_u \to 0} \|F(\theta^*) - Q^*\|_\infty = 0$, which is equivalent to the desired result.

Define $\underline{Q} = FPQ^*$, i.e.,

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

This Q-function is a fixed point of FP. We now establish an upper bound on $\left\| \underline{Q} - Q^* \right\|_\infty$. Obviously, $\left| Q^*(x_i,u_j) - \underline{Q}(x_i,u_j) \right| = 0$ because $\underline{Q}(x_i,u_j) = Q^*(x_i,u_j)$. Take now x, u such that $x \ne x_i$ $\forall i$, or $u \notin U_0$, and let $j = \arg\min_{\underline{j}} \|u - u_{\underline{j}}\|$. Then:

$$\begin{aligned}
\left| Q^*(x,u) - \underline{Q}(x,u) \right| &= \left| Q^*(x,u) - \sum_{i=1}^{N} \varphi_i(x) Q^*(x_i,u_j) \right| \\
&\le |Q^*(x,u) - Q^*(x,u_j)| + \\
&\quad \left| Q^*(x,u_j) - \sum_{i=1}^{N} \varphi_i(x) Q^*(x_i,u_j) \right|
\end{aligned} \tag{13}$$

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

$$\begin{aligned}
&\left| \sum_{i=1}^{N} \varphi_i(x) \left[Q^*(x,u_j) - Q^*(x_i,u_j) \right] \right| \\
&\qquad \le \sum_{i=1}^{N} \varphi_i(x) \left| Q^*(x,u_j) - Q^*(x_i,u_j) \right| \\
&\qquad \le \sum_{i=1}^{N} \varphi_i(x) L_Q \|x - x_i\| \le L_Q \nu \delta_x
\end{aligned} \tag{14}$$

where the last step follows from Assumption 2, and the Lipschitz continuity of Q^* was used. Using again the Lipschitz continuity of Q^*, and the definition of δ_u, $|Q^*(x,u) - Q^*(x,u_j)| \le L_Q \|u - u_j\| \le L_Q \delta_u$. Using this and (14) in (13), we find:

$$\left| Q^*(x,u) - \underline{Q}(x,u) \right| \le L_Q(\delta_u + \nu \delta_x)$$

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

Triangular fuzzy partitions

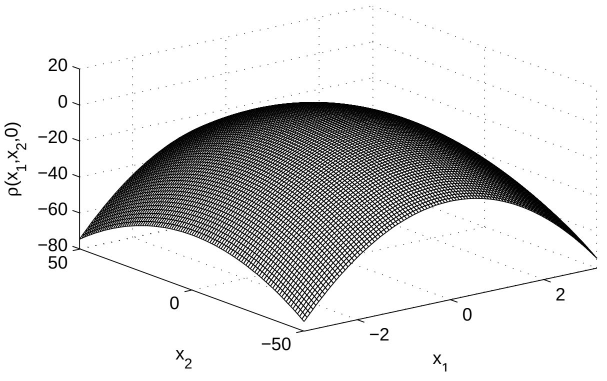
A simple type of fuzzy partition that satisfies Assumption 1 can be obtained as follows. For each state variable x_d with $d = 1, \ldots, n$, a number N_d of triangular fuzzy membership functions are defined as follows:

$$\begin{aligned}
\varphi_{d,1}(x_d) &= \max\left(0, \frac{c_{d,2} - x_d}{c_{d,2} - c_{d,1}}\right) \\
\varphi_{d,i}(x_d) &= \max\left[0, \min\left(\frac{x_d - c_{d,i-1}}{c_{d,i} - c_{d,i-1}}, \frac{c_{d,i+1} - x_d}{c_{d,i+1} - c_{d,i}}\right)\right] \\
&\qquad \text{for } i = 2, \ldots, N_d - 1 \\
\varphi_{d,N_d}(x_d) &= \max\left(0, \frac{x_d - c_{d,N_d-1}}{c_{d,N_d} - c_{d,N_d-1}}\right)
\end{aligned}$$

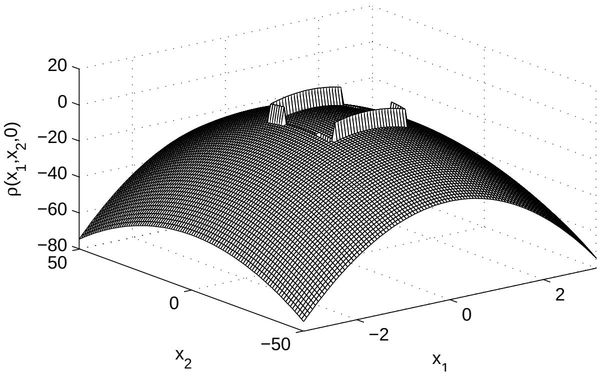
where $c_{d,1} < \ldots < c_{d,N_d}$ is the array of cores, which completely determines the shape of the membership functions, and $x_d \in [c_{d,1}, c_{d,N_d}]$. Adjacent functions always intersect at a 0.5 membership level. Then, the product of each combination of (single-dimensional) membership functions yields a pyramidal-shaped n-dimensional membership function in the fuzzy partition of X. Next, it is shown that this type of fuzzy partition also satisfies Assumption 2.

For simplicity, assume that the fuzzy cores are equidistant along each dimension of X, and the distance between two adjacent cores is the same along every dimension. This means that every point $x \in X$ falls inside a hypercube, or on the common boundary of several identical hypercubes. In the latter case, just pick any of these hypercubes. Only the membership functions with the cores in the corners of the hypercube can take non-zero values, and the number of corners is 2^n where n is the dimension of X. Furthermore, the distance between any point inside the hypercube and any of its corners is at most $2\delta_x$. Therefore, we have:

$$\sup_{x \in X} \sum_{i=1}^{N} \varphi_i(x) \|x - x_i\| \le 2^n \, 2 \, \delta_x$$



(a) Quadratic reward (16).



(b) Discontinuous reward (17).

Fig. 1. A projection of the reward functions on the state space, for $u = 0$.

and a choice of $\nu = 2^{n+1}$ indeed satisfies Assumption 2.

V. Experimental Study

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 motion system described by the following discrete-time dynamics:

$$\begin{gathered} x_{k+1} = f(x_k, u_k) = Ax_k + Bu_k \\ A = \begin{bmatrix} 1 & 0.0049 \\ 0 & 0.9540 \end{bmatrix}, \quad B = \begin{bmatrix} 0.0021 \\ 0.8505 \end{bmatrix} \end{gathered} \tag{15}$$

The sample time is $T_s = 0.005$ seconds. The position $x_{1,k}$ is bounded to $[-\pi, \pi]$, and the velocity $x_{2,k}$ to $[-16\pi, 16\pi]$. The control input $u_k \in [-10, 10]$.

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

$$\begin{gathered} r_{k+1} = \rho(x_k, u_k) = -x_k^{\mathrm{T}} W x_k - w u_k^2 \\ W = \begin{bmatrix} 5 & 0 \\ 0 & 0.01 \end{bmatrix}, \quad w = 0.01 \end{gathered} \tag{16}$$

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 $L_f \leq \max\{\|A\|_2, \|B\|_2\} = 1.0001 < 1/\gamma$. Therefore, the problem satisfies Assumption 3.

In the second part of the experiment, a discontinuous term was added to the reward function (16):

$$\begin{gathered} \rho'(x_k, u_k) = \rho(x_k, u_k) + \gamma\psi(f(x_k, u_k)) - \psi(x_k) \\ \psi(x) = \begin{cases} 10 & \text{if } |x_1| \leq \pi/4 \text{ and } |x_2| \leq 4\pi \\ 0 & \text{otherwise} \end{cases} \end{gathered} \tag{17}$$

The term $\gamma\psi(f(x_k, u_k)) - \psi(x_k)$ is called a shaping reward [14]. It needs to have this specific form in order to preserve the quality of the policies, in the sense that for any policy h, $\left\|Q^h_{\rho'} - Q^*_{\rho'}\right\|_\infty \leq \varepsilon$ implies $\left\|Q^h_{\rho} - Q^*_{\rho}\right\|_\infty \leq \varepsilon$, where Q_ρ is a Q-function under the reward ρ [14]. The function ψ is chosen positive in a rectangular region around the origin. Therefore, the shaping term rewards transitions that take the state inside this rectangular region, and penalizes transitions that take it outside. A projection of ρ' on X, for $u = 0$, is presented in Figure 1(b).

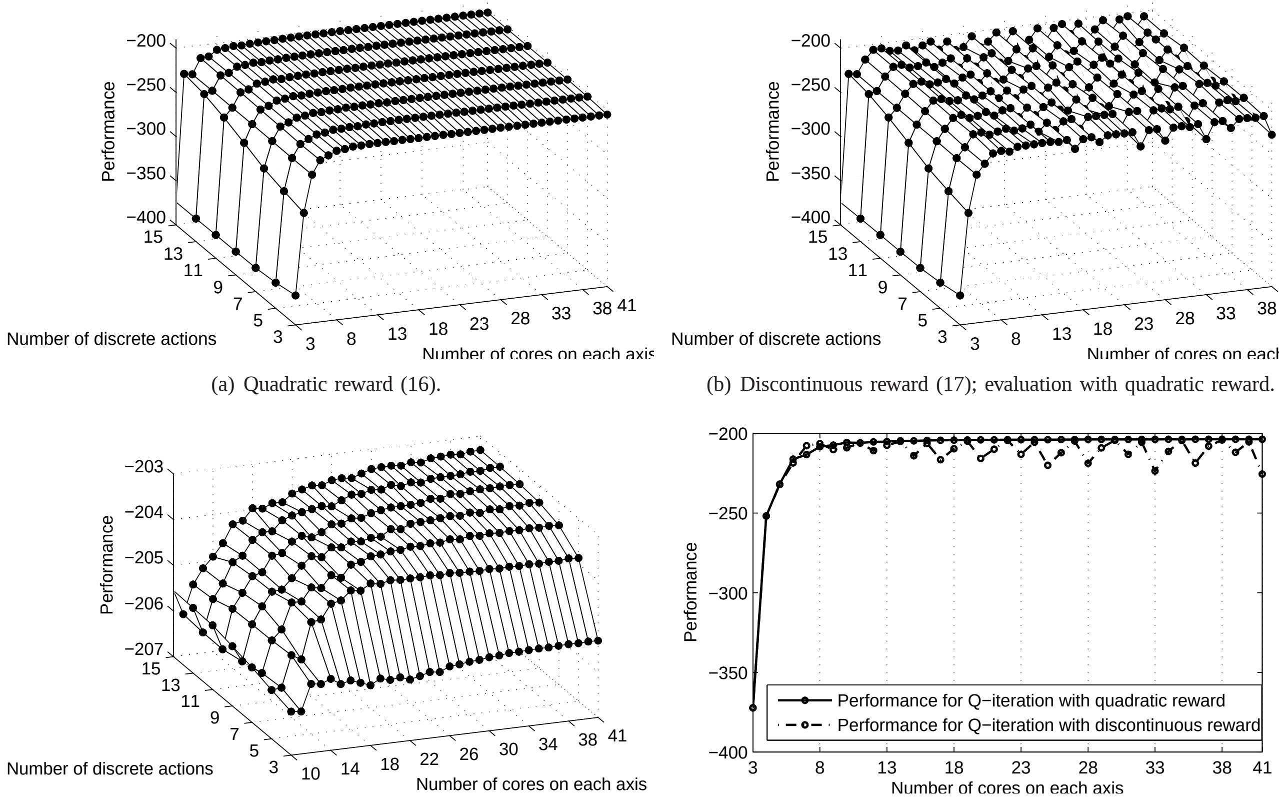
In order to study the consistency of fuzzy Q-iteration, a triangular fuzzy partition with $\underline{N}$ equidistant cores for each state variable was defined, leading to a total number of $N = \underline{N}^2$ fuzzy sets. The value of $\underline{N}$ was gradually increased from 3 to 41. Similarly, 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 (16) and (17). The convergence threshold was set to $\varepsilon_{\mathrm{QI}} = 10^{-5}$ to ensure the obtained parameter vector 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 0.1. Whereas the reward functions used for Q-iteration are different, the performance evaluation is always done with the reward (16). As explained, the change in the reward function preserves the quality of the policies, so comparing policies in this fashion is meaningful. The qualitative evolution of the performance is similar when evaluated with (17).

Discussion

When the continuous reward is used, the performance of fuzzy Q-iteration is close to optimal for $\underline{N} = 10$ and remains relatively smooth thereafter – see Figures 2(a) and 2(c). Also, the influence of the number of discrete actions is small for $\underline{N} \neq 4$. However, when the reward is changed to the discontinuous (17), the performance varies significantly as $\underline{N}$ increases – see Figure 2(b). For many values of $\underline{N}$, the influence of M becomes significant. Additionally, for many values of $\underline{N}$ 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 $\underline{N}$, the performance sometimes *decreases*



(a) Quadratic reward (16).

(b) Discontinuous reward (17); evaluation with quadratic reward.

(c) Quadratic reward, detail.

(d) Average performance over M, for varying $\underline{N}$.

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

as M increases. Similar situations occur as M is kept fixed and $\underline{N}$ 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 $\underline{N}$ 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, the consistency of the approximate, fuzzy Q-iteration algorithm was proven. A discretization procedure was introduced to approximate the continuous (or discrete but large) action space. Fuzzy Q-iteration was applied to a control problem where using a smooth reward function provided more predictable performance than using discontinuous rewards, as the accuracy increased. This shows that, at least in some continuous-variable RL tasks, discontinuous rewards can harm performance. Discontinuous rewards are common practice due to the origins of RL in artificial intelligence, where discrete-valued tasks are often considered.

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 membership functions 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 membership functions. 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, etc.). Finally, action-space approximators more powerful than Voronoi partitions could be investigated (e.g., approximators based on fuzzy partitions of the action space).

Acknowledgement: 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 Artificial Intelligence, K. Tuyls, A. Nowé, Z. Guessoum, and D. Kudenko, Eds., 2007, vol. 4865, accepted.

[5] 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.

[6] 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.

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

[8] C. Szepesvári and W. D. Smart, “Interpolation-based Q-learning,” in *Proceedings 21st International Conference on Machine Learning (ICML-04)*, Bannf, Canada, 4–8 July 2004.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.