Comparison of Reinforcement Learning Algorithms applied to the Cart-Pole Problem

Savinay Nagendra
Dept. Elect. Electro. Engg.,
PES Inst. of Tech.,
Bangalore, India.
nagsavi17@gmail.com

Nikhil Podila
Dept. Elect. Electro. Engg.,
PES Inst. of Tech.,
Bangalore, India.
nikhilpodila.94@gmail.com

Rashmi Ugarakhod
PES Cen. Int. Systems;
Dept. Electro. Commun. Engg.,
PES University,
Bangalore, India.
rashmi.ugarakhod@gmail.com

Koshy George
PES Cen. Int. Systems;
Dept. Electro. Commun. Eng.,
PES University,
Bangalore, India.
kgeorge@pes.edu

***Abstract*—Designing optimal controllers continues to be challenging as systems are becoming complex and are inherently nonlinear. The principal advantage of reinforcement learning (RL) is its ability to learn from the interaction with the environment and provide an optimal control strategy. In this paper, RL is explored in the context of control of the benchmark cart-pole dynamical system with no prior knowledge of the dynamics. RL algorithms such as temporal-difference, policy-gradient actor-critic, and value-function approximation are compared in this context with the standard linear quadratic regulator solution. Further, we propose a novel approach for integrating RL and swing-up controllers.**

***Index Terms*—Intelligent systems, learning systems, nonlinear control systems.**

I. Introduction

Reinforcement learning (RL) is a branch of machine learning inspired by human and animal behaviourist psychology. When RL is applied to a system, its agents learn to take those actions in an environment that maximise cumulative reward. Learning is based on several forms of evaluative feedback [1], [2]. In contrast to supervised learning methods, RL is used when the target outputs are not known. The effect of an output on the environment is used to evaluate the performance of an agent; this effect is quantified in terms of an evaluation or reinforcement signal. When an agent exploits current knowledge, reward is maximised. However, better action selections are possible only if it explores uncharted space. Exploration or exploitation should not be pursued exclusively. Accordingly, finding a balance between the two is typically incorporated in an RL algorithm.

The cart-pole inverted-pendulum balancing problem is a classical benchmark problem for control purposes. It is an inherently unstable and under-actuated mechanical system. The dynamics of this system is used to understand tasks such as the maintenance of balance (e.g., a human walking), control of rocket thrusters and self-balancing mechanical systems. A number of control design techniques for swing-up and stabilisation of an inverted pendulum have been investigated. Examples include energy-based controllers, PID controllers, linear quadratic regulators, and fuzzy logic controllers; e.g., [3], [4].

An increase in the complexity of systems requires the need for sophisticated controllers especially in the presence of nonlinearities, uncertainty and time-variations. By its inherent nature, RL has the capability to use knowledge from the environment to provide optimal controllers without the knowledge of the environment. Moreover, such controllers have the capability of adapting to a changing environment.

Several researchers have applied RL to the cart-pole inverted-pendulum balancing problem. In [5], the authors introduced the notion of BOXES to adaptively control the pole. A computationally more efficient actor-critic method was proposed in [6] where they use neuron-like adaptive elements. Optimal control was achieved using a single-layer neural network. Algorithms such as Q-learning and state-action-reward-state-action (SARSA) were subsequently formulated from the temporal credit assignment learning suggested in [7]. Multi-layer neural networks were used in [8] to improve the efficiency. More recently, offline (or batch) RL algorithms such as the least squares policy iteration [9] have been tested.

The goal of this paper is two-fold. First, we compare several RL algorithms in the context of the cart-pole inverted-pendulum balancing problem. Specifically, we consider Q-learning, actor-critic policy-gradient method, and temporal difference with value-function approximation. The model for the cart-pole dynamics is unknown to these RL algorithms. Through the trade-off between exploration and exploitation, an RL agent eventually learns the set of actions that minimises the occurrence of a failure signal. In turn, this leads to an optimal control law.

In Q-learning, the quality of a particular state is assessed by means of a value-function, and one-step updates are determined based on the reward that a system gets at each time instant. The selection of an action in the actor-critic policy-gradient algorithm is stochastic in nature. Such algorithms are particularly useful for those problems wherein determining the actual value-function is rather complex; an example problem is the one considered in this paper. In those problems modelled by continuous-state Markov decision processes, the use of value-function approximation increases memory efficiency. The value-function is learnt by generalising the values obtained from those states visited by the RL agent to those states

978-1-5090-6367-3/17/$31.00 ©2017 IEEE

in their neighbourhood that are not visited.

The second objective in this paper is to propose a method to integrate any RL algorithm with a swing-up controller. We use this to compare the performances of the aforementioned RL algorithms with that of a linear quadratic regulator (LQR) solution.

The rest of the paper is organised as follows: The cart-pole inverted-pendulum balancing problem is described in Section II. The different RL algorithms that are considered in this paper are presented in Section III. The classical swing-up control together with LQR stabilisation is dealt with in Section IV. The manner in which any RL algorithm is integrated with a swing-up controller is given in Section V. The performances of these controllers are compared in Section VI. A few concluding remarks are made in Section VII.

II. The Cart-Pole Inverted-Pendulum Balancing Problem

The cart-pole inverted-pendulum balancing problem (CPBP) is a benchmark for RL algorithms; e.g., [5]–[8]. The fundamental problem statement has been derived from an adaptive control technique known as BOXES [5]. The problem is challenging as the RL agent has to select and take actions in a very limited and discrete action space.

A. Cart-Pole Dynamics

A pole is pivoted to a cart which is allowed to move along the horizontal axis. The goal of the problem is to balance the pole in an upright position by using bi-directional forces that is imparted on the cart by an electromechanical system. The state of the system at any time instant is defined by the angular position θ (measured with respect to the upright position) and velocity $\dot{\theta}$ of the pendulum, and the linear position x and velocity $\dot{x}$ of the cart. The dynamics of the nonlinear system is described by the following equations:

$$\begin{aligned} \ddot{\theta} &= \frac{(M+m)g\sin\theta - \cos\theta(F + ml\dot{\theta}^2\sin\theta)}{\frac{4}{3}(M+m)l - ml\cos^2\theta} \\ \ddot{x} &= \frac{F + ml(\dot{\theta}^2\sin\theta - \ddot{\theta}\cos\theta)}{M+m} \end{aligned} \quad (1)$$

Here, M is the mass of the cart (0.711 kg), m is the mass of pendulum (0.209 kg), g is the acceleration due to gravity (9.8 m/s^2), F is the force applied on the cart ($\pm F_m$ N), and l is the length from the centre of mass to the pivot (0.326 m). For purposes of simulation we assume a sampling interval of $\tau = 0.02$ s.

B. Formulation of RL Model for the CPBP

The objective of the RL agent is to control the plant (1) using an appropriate sequence of control actions in order to balance the pendulum so that (i) the angular position of the pendulum remains within $\pm 12^\circ$ from the upright position, and (ii) the linear position of the cart is within ± 2.4 m from the centre of the track. This admissible space is denoted $\mathcal{O}$.

The state variables θ, $\dot{\theta}$, x and $\dot{x}$ are quantised and separated into multiple bins. A box is defined as an n-tuple comprising of one bin from each of the four state variables. Two functions *getBox* and *getBox2* have been defined with different quantisation levels. The former comprises 162 boxes when quantised into 15 bins in the following manner: θ: $[-12,-6)$, $[-6,-1)$, $[-1,0)$, $[0,1)$, $[1,6]$, $[6,12]$; $\dot{\theta}$: $(-\infty,-50)$, $[-50,50]$, $[50,\infty)$; x: $[-2.4,-0.8)$, $[-0.8,0.8]$, $(0.8,2.4]$; and, $\dot{x}$: $(-\infty,-0.5)$, $[-0.5,0.5]$, $(0.5,\infty)$. In contrast, getBox2 comprises 324 boxes when quantised into 21 bins: θ: $[-12,-6)$, $[-6,-4)$, $(-4,-3]$, $(-3,-2)]$, $(-2,-1]$, $[-1,0)$, $[0,1)$, $(1,2]$, $(2,3]$, $(3,4]$, $(4,6]$, $(6,12]$; $\dot{\theta}$: $(-\infty,-50)$, $[-50,50]$, $[50,\infty)$; x: $[-2.4,-0.8)$, $[-0.8,0.8]$, $(0.8,2.4]$; and, $\dot{x}$: $(-\infty,-0.5)$, $[-0.5,0.5]$, $(0.5,\infty)$. If at any time the state does not belong to these discrete bins, then a failure is said to have occurred and a reinforcement signal of value -1 is generated, which marks the end of an episode. After a failure, the pole is reset to the upright unstable equilibrium position, which is the initial position for the next episode.

Markov decision processes (MDP) play an important role in RL [10]. They provide a mathematical framework to enable decision-making in situations where the outcome is uncertain despite a specific action chosen by an agent. The core problem of MDPs is to find a policy for the agent based on its current state. The CPBP can also be described using an MDP: It consists of (i) $\mathcal{S}$, a set of states S_t: θ, $\dot{\theta}$, x, $\dot{x}$; (ii) $\mathcal{A}$, a set of actions A_t: moving the cart left or right by applying a force $\pm F_m$; (iii) $\mathcal{P}$, a state-transition probability matrix (STPM), with elements $p(a,s,s') = Pr\{S_{t+1} = s' | S_t = s, A_t = a\}$. Since the environment is deterministic, taking an action a from state s would always result in the next state s'. Thus, the STPM simplifies to the state transition matrix (STM). STM can be defined based on the dynamics of the system, given by (1); (iv) $\mathcal{R}$, the set of rewards R_t; the expected reward is $r(a,s) = \mathcal{E}\{R_{t+1} | S_t = s, A_t = a\}$. A reward of -1 is given to the system whenever the boundary of $\mathcal{O}$ is crossed, and a reward of zero is awarded otherwise; and (v) a discount factor $\gamma \in [0,1]$ which determines the extent to which future rewards rely on the current state and action. The discount factor is varied according to the algorithm used.

In CPBP — like many other complex control problems — complete knowledge of the MDP cannot be obtained as the STPM cannot be determined before achieving optimal control. This is because the RL agent learns by taking random actions with no knowledge of correct actions until the optimal policy has been obtained. The RL algorithms considered in this paper are model-free. We discuss these in the next section.

III. Model-Free Learning

The class of algorithms that are used to solve MDPs without the knowledge of the reward function $r(a,s)$, or the state-transition probabilities $p(a,s,s')$, are termed as model-free methods [11]. They rely on MDPs to sample various sizes of experiences from the environment depending on the algorithm used. These experiences are then used by an agent to make decisions to take appropriate actions. Algorithms for model-free learning can be categorised into various types based on

the length of backups, which is the number of steps for which the RL agent interacts with the environment before updating its estimate of the quality of a state. These include one-step backup or temporal difference learning (denoted TD(0)), infinite-step or full-length backup or Monte Carlo learning, n-step backup or multi-step backup with TD(λ) as a special case that uses the discount factor whilst taking backups. Model-free methods largely use the generalised policy iteration (GPI), initially developed for dynamic programming [12] to evaluate the environment and select actions.

In GPI, policy evaluation and policy improvement work together to find the optimal solution whilst dealing with the exploration-exploitation trade-off that exists in reinforcement learning. In this paper, we consider off-policy RL algorithms for optimal control. Here, an agent uses one policy — usually a greedy one — to select actions from action value estimates, and another policy — usually an exploratory one — to generate action value estimates.

A. Q-learning: Off-Policy TD Control Algorithm

In temporal-difference (TD) learning, the estimate of the quality of a state is updated by bootstrapping without the knowledge of the final outcome. The observed reward and an old estimate of the quality of the next state [11] is used to obtain the one-step update in TD(0). Q-learning is an off-policy TD algorithm. The policy used to select actions using the state-action values is the greedy policy, given by $\max_{a'} Q(S_{t+1}, a')$. On the other hand, usually an exploratory policy is used to generate the action-value estimates. The Q-learning update equation is given by:

$$\begin{aligned} Q(S_t, A_t) \quad \leftarrow \quad & Q(S_t, A_t) + \alpha R_{t+1} \\ & + \alpha \left(\gamma \max_{a'} Q(S_{t+1}, a') - Q(S_t, A_t) \right) \end{aligned} \tag{2}$$

where (S_t, A_t) corresponds to the current state and action, (S_{t+1}, a') corresponds to the next state and action, $Q(S_t, A_t)$ is the quality of being in the state S_t whilst taking action A_t (and approximates the optimal action-value function), R_{t+1} is the reward obtained by taking an action from state S_t, α is the step-size, and γ is the discount factor. Each step of Q-learning updates a state-action pair: (i) The current state S_t, or the internal state of the cart-pole dynamics represented in a form that the RL agent can interpret. (ii) The action A_t to be taken at state S_t.

In CPBP, action A_t is a constant force on the cart towards the left or right. Also, the RL agent is punished with a reinforcement signal R_t of -1 if either x or θ leaves $\mathcal{O}$, and zero otherwise. Using (2), Q-learning is applied to CPBP as follows:

- Initialise $Q(s, a)$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$.
- For each step in an episode:
  - Given current state S_t, let $A_t = \arg\max_a Q(S_t, a)$.
  - Take the action A_t.
  - Observe R_{t+1} and S_{t+1} from the environment.
  - Update the action-value function $Q(S_t, A_t)$, to the Q-target $R_{t+1} + \gamma \max_{a'} Q(S_{t+1}, a')$ using (2).
  - Proceed until the terminal state, where the state S_t exceeds the limits set by the objective.

Since the state-space of the MDP corresponding to CPBP is explored even with a greedy policy, the two policy algorithms followed by off-policy control are chosen as greedy policies. As a result of this selection, this special case of off-policy method converges to an on-policy approach.

B. Actor-Critic Policy-Gradient Method

1) Policy Gradient: The RL algorithms considered so far involve a value-function which quantifies the significance of the system being in the current state and taking the specified action. Action selection using these value-function based methods can be performed either deterministically or with some stochasticity that can be reduced to a deterministic form based on the overall performance of the agent [11].

In policy-gradient methods, the policy is represented by a parametrised probability distribution

$$\pi(s, a, \theta) = Pr\{A_t = a | S_t = s; \theta\}, \tag{3}$$

differentiable with respect to θ, that selects action a in state s in accordance to the parameter θ. In order to determine the optimal θ we use the following average reward per time-step

$$J_R(\theta) = \sum_s d_\pi(s) \sum_a \pi(s, a, \theta) r(s, a) \tag{4}$$

where $d_\pi(s)$ is the stationary distribution of the Markov chain under π. The policy gradient is then $\nabla_\theta J_R(\theta)$ which requires $\nabla_\theta \pi(s, a, \theta)$. Clearly,

$$\begin{aligned} \nabla_\theta \pi(s, a, \theta) &= \pi(s, a, \theta) \frac{\nabla_\theta \pi(s, a, \theta)}{\pi(s, a, \theta)} \\ &= \pi(s, a, \theta) \nabla_\theta \log \pi(s, a, \theta) \end{aligned} \tag{5}$$

The score function $\nabla_\theta \log \pi(s, a, \theta)$ is easier to determine if a *softmax* policy is chosen:

$$\nabla_\theta \log \pi(s, a, \theta) = \phi(s, a) - \mathcal{E}\{\phi(s, \cdot)\} \tag{6}$$

where $\phi(s, a)$ is a feature vector.

2) Actor-Critic Method: The high variance observed in policy-gradient methods is a drawback. To overcome this drawback, actor-critic method was proposed in [8], [13]. Two networks are proposed in [8] — action and critic networks. The former learns to select actions as a function of the states of the CPBP and consists of a single neuron having two possible outputs, $\pm F_m$. The probability of generating each action depends on the box which contains the state S_t. The weights are initialised to zero. Accordingly, the two actions are equally probable initially. These weights are incrementally updated (and hence the action probabilities) after receiving non-zero reinforcement signals after each failure.

The critic network provides an association between the state vectors and the failure signal. The evaluation network also consists of a single neuron which learns the expected value of a discounted sum of future failure signals by means of TD learning [8]. This network learns to predict how soon a failure is expected to occur given that the current state is S_t.

This provides feedback for the action network enabling it to learn to select the appropriate action A_t when the state is S_t.

By estimating the action-value function using a critic, the high variance of the policy-gradient method can be reduced. Thus, policy-gradient actor-critic methods are considered with the following parts [7]: The critic which updates action-value function $Q(S_t, A_t)$ or their parameters, and the actor which updates the policy parameters θ in the direction of the action-value function as estimated by the critic.

The update equations for the actor network parameters are given by:

$$\theta_{i,t+1} = \theta_{i,t} + \alpha \left(r_{t+1} + \gamma p^s_{t+1} - p^s_t\right) \bar{e}_{i,t} \tag{7}$$

where $\bar{e}_{i,t} = (1-\delta)\bar{e}_{i,t-1} + \delta e_{i,t}$, $e_{i,t} = (y_t - 0.5)x_{i,t}$, with $\bar{e}_{i,0} = 0$ for $0 < \delta < 1$, and

$$y_t = \begin{cases} 1, & s_t + \eta_t > 0 \\ 0, & \text{otherwise} \end{cases}.$$

Here, η_t is a zero mean random variable with standard deviation 0.1, $s_t = \sum_{i=1}^{n} \theta_{i,t} x_{i,t}$, $p^s_t = \sum_{i=1}^{n} \theta_{i,t} x_{i,t}$, and $p_t = r_1$. The update equations for the critic network parameters are given by

$$w_{i,t+1} = w_{i,t} + \beta(r_{t+1} + \gamma p^s_{t+1} - p^s_t)\bar{x}_{i,t} \tag{8}$$

where $\bar{x}_{i,t} = (1-\delta)\bar{x}_{i,t-1} + \delta x_{i,t}$, with $x_{i,0} = 0$ for $0 < \delta < 1$, $p^s_t = \sum_{i=1}^{n} \theta_{i,t} x_{i,t}$, $p_t = r_1$. By varying the parameters α, β, δ, γ and η, and applying the update equations for the cart-pole balancing problem, the different results are obtained and compared in this paper.

C. TD with Value-Function Approximation

The results obtained by using the above RL methods on the cart-pole problem have a major drawback. The assumption of a discretised state-space increases the effect of the value of constant force action on the performance of the system. Also, the constant force must be chosen to ensure that the system changes its state at the end of every time-instant. In other words, the constant force should cause $p(a, s, s) = 0$. A continuous-state MDP is considered in order to overcome these drawbacks [11]. Accordingly, for the CPBP the four state variables are treated as continuous-time variables, $x(t) \in \mathbb{R}^4$.

1) Value-Function Approximation (VFA): With a continuous-state MDP, it is not feasible to update every point in $\mathbb{R}^4$ individually. Further, storing a separate value to represent the quality of each state requires a large memory. To overcome this, an approximate value-function is considered to generalise the values from the states visited by the agent to those states in the neighbourhood that are not visited. Specifically, the parametrised function $\hat{q}(s, a, w)$ with $w \in \mathbb{R}^4$ is used to approximate the state-action value-function $q(s, a)$; this approximation is used in CPBP.

2) Stochastic Gradient Descent: In value-function approximation, the goal of the RL agent is to update the approximate value-function towards the Q-target in case of the control problem. The parameters are updated using the gradient of the following cost function:

$$J(w) = \mathcal{E}\left\{\left(R_{t+1} + \gamma\hat{q}(S_{t+1}, A_{t+1}, w) - \hat{q}(S_t, A_t, w)\right)^2\right\}$$

Clearly, $J(w)$ is a function of the error between the Q-target and the approximate value. The corresponding stochastic gradient is given by

$$\begin{aligned} \Delta w &= -0.5\alpha\nabla_w J(w) \\ &= \alpha(R_{t+1} + \gamma\hat{q}(S_{t+1}, A_{t+1}, w) \\ &\quad -\hat{q}(S_t, A_t, w))\nabla_w\hat{q}(S_t, A_t, w) \end{aligned} \tag{9}$$

The problem is considerably simplified if the approximation is assumed to be linear in the parameters:

$$\hat{q}(s, a, w) = w^T\phi(s, a) \tag{10}$$

Accordingly, the gradient $\nabla_w\hat{q}(S_t, A_t, w) = \phi(S_t, A_t)$. Thus, the parameters are updated using

$$\Delta w = \alpha(R_{t+1} + \gamma\hat{q}(S_{t+1}, A_{t+1}, w) - \hat{q}(S_t, A_t, w))\phi(S_t, A_t) \tag{11}$$

in accordance to the steepest-descent rule to minimise the error between the Q-target and the linear value-function approximation.

The algorithm applied to CPBP can be summarised as follows:

- Initialise $\hat{q}(S, A, w)$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$, w.
- For each step in an episode:
  - Given S_t, let $A_t = \arg\max_a \hat{q}(S_t, a, w)$.
  - Take the action A_t.
  - Observe R_{t+1} and S_{t+1} from the environment.
  - Update the action-value approximation $\hat{q}(S_t, A_t, w)$ to the Q-target using (11).
  - Proceed until the terminal state, where the state S_t exceeds the limits set by the objective.

IV. Swing-up and LQR-based Stabilisation

In this section, we consider an end-to-end control solution to ensure that the pendulum is shifted from the stable equilibrium point $\theta = \pi$ to stabilising it at the unstable equilibrium point $\theta = 0$. This is achieved using a combination of the energy method and a linear quadratic regulator. The control law is switched from the former to the latter when $\theta \in [-12, 12]^\circ$.

A. Swing-up using Energy Control Method

The swing-up strategy [14] is based on (1). The total energy of the pendulum at any state is given by

$$E = 0.5\bar{I}_p\dot{\theta}^2 + mgl(\cos\theta - 1). \tag{12}$$

where $\bar{I}_p = I_p + ml^2$ with $I_p = \frac{4}{3}(M+m)l^2$. We note that a change in energy depends on $\dot{\theta}\cos\theta$. The Lyapunov function $0.5(E - E_0)^2$ results in the control law $u = k(E - E_0)\dot{\theta}\cos\theta$ required to reach the target energy E_0. However, this does

not take into consideration that the cart track length is finite. Accordingly, using the modified Lyapunov function $0.5(E - E_0)^2 + ml\lambda\dot{x}^2$ [15], the control law is as follows:

$$u = k((E - E_0)\dot{\theta}\cos\theta - \lambda\dot{x} \tag{13}$$

where λ is a parameter to restrict the linear motion of the cart.

B. Stabilisation using Linear Quadratic Regulator

When the pendulum position is such that $-12^\circ \leq \theta \leq 12^\circ$, the swing-up control law is switched to a stabilisation controller. Here, we consider an LQR-based stabilising control law determined from (1) linearised about $\theta = 0$; it is well-known that LQR is robust. The linearised model is $\dot{x} = Ax + Bu$, and $y = Cx$, where x is the state-vector defined earlier, and

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{I_p}{M+m+m^2l^2} & \frac{m^2l^2g}{(M+m)I_p+m^2l^2} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{bml}{(M+m)I_p+m^2l^2} & \frac{(M+m)gml}{(M+m)I_p+m^2l^2} & 0 \end{pmatrix},$$

$$B = \begin{pmatrix} 0 \\ \frac{I_p}{M+m+m^2l^2} \\ 0 \\ \frac{-ml}{(M+m)I_p+m^2l^2} \end{pmatrix}, \quad C = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

The state-feedback stabilizing control is $u = -Kx$, where the LQR gain $K = R^{-1}B^TP$ is obtained from the solution of the corresponding Algebraic Riccati Equation (ARE) $A^T + PA - PBR^{-1}B^TP + Q = 0$ for specified $Q = C^TC$ and $R = 1$. For the example system considered here, $K = \begin{pmatrix} -1 & -1.7788 & -26.3106 & -3.8440 \end{pmatrix}$.

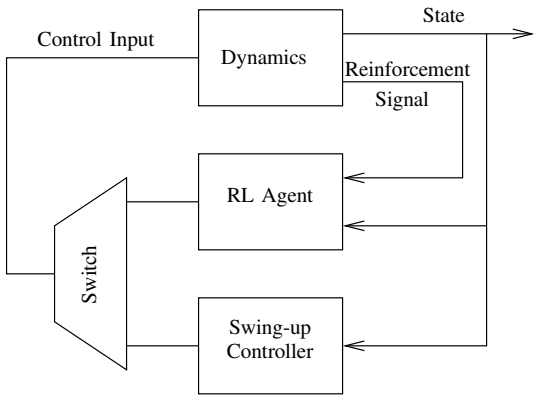

Fig. 1. Combining swing-up controller with RL-based stabiliser.

V. Swing-up and RL-Based Stabilisation

During the process in which an RL agent learns, the pole in CPBP is reset to the unstable equilibrium point ($\theta = 0$) after every failure. When applied to the physical system, this appears both redundant and cumbersome. In contrast, after every failure we allow the pendulum to asymptotically approach the stable equilibrium point ($\theta = \pi$) using its natural damping. When the pendulum is within an ϵ-neighbourhood of this stable point, the swing-up control law described in Section IV is activated to ensure the pole position reaches $\mathcal{O}$. Subsequently, the RL controller attempts to stabilise the pole around $\theta = 0$. This proposed integration of the swing-up and RL controllers is shown in Fig. 1. Such an approach has several other advantages: (i) It allows a comparison of the performance of RL-based and LQR-based controllers. (ii) The learning process of the RL agent is automated and does not require manual intervention to reset the pole position after every failure. (iii) The resultant state after every swing-up action may differ with each failure. This allows greater exploration during the learning process.

VI. Results

The controller for the CPBP is said to have achieved its objective if θ and x reaches and remains in the admissible space $\mathcal{O}$ specified earlier for more than 100,000 steps, where each step is of duration τ.

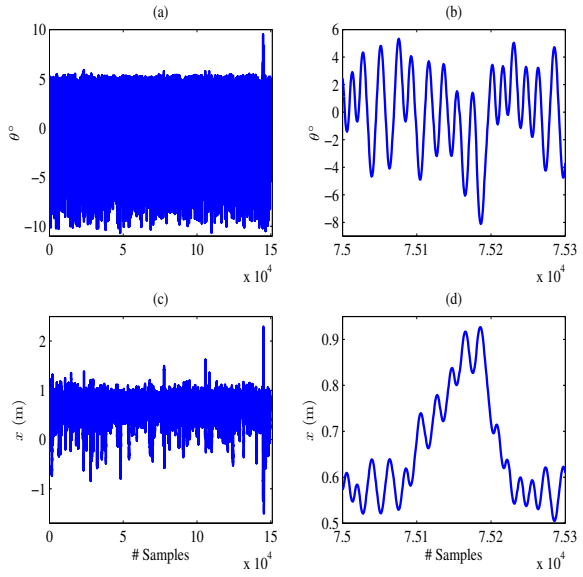

Fig. 2. Q-learning applied to CPBP with $F = \pm 10$ N, $\alpha = 0.5$, and $\lambda = 0.99$. (a) The angle θ that the pole makes with the vertical as a function of time. (b) The variations of θ on a smaller time-frame. (c) The position x of the cart as a function of time. (d) The variations in x on a smaller time-frame.

Off-Policy TD Control Algorithm (Q-learning): We recall that this is a basic RL control algorithm. The results with $F = \pm 10$ N, $\alpha = 0.5$ and $\gamma = 0.99$ are shown in Fig. 2. Q-learning achieves optimal policy; however, it does so after 420 trials. The cart-position x approaches the boundary of $\mathcal{O}$ and the pole-angle θ reaches the boundary of $\mathcal{O}$. Indeed, $\theta \in [-11, 10]^\circ$ and $x \in [-1.5, 2.4]$ m. The performance can be improved with different choices of the parameters; the results are summarised in Table I.

Actor-Critic Policy Gradient Method: The policy gradient actor-critic (PGAC) method involves the calculation of a value-

TABLE I
COMPARISON OF PERFORMANCE.

Algorithm	Force (N)	α	γ	Quantiser	λ_w	λ_v	θ°	x (m)	Episodes
Q-learning	± 10	0.5	0.99	getBox	—	—	$[-11, 10]$	$[-1.5, 2.4]$	420
Q-learning	± 15	0.6	0.99	getBox2	—	—	$[-2.9, 2.9]$	$[-1, 1]$	380
Q-learning	± 30	0.4	0.99	getBox	—	—	$[-3, 6.3]$	$[-0.1, 0.6]$	24
VFA	± 10	0.07	0.99	—	—	—	$[-0.4, 0.3]$	$[-0.062, 0.054]$	19
PGAC	± 10	1000	0.95	—	0.9	0.8	$[-12, 12]$	$[-1.7, 0.2]$	62
Q-learning + Swing-up	± 10	0.5	0.99	getBox	—	—	$[-8.6, 11.5]$	$[-0.6, 0.5]$	430
PGAC + Swing-up	± 10	1000	0.95	getBox	0.8	0.9	$[-11, 11.]$	$[-1.0, 1.7]$	255
VFA + Swing-up	± 10	0.07	0.99	—	—	—	$[-0.4, 0.3]$	$[-0.062, 0.054]$	19

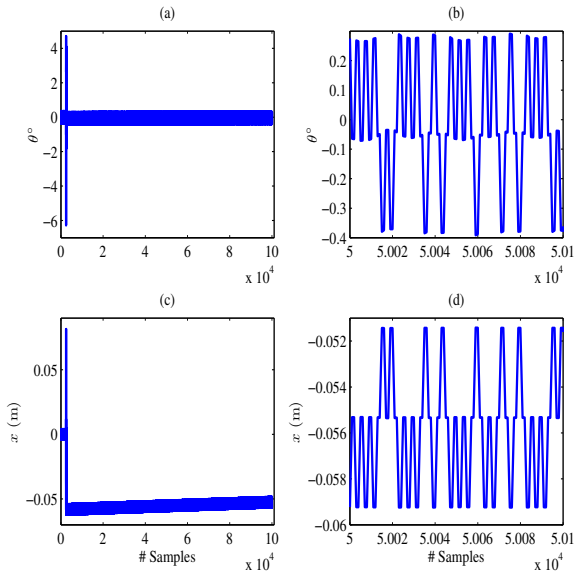

Fig. 3. Actor-critic method applied to CPBP with $F = \pm 10$ N, $\alpha = 1000$, and $\lambda = 0.95$. (a) The angle θ that the pole makes with the vertical as a function of time. (b) The variations of θ on a smaller time-frame. (c) The position x of the cart as a function of time. (d) The variations in x on a smaller time-frame.

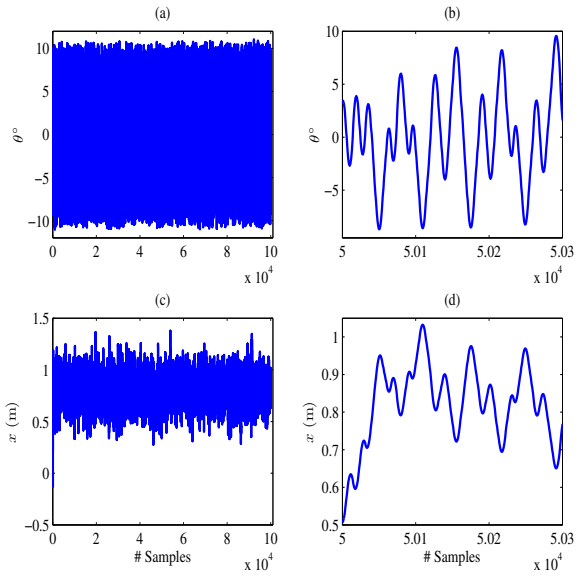

Fig. 4. TD with value-function approximation applied to CPBP with $F = \pm 10$ N, $\alpha = 0.07$, and $\lambda = 0.992$. (a) The angle θ that the pole makes with the vertical as a function of time. (b) The variations of θ on a smaller time-frame. (c) The position x of the cart as a function of time. (d) The variations in x on a smaller time-frame.

function by the critic network as well as the selection of the best action for a given state and state-value function, through a probabilistic approach. The chosen parameters are as follows: $F = \pm 10$ N, $\alpha = 1000$, $\gamma = 0.95$, $\lambda_w = 0.9$, and $\lambda_v = 0.8$. These parameters have been adjusted to achieve optimal policy. From Fig. 3 it is clear that x and $\theta \in \mathcal{O}$; however, the performance is rather poor.

With a comparatively large α, the RL agent is expected to diverge from the optimum behaviour. However, even with $\alpha = 1000$, an optimal policy is achieved albeit with large oscillations in the steady-state implying a large expenditure of control energy. Further, x and θ takes values that cover nearly all of the admissible space. Furthermore, the variations in these state variables are nearly periodic which is perhaps a characteristic of actor-critic methods.

TD with Value-Function Approximation: The results with a linear value-function approximation are shown in Fig. 4 with $F = \pm 10$ N, $\alpha = 0.07$, $\gamma = 0.992$. Evidently, this approach provides rather satisfactory results. Observe that the RL agent achieves the optimal policy in merely 19 episodes, a remarkable improvement over the earlier approaches. Further, the amplitude of oscillations is quite minimal; specifically, $\theta \in [-0.4, 0.3]^\circ$ and $x \in [-0.062, 0.054]$ m. Observe that in the steady-state $x \neq 0$.

Classical Swing-up and LQR Stabilisation: The results of this experiment are shown in Fig. 5(a) and (b). The swing-up controller ensures that the pole is moved from its stable position to the desired position ($\theta = \pi$). This is carried out in a manner such that the cart never hits the boundaries of the rails. This requires several oscillations about the stable position. The parameter λ affects the largest value that $|x|$ can take. Once θ reaches $\mathcal{O}$, the LQR controller stabilises the pole

at the desired position.

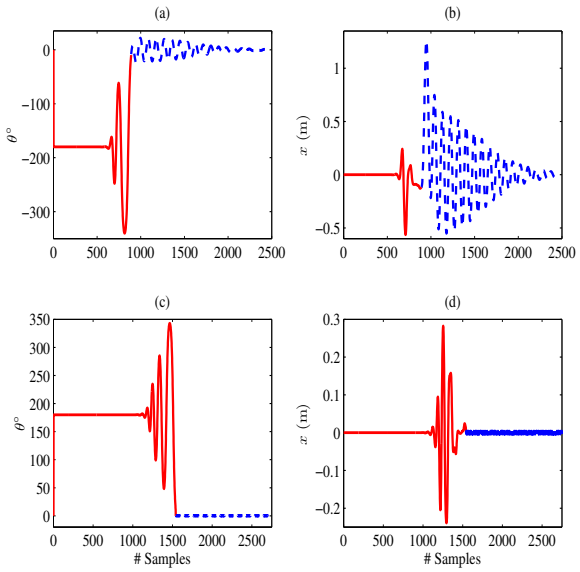

Fig. 5. Performance of the overall controller. (a) and (b): Swing-up with LQR-based stabilisation. (c) and (d): Swing-up with RL-based stabilisation. (The solid lines indicate swing-up control action and broken lines indicate stabilisation.)

Classical Swing-up and RL Stabilisation: The results of the integration of the classical swing-up with the value-function approximation are shown in Fig. 5(c) and (d). Evidently, pole-stabilisation is independent of the swing-up, and the scope for exploration of the RL controller has been increased. Further, the system is now fully automated in that whenever the RL policy fails, the swing-up controller is activated automatically. Discussions: The different algorithms considered in this paper show varied results when applied to CPBP. A large variation in the results is also the effect of the parameters in the implementation of each algorithm. The Q-learning algorithm is selected as the standard for comparing the RL algorithms. Unlike LQR, its effect on the system is discrete and of a bang-bang type control. In contrast to LQR-based stabilisation, RL-based stabilisation results in much smaller transients. PGAC is an improvement over the basic Q-learning. The results show a pattern in the agent's behaviour, suggesting a well-structured input-output relation. The policy-gradient based actor network introduces stochasticity, which is particularly useful for applications with hidden or occluded states, and partially observable environments. But due to the number and sensitivity of the parameters, PGAC is more complex to design. TD with value-function approximation performs the best among the stabilisation algorithms since it uses a continuous MDP. Here, the approximation of the value-function causes a sharp decrease in the sensitivity of the parameters. The addition of the energy-based swing-up controller for the CPBP does not affect the working of an RL algorithm during stabilisation. Results similar to Fig. 5 are obtained with Q-learning and actor-critic policy-gradient method.

VII. Conclusions

The performances of several reinforcement learning algorithms are compared when applied to the cart-pole inverted-pendulum balancing problem. In discrete state-space, the actor-critic policy-gradient method converges faster and provides better stabilisation when compared to the Q-learning approach. The range of angular positions of the pendulum and linear positions of the cart for the optimal policy decreases with the transition from discrete to continuous state-space. Value-function approximation provides the best performance amongst the three RL algorithms considered in this paper. Further, the integration of swing-up using the energy method has not affected the performance of the individual algorithms. In contrast, this integration has achieved automation.

References

[1] G. E. Hinton, "Connectionist learning procedures," *Artificial Intelligence*, vol. 40, no. 1–3, pp. 185–234, September 1989.

[2] C. Zhou and Q. Meng, "Reinforcement learning with fuzzy evaluative feedback for a biped robot," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, San Francisco, CA, USA, April 2000, pp. 3829–3834.

[3] N. Muskinja and B. Tovornik, "Swinging up and stabilization of a real inverted pendulum," *IEEE Transactions on Industrial Electronics*, vol. 53, no. 2, pp. 631–639, April 2006.

[4] L. B. Prasad, B. Tyagi, and H. O. Gupta, "Modelling and simulation for optimal control of nonlinear inverted pendulum dynamical system with disturbance input using PID controller & LQR," in *Proceedings of the Sixth Asia Modelling Symposium (AMS)*, Bali, Indonesia, May 2012.

[5] D. Michie and R. A. Chambers, "Boxes: An experiment in adaptive control," in *Machine Intelligence 2*, E. Dale and D. Michie, Eds. Edinburgh, UK: Oliver and Boyd, 1968, pp. 137–152.

[6] A. G. Barto, R. S. Sutton, and C. W. Anderson, "Neuronlike adaptive elements that can solve difficult learning control problems," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 13, no. 5, pp. 834–846, September-October 1983.

[7] R. S. Sutton, "Temporal credit assignment in reinforcement learning," Ph.D. dissertation, University of Massachusetts Amherst, 1984.

[8] C. W. Anderson, "Learning to control an inverted pendulum using neural networks," *IEEE Control Systems Magazine*, vol. 9, no. 3, pp. 31–37, April 1989.

[9] M. Riedmiller, "Neural fitted Q iteration — First experiences with a data efficient neural reinforcement learning method," in *Machine Learning: European Conference on Machine Learning (ECML)*, J. Gama, R. Camacho, P. B. Brazdil, A. M. Jorge, and L. Torgo, Eds. Berlin, Germany: Springer, 2005, pp. 317–328, Lecture Notes in Computer Science, vol. 3720.

[10] R. S. Sutton, "On the significance of Markov decision process," in *Artificial Neural Networks: International Conference on Artificial Neural Networks (ICANN)*, W. Gerstner, A. Germond, M. Hasler, and J. D. Nicoud, Eds. Berlin, Germany: Springer, 1997, pp. 273–282, Lecture Notes in Computer Science, vol. 1327.

[11] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. Cambridge, MA, USA: MIT Press, 1998.

[12] J. Van der Wal, "Discounted Markov games: Generalized policy iteration method," *Journal of Optimization Theory and Applications*, vol. 25, no. 1, pp. 125–138, May 1978.

[13] E. Greensmith, P. L. Bartlett, and J. Baxter, "Variance reduction techniques for gradient estimates in reinforcement learning," *Journal of Machine Learning Research*, vol. 5, pp. 1471–1530, November 2004.

[14] K. Furuta, M. Yamakita, and S. Kobayashi, "Swing up control of inverted pendulum," in *Proceedings of the International Conference on Industrial Electronics, Control and Instrumentation (IECON)*, Kobe, Japan, October-November 1991, pp. 2193–2198.

[15] J.-H. Yang, S.-Y. Shim, J.-H. Seo, and Y.-S. Lee, "Swing-up control for an inverted pendulum with restricted cart rail length," *International Journal of Control, Automation and Systems*, vol. 7, no. 4, pp. 674–680, August 2009.