International Journal of Fuzzy Logic and Intelligent Systems, vol. 11, no. 4, December 2011, pp. 267−274
http://dx.doi.org/10.5391/IJFIS.2011.11.4.267

Kernel-based actor-critic approach with applications

Baeksuk Chu[1], Keunwoo Jung[2], and Jooyoung Park[2*]

[1] Department of Intelligent Mechanical Engineering, Kumoh National Institute of Technology, Gumi, Gyeongbuk, 730-701, Korea
[2] Department of Control and Instrumentation Engineering, Korea University, Jochiwon, Chungnam, 339-700, Korea

Abstract

Recently, actor-critic methods have drawn significant interests in the area of reinforcement learning, and several algorithms have been studied along the line of the actor-critic strategy. In this paper, we consider a new type of actor-critic algorithms employing the kernel methods, which have recently shown to be very effective tools in the various fields of machine learning, and have performed investigations on combining the actor-critic strategy together with kernel methods. More specifically, this paper studies actor-critic algorithms utilizing the kernel-based least-squares estimation and policy gradient, and in its critic's part, the study uses a sliding-window-based kernel least-squares method, which leads to a fast and efficient value-function-estimation in a nonparametric setting. The applicability of the considered algorithms is illustrated via a robot locomotion problem and a tunnel ventilation control problem.

Key words : reinforcement learning, actor-critic algorithm, kernel methods, least-squares, sliding-windows

1. Introduction

Recently, actor-critic methods have drawn a great deal of interests in the areas of reinforcement learning, and several algorithms have been studied along the line of the actor-critic strategy [1-5]. In particular, this strategy has been successfully applied to the field of control problems [5-7], and is expected to be a very practical alternative in the area since it can deal with control law synthesis via observing only inputs, rewards, and states without having exact knowledge over the systems under consideration. In the actor-critic methods, a separate structure is used to explicitly represent the policy independent of the value functions, and the policy structure is known as the actor, because it is used to select actions, while the part handling value functions is called the critic, because it criticizes the actions made by the actor [8]. As is well-known, dealing with large-scale real-world application problems involves representing the value functions in the critic's part approximately. Since the quality of approximations may influence the performance significantly, to employ the high-quality function approximation methods is an important sub-problem here. Kernel methods are nowadays one of the most active research areas in modern machine learning techniques, and there have been a number of recent advances in various kernel methods including kernel-based regression [9]. In this paper, we consider a new type of actor-critic strategy employing kernel methods, which have recently shown to be very effective in the area of function approximation, and have performed investigations on combining the actor-critic strategy together with kernel methods. More specifically, this paper studies actor-critic algorithms utilizing the kernel-based least-squares estimation and policy gradients, and in their critic part, we use a sliding-window-based kernel least-squares method, which leads to a fast and efficient value-function-estimation in a nonparametric setting. A preliminary work leading to a part of this paper's results was reported in our conference papers [7,10]. The applicability of the considered algorithms is illustrated via locomotion of a two-linked robot arms and a tunnel ventilation control problem.

The remaining parts of this paper are organized as follows: In Section 2, we show how the considered algorithms based on kernel methods and policy gradient can be derived in the framework of general actor-critic approach. Section 3 reports on the applicability of the considered algorithms via a robot locomotion problem and a tunnel ventilation control problem. Finally, in Section 4, concluding remarks are given.

2. Methods using kernel-based least-squares estimation and policy gradient

The reinforcement method is a computational approach for learning from direct interactions with the environment. It learns control policy by maximizing a numerical reward signal provided by the environment [8]. In this paper, we consider an RL approach employing the actor-critic architecture, the kernel-based least-squares estimation, and policy gradient, and apply the considered approach to a robot locomotion problem and a tunnel ventilation control problem.

Manuscript received Aug. 25, 2011; revised Oct. 6, 2011; accepted Oct. 22;
+Corresponding author

In general, the actor-critic methods consider a discounted reward reinforcement learning problem [5, 8] with states $s \in S$, actions $a \in A$, rewards $r \in \Re$, and time steps $t \in \{0,1,2,\cdots\}$, in which the learning agent interacts with the environment. In the framework of RL, environment dynamics is usually characterized by state transition probabilities

$$p(s' \mid s,a) \triangleq Pr(s_{t+1} = s' \mid s_t = s, a_t = a), \tag{1}$$

and expected rewards

$$r(s,a) \triangleq E[r_t \mid s_t = s, a_t = a]. \tag{2}$$

The objective of the learning agent is to pursue a policy that can maximize the discounted sum of the rewards

$$J(\pi) \triangleq E[\sum_{i=0}^{\infty} \gamma^i r_i \mid s_0, \pi], \tag{3}$$

where $\gamma \in (0,1)$ is the discount rate, r_i is the immediate reward observed after the state transition from state s_i to s_{i+1}, s_0 is a designated start state, and π denotes the policy from which actions are chosen. The action-generating policies can be deterministic or stochastic, and when it is stochastic, as in this paper, the policy is generally described by a conditional probability

$$\pi(a \mid s) \triangleq p(a_t = a \mid s_t = s). \tag{4}$$

Note that by introducing the value functions

$$V^{\pi}(s) \triangleq E[\sum_{i=0}^{\infty} \gamma^i r_{t+i} \mid s_t = s, \pi] \tag{5}$$

and

$$Q^{\pi}(s,a) \triangleq E[\sum_{i=0}^{\infty} \gamma^i r_{t+i} \mid s_t = s, a_t = a, \pi], \tag{6}$$

one can rewrite the objective function in the following form (Dealing with continuous states and actions would require the corresponding summations changed into integral representation. Throughout this paper, the summation representation is used for notational simplicity):

$$J(\pi) = V^{\pi}(s_0) = \sum_s d^{\pi}(s) \sum_a \pi(a \mid s) r(s,a), \tag{7}$$

where $d^{\pi}(s)$ is the discounted state distribution [13] defined by

$$d^{\pi}(s) \triangleq \sum_{i=0}^{\infty} \gamma^i Pr(s_i = s \mid s_0, \pi). \tag{8}$$

In this paper, we consider a new type of actor-critic method utilizing the kernel-based least-squares estimation and policy gradient. In the critic part, we use a sliding-window-based kernel least-squares method for fast and efficient value-function-estimation in a nonparametric setting. In the following, we explain about the critic part and the actor part of the considered approach in detail.

First, we consider the use of kernel-based LS estimation for the critic. As mentioned before, the essence of the actor-critic method is the use of separate parameterized families for the actor part, which is represented by the policy distribution $\pi_{\theta}(a \mid s)$, and the critic part, which is represented by value functions. For the parameterized families of the critic part, this paper considers a general class of functions derived from the form

$$\tilde{V}_v(s) \triangleq \phi^T(s) v, \tag{9}$$

which approximates the state value function $V^{\pi_\theta}(s)$ for action-generating policy π_θ. Also, to train the critic part, we utilize a strategy combining the kernel method [9] with the LSTD (least-squares temporal difference) method [11]. From the Bellman equation [8],

$$V^{\pi_\theta}(s) = \sum_a \pi(a \mid s) \left(r(s,a) + \gamma \sum_{s'} p(s' \mid s,a) V^{\pi_\theta}(s') \right), \tag{10}$$

one can see that through a sampled trajectory, $V^{\pi_{\theta_i}}(s_i)$ can be approximated by $r_i + \gamma V^{\pi_{\theta_i}}(s_{i+1})$; thus $r_i + \gamma \tilde{V}_v(s_{i+1})$ is a valid estimate for $V^{\pi_{\theta_i}}(s_i)$ when v is properly chosen. Applying the LSTD method [11] to the approximation of the state value functions on sliding-windows, one can see that in order for the approximator $\tilde{V}_v(s)$ to be useful at the t-th time step (where $t \geq N-1$), it is desirable to achieve the following:

$$\begin{aligned} &\tilde{V}_v(s_i) \approx r_i + \gamma \tilde{V}_v(s_{i+1}), i.e., \\ &\left(\phi^T(s_i) - \gamma \phi^T(s_{i+1})\right) v \approx r_i \quad \text{for } i = t-N+1, \cdots, t, \end{aligned} \tag{11}$$

where N is the width of a sliding-window. In this least-squares estimation, the sliding-window technique is applied to fix the size of the data, so that the algorithm can operate online in time-varying environments [12]. Here the number of computations required per new sample does not increase as the number of samples increases. Therefore, the complexity of the algorithm is reduced by considering only the observations in a window of fixed length. Since a recent observation should be emphasized more, the so-called forgetting factor $\beta \in (0,1)$ should be used. In this case, the following equation should be used instead of Eq. (11):

$$\beta^{(t-i)} \left[\left(\phi^T(s_i) - \gamma \phi^T(s_{i+1}) \right) v \right] \approx \beta^{(t-i)} r_i, \tag{12}$$

where $i = t-N+1, \cdots, t$.

In order to make rich functional expressions with $\phi(s)^T v$ in Eq. (9), this paper requires that $\phi : S \to F$ be chosen as the feature map associated with a Mercer kernel $k : S \times S \to R$ (For details on Mercer kernels, refer to books on kernel methods, *e.g.*, [9]). Kernel methods are based on the implicit nonlinear transformation of the data from the input space to a high-dimensional feature space. Note that the approximation of state value functions in a reproducing kernel Hilbert space gives a rigorous extension of the conventional approximation techniques for nonparametric settings. In particular, we restrict our attention to the most popular Gaussian Mercer kernel k; thus, the following "kernel trick" holds true [9]:

$$\begin{aligned} < \phi(x), \phi(z) > &= \phi(x)^T \phi(z) = k(x,z) \\ &= \exp\left(-\frac{\| x - z \|^2}{2\sigma^2} \right) \end{aligned} \tag{13}$$

where scalar products in the feature space can be seen as nonlinear (kernel) functions of the data in the input space. Since this property avoids explicit mapping to the feature space, any conventional scalar-product-based algorithm can be used in the feature space by solely replacing the scalar products with the Mercer kernel function in the input space [9]. According to

the representer theorem of [9], the optimal solution of Eq. (12) can be conveniently represented by a linear combination of the relevant feature vectors, i.e.,

$$v_t^* \triangleq \sum_{i=t-N+1}^{t+1} \phi(s_i)\alpha_i, \tag{14}$$

where s_i are the observed states and α_i are the corresponding coefficients. Thus, the approximation problem, Eq. (12), can now be expressed as

$$\begin{aligned}&\beta^{(t-i)}\left[\left(\phi^T(s_i)-\gamma\phi^T(s_{i+1})\right)\left(\sum_{i=t-N+1}^{t+1}\phi(s_i)\alpha_i\right)\right] \\ &\approx \beta^{(t-i)} r_i\end{aligned} \tag{15}$$

and all instances in the sliding window correspond to the following compact form:

$$DH\Phi_t^T\Phi_t A_t \approx DR_t, \tag{16}$$

where

$$D \triangleq \mathrm{diag}\{\beta^{N-1},\ldots,\beta^0\} \in R^{N\times N}, \tag{17}$$

$$H \triangleq \begin{bmatrix} 1 & -\gamma & 0 & \cdots & 0 \\ 0 & 1 & -\gamma & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & -\gamma \end{bmatrix} \in R^{N\times(N+1)}, \tag{18}$$

$$\Phi_t \triangleq [\phi(s_{t-N+1})\cdots\phi(s_{t+1})], \tag{19}$$

$$A_t \triangleq [\alpha_{t-N+1} \ \cdots \ \alpha_{t+1}]^T \in R^{(N+1)\times 1}, \tag{20}$$

and

$$R_t \triangleq [r_{t-N+1} \ \cdots \ r_t]^T \in R^{N\times 1}. \tag{21}$$

Here, note that based on the notations in Eqs. (19) and (20), the optimal solution v_t^* of Eq. (14) can be written more compactly as

$$v_t^* \triangleq \Phi_t A_t. \tag{22}$$

From the kernel trick, Eq. (13), $\Phi_t^T\Phi_t$ in Eq. (16) can be computed easily on the state space as follows:

$$\begin{aligned}\Phi_t^T\Phi_t &= \begin{bmatrix}\phi^T(s_{t-N+1})\\ \vdots \\ \phi^T(s_{t+1})\end{bmatrix}\begin{bmatrix}\phi(s_{t-N+1}) & \cdots & \phi(s_{t+1})\end{bmatrix} \\ &= \begin{bmatrix} k(s_{t-N+1},s_{t-N+1}) & \cdots & k(s_{t-N+1},s_{t+1}) \\ \vdots & \ddots & \vdots \\ k(s_{t+1},s_{t-N+1}) & \cdots & k(s_{t+1},s_{t+1})\end{bmatrix}\end{aligned} \tag{23}$$

which will be called the kernel matrix, K_t. Note that diagonal entries of K_t are all equal to one, because k is Gaussian. Also, note that kernel matrix K_t can be constructed by removing the first row and column of K_{t-1}, and the resultant matrix is referred to as $\hat{K}_{t-1}$. Then, the kernel values on the new data are inserted into the last row and column, i.e.,

$$K_t = \begin{bmatrix} \hat{K}_{t-1} & k_{t-1}(s_{t+1}) \\ k_{t-1}^T(s_{t+1}) & 1 \end{bmatrix} \tag{24}$$

where $k_{t-1}(s_{t+1}) \triangleq [k(s_{t-N+1},s_{t+1}) \ \cdots \ k(s_t,s_{t+1})]^T$. From Eqs. (9), (13), (14), (16), and (23), an estimation of the state value function useful at the t- th time step is

$$\begin{aligned}\tilde{V}_{v_t^*}(s) &= \phi^T(s)v_t^* = \sum_{i=t-N+1}^{t+1}\alpha_i\phi^T(s_i)\phi(s) \\ &= \sum_{i=t-N+1}^{t+1}\alpha_i k(s_i,s),\end{aligned} \tag{25}$$

and the least-squares solution for the coefficients of the above kernel expression is

$$\begin{aligned}A_t &= [\alpha_{t-N+1} \ \cdots \ \alpha_{t+1}]^T \\ &= (K_t H^T D^2 H K_t)^{-1} K_t^T H^T D^2 R_t.\end{aligned} \tag{26}$$

In this paper, we utilize the following to avoid numerical instabilities in the regression:

$$\begin{aligned}A_t &= [\alpha_{t-N+1} \ \cdots \ \alpha_{t+1}]^T \\ &= (\varepsilon I + K_t H^T D^2 H K_t)^{-1} K_t^T H^T D^2 R_t,\end{aligned} \tag{27}$$

where ε is a small positive number added to ensure matrix nonsigularity. Finally, note that the resultant state value function approximator

$$\tilde{V}_{v_t^*}(s) = \phi^T(s)v_t^* = \sum_{i=t-N+1}^{t+1}\alpha_i k(s_i,s) \tag{28}$$

will play an important role in the update process for the actor part.

As an alternative to our main approach based on Eq. (27), a modification that will lead to a recursive solution is also proposed. In the modification, the least-square solution, Eq. (27), is replaced by the following:

$$\begin{aligned}A_t &= [\alpha_{t-N+1} \ \cdots \ \alpha_{t+1}]^T \\ &= K_t^{-1}(DH)^* DR_t,\end{aligned} \tag{29}$$

where $(DH)^*$ means the pseudo-inverse of DH. Eq. (29) may be viewed as the solution resulting from a two-stage solver for the approximation problem, Eq. (16). In its first stage, the least-squares problem

$$(DH)Y_t \approx DR_t \tag{30}$$

is considered, where Y_t is the intermediary vector defined by

$$Y_t = K_t A_t, \tag{31}$$

and its solution is obviously

$$Y_t = (DH)^* DR_t. \tag{32}$$

Then its second stage solves the intermediary equation, Eq. (31), to obtain the solution, Eq. (29). One of the merits of this alternative approach is that K_t^{-1} in Eq. (29) can be computed recursively. To clarify, note that with

$$K_{t-1} \triangleq \begin{bmatrix} 1 & b^T \\ b & B \end{bmatrix} \text{ and } K_{t-1}^{-1} \triangleq \begin{bmatrix} e & f^T \\ f & F \end{bmatrix}, \tag{33}$$

in Eq. (24), its inverse can be written as follows [12]:

$$\hat{K}_{t-1} = B, \tag{34}$$

$$\hat{K}_{t-1}^{-1} \triangleq F - ff^T/e. \tag{35}$$

Also, note that from the matrix inversion formula, the inverse of the kernel matrix K_t can be obtained by the following:

$$K_t^{-1} = \begin{bmatrix} \hat{K}_{t-1}^{-1}G & -\hat{K}_{t-1}^{-1}k_{t-1}(s_{t+1})g \\ -\left(\hat{K}_{t-1}^{-1}k_{t-1}(s_{t+1})\right)^T g & g \end{bmatrix}, \tag{36}$$

where

$$G \triangleq I + k_{t-1}(s_{t+1})k_{t-1}^{T}(s_{t+1})\hat{K}_{t-1}^{-1}g, \tag{37}$$

and

$$g \triangleq \left(1 - k_{t-1}(s_{t+1})\hat{K}_{t-1}^{-1}k_{t-1}(s_{t+1})\right)^{-1}. \tag{38}$$

Eqs. (33) to (38) show that K_t^{-1} can be indeed computed recursively based on K_{t-1}^{-1} and $k_{t-1}(s_{t+1})$.

In the following, we describe the actor trained via policy gradient [5, 13] in details for readers' convenience. The main role of the actor is to generate actions via a parameterized family. At each state $s \in S$, an action $a \in A$ is drawn in accordance with the conditional distribution $\pi_\theta(a \mid s)$, where θ is the parameter vector characterizing the distribution. Thus, the objective to maximize can be written as follows:

$$J(\pi) = J(\theta) = \sum_s d^{\pi_\theta}(s) \sum_a \pi_\theta(a \mid s) r(s,a) \tag{39}$$

One of the convenient strategies for seeking the best distribution parameter θ is to utilize the direction of $\nabla_\theta J(\theta)$, which is often called the policy gradient. According to the famous policy gradient theorem [13], the policy gradient can be written as follows:

$$\nabla_\theta J(\theta) = \sum_a \pi_\theta(a \mid s) \nabla_\theta \log \pi_\theta(a \mid s) \left(Q^{\pi_\theta}(s,a) - V^{\pi_\theta}(s) \right). \tag{40}$$

From the Bellman equation [8]

$$Q^{\pi_\theta}(s,a) = r(s,a) + \gamma \sum_{s'} p(s' \mid s,a) V^{\pi_\theta}(s'), \tag{41}$$

we see that through a sampled trajectory, $Q^{\pi_{\theta_k}}(s_k, a_k)$ can be approximated by $r_k + \gamma V^{\pi_{\theta_k}}(s_{k+1})$; thus $r_k + \gamma \tilde{V}_{v_k}(s_{k+1})$ and $\tilde{V}_{v_k}(s_k)$ are valid estimates for $Q^{\pi_{\theta_k}}(s_k, a_k)$ and $V^{\pi_{\theta_k}}(s_k)$, respectively. Hence, the approximation step via the sampled trajectory yields the following estimate:

$$\begin{aligned}
&[\nabla_\theta J(\theta)]_{\theta=\theta_t} \\
&= \left[\sum_s d^{\pi_\theta}(s) \sum_a \pi_\theta(a \mid s) \nabla_\theta \log \pi_\theta(a \mid s) \right. \\
&\qquad \left. \times \left(Q^{\pi_\theta}(s,a) - V^{\pi_\theta}(s) \right) \right]_{\theta=\theta_t} \\
&\approx \left(Q^{\pi_{\theta_t}}(s_t,a_t) - V^{\pi_{\theta_t}}(s_t) \right) \left[\nabla_\theta \log \pi_\theta(a_t \mid s_t) \right]_{\theta=\theta_t} \\
&\approx \left(r_t + \gamma \tilde{V}_{v_t}(s_{t+1}) - \tilde{V}_{v_t}(s_t) \right) \left[\nabla_\theta \log \pi_\theta(a_t \mid s_t) \right]_{\theta=\theta_t}.
\end{aligned} \tag{42}$$

Therefore, one can update the actor parameter θ via the following simple gradient-based rule:

$$\begin{aligned}
\theta_{t+1} &\leftarrow \theta_t + \alpha(\widetilde{TD}_t)[\nabla_\theta \log \pi_\theta(a_t \mid s_t)]_{\theta=\theta_t} \\
&\approx \theta_t + \alpha[\nabla_\theta J(\theta)]_{\theta=\theta_t},
\end{aligned} \tag{43}$$

in which $\alpha > 0$ is the learning rate, and $\widetilde{TD}_t$ is the so-called temporal difference [8] computed by

$$\widetilde{TD}_t \triangleq r_t + \gamma \tilde{V}_{v_t}(s_{t+1}) - \tilde{V}_{v_t}(s_t). \tag{44}$$

As a conditional probability density function for the continuous input space, one can consider the normal distribution. In this case, a normal distribution is generally employed as the density $\pi_\theta(a \mid s)$ that governs the control selection. Then the actual output of the stochastic unit can be set as:

$$a \sim N(k_\theta(s), \sigma_a^{\,2}) \tag{45}$$

where a is a Gaussian random variable with a density function expressed as:

$$\pi_\theta(a \mid s) = \frac{1}{\sigma_a \sqrt{2\pi}} \exp\left\{ \frac{-(a - k_\theta(s))^2}{2\sigma_a^{\,2}} \right\}. \tag{46}$$

When the application domain concerns the discrete action space, one can consider the Gibbs policy [13] instead.

3. Applications

In this section, we address the application of the considered kernel-based actor-critic approach to two control problems. The first problem deals with locomotion of a robot arm [14]. More specifically, this problem considers a planar two-linked manipulator in a gravitational environment. The mission of the robot is to move forward as fast as possible, without knowing the environment in advance. So, the agent is required to find out an efficient policy based on the observed agent-environment interactions. The immediate reward for this problem is defined as the distance that the body of the robot moves forward in the current step. At the t-th time step, the agent reads the normalized joint angles obtained via a simple piecewise linear scaling, and outputs a binary action value $(+1 \text{ or } -1)$ for each joint, which indicates turning direction of the joint motor, according to its stochastic policy. For more details on the problem, one may refer to specifications in [7, 10]. The approach of this paper was applied to the robot for sufficiently many time steps. For each parameter setting, 20 independent trials were tested, and in each trial, its initial state of the robot was set such that all links are aligned horizontally. In order to investigate the robustness of the kernel-based actor-critic approach, we performed experiments for a range of window size (N). Results reported in Fig. 1 show that the considered algorithms yields robust performance around the range used. In other words, performance tends to remain the same as long as the window size is sufficiently large.

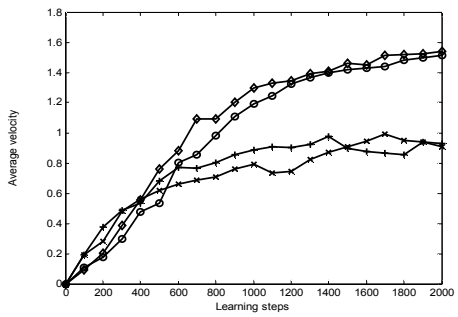

Fig. 1. The average performance of 20 trials on varying window size (N). The main approach with $N = 15$ case (denoted 'o'), the main approach with $N = 25$ case (denoted '◇'), the alternative approach with $N = 15$ case (denoted '+'), and the alternative approach with $N = 25$ case (denoted 'x').

As our second application domain, we considered a tunnel ventilation control problem [15]. This problem focuses on

adjusting the CO pollutant level inside the tunnel as one of the control targets. The pollutant source setting the pollutant level inside the tunnel is estimated by using actual traffic data collected from a real tunnel system, the Dunnae Tunnel located on Youngdong Highway in Korea, as the linear combination of the number of vehicles and the pollutant emission rates according to the vehicle types [16]. The actor-critic algorithms considered in this paper determines the optimal number of operational jet-fans with respect to the two aforementioned control purposes. The reward for this problem consists of a combination of two control objectives: maintaining an adequate level of pollutants and minimizing electrical power consumption. As such, in Eq. (47), the pollutant level over an allowable limit and the energy consumption proportional to the number of running jet-fans are combined with an appropriate weighting factor, K :

$$r = \begin{cases} -\left\{(CO_{current} - CO_{ref}) + K \cdot E_{JF}\right\}, & if\ CO_{current} > CO_{ref} \\ -E_{JF}, & if\ CO_{current} < CO_{ref} \end{cases} \quad (47)$$

- CO_{ref} : the allowable reference CO pollutant level, chosen to be 25ppm in this study
- $CO_{current}$: the current CO pollutant level
- E_{JF} : the electrical energy consumption by the operation of jet-fans

In Eq. (47), K is manually selected according the objective considered to be relatively critical. For the details on the system parameters, one may refer to [15]. The suggested algorithms utilize the following parameters for the tunnel ventilation control simulations: Learning rate $\alpha = 0.3$; Discount rate $\gamma = 0.5$; Kernel width $\sigma = 0.3$; Standard deviation of the normal distribution $\sigma_a = 2.5$; Window size $N = 60$; Forgetting factor $\beta = 0.999$; Initial actor parameter vector $\theta_{initial} = [0 \ \cdots \ 0]^T$; Reward weighting factor $K = 0.045$. Simulations for the second application problem were implemented with each unit time step equal to one minute. In a real tunnel ventilation system, the jet-fans should not be turned on and off very often to assure a reasonably long lifetime of the jet-fans. Another reason for the restricted switching on-off of the jet-fans is that the start-up current is higher than normal operation current. Specifically, when a jet-fan begins to operate, for the first one minute, about 30% of the total electric power is additionally required. Therefore, in this study, a jet-fan-alternating algorithm is adopted to maximize the lifetime of the jet-fans; the algorithm evenly uses all the jet-fans and prevents frequent switching of each jet-fan [17]. Fig. 2 shows the 3-D plot of the pollutant distribution for the 'uncontrolled case' for the last 50 time steps of all 5000 time steps. The 'uncontrolled case' is defined as when only the nominal number of jet-fans, which is chosen as 15 out of the 32 total jet-fans, is constantly operated. In this case, the pollutant source, which is the only input for setting the pollutant distribution in the tunnel, is calculated from traffic volume information measured in the real tunnel. In this case, the maximum CO pollutant level considerably fluctuates to the allowable level, 25ppm.

The same traffic data as that of the uncontrolled case was used to perform the simulations for the 'controlled case'. However, in this case, instead of using the nominal number of jet-fans, the control inputs generated by the considered actor-critic algorithm were applied to the tunnel ventilation system. Fig. 3 presents the performance of a learning controller trained by the suggested approach through a sample case. With respect to the two control objectives expressed by the reward formulation, the result of the simulation can be explained as follows. First, when the CO pollutant level exceeded the allowable limit, 25ppm, the controller increased the number of operational jet-fans to maintain the CO level under the limit value; this is the first control objective. On the other hand, if the CO pollutant was maintained well below the allowable limit and an excessive amount of energy was consumed by the operating jet-fans unnecessarily, the controller decreased the number of operational jet-fans and conserved electricity; this is the second control objective. After sufficient time, which was approximately 2500 time steps in this sample case, had passed for learning, as shown in Fig. 4, the CO pollutant level along the time axis stayed near the allowable limit. This result demonstrates that the approach of this paper successfully satisfied two goals of the tunnel ventilation control.

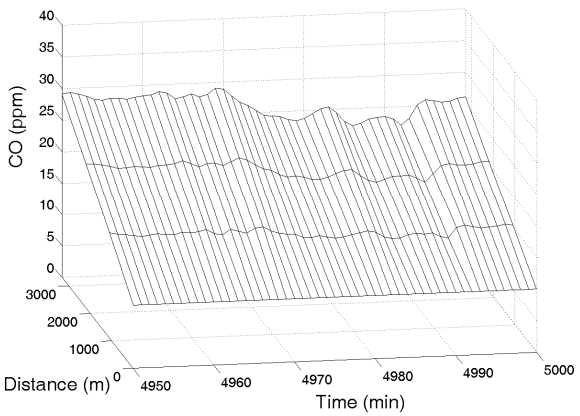

Fig. 2 3-D plot of the pollutant distribution for the last 50 time steps for the 'uncontrolled case' (CO vs. time and longitudinal distance along tunnel)

To quantitatively evaluate the performance of the control algorithm, the standard deviation of the CO peak values was calculated every 500 steps during the learning process composed of total 5000 steps. A low standard deviation of CO peak values in the vicinity of the allowable limit, 25ppm, translates to the prevention of excessive pollutant levels and reduction of unnecessary energy consumption, simultaneously. In Fig. 5, the standard deviations of three cases controlled by three different controllers are compared with each other: (i) Controller 1: a previously developed RL-based control algorithm by Chu et al. [15]; (ii) Controller 2: an actor-critic

control algorithm using kernel-based LS estimation (alternative algorithm utilizing Eq.(29)); (iii) Controller 3: an actor-critic control algorithm using kernel-based LS estimation (main algorithm utilizing Eq. (27)).

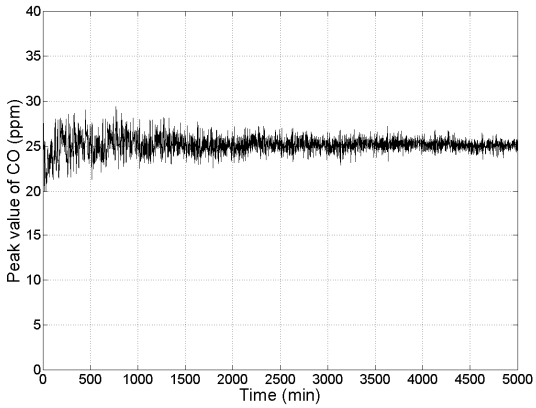

Fig. 3 Peak value of CO for the all 5000 time steps for the 'controlled case by the considered RL-based controller (main algorithm based on Eq. (27))'

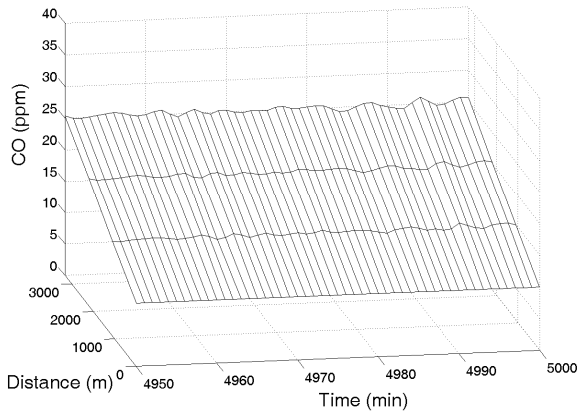

Fig. 4 3-D plot of the pollutant distribution for the last 50 time steps for the 'controlled case by the considered RL-based controller (main algorithm based on Eq. (27))' (CO vs. time and longitudinal distance along tunnel)

To increase the credibility of the evaluation, the standard deviations of the three cases were averaged with 10 episodic tasks. The standard deviations of all the three controllers show a decreasing trend with time; this means that the controllers were designed to basically pursue the control objectives. Note that the two controllers using kernel-based LS estimation show a superior performance to the previously developed RL-based control algorithm. A non-parametric setting based on kernel methods enables a more rigorous formulation of the approximation of the state value function. Therefore, the quality of approximation was increased, and the kernel-based controllers attained better performances than the previously developed RL-based controller. In the meantime, the controller using the alternative algorithm depended on how accurately the two-stage solution, Eqs. (29) to (38), could approximate the true least-squares solution. The inaccuracy resulting from the sophistication of the alternative algorithm lowered the quality of approximation of the state value function and the control performance below that of the controller using the main algorithm based on Eq. (27). In Table 1, four controllers, including the uncontrolled case, are compared with respect to the mean value, standard deviation, maximum/minimum value of the peak CO level and the consumed energy during the last 500 time steps averaged with 10 episodic tasks. While the four cases have similar mean values for the CO level, the maximum CO levels and consumed energies are lower in the controlled cases. These results mean that the controllers employing RL methodology were designed to satisfy the two control purposes well. In the comparison of the three controlled cases, the actor-critic control algorithm using kernel-based exact LS estimation (main algorithm) approached the lowest standard deviation as the learning progressed, indicating the best control performance.

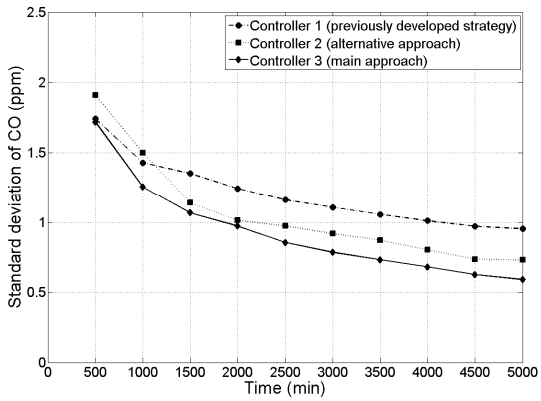

Fig. 5 Standard deviations averaged with 10 episodic tasks for the considered controllers (main algorithm and alternative algorithm) and a previously developed RL-based controller

Table 1 Mean, standard deviation, maximum/minimum value of peak CO level and consumed energy during the last 500 time steps averaged with 10 episodic tasks

Case	CO level (ppm)				Energy (kWh)
	CO_{mean}	CO_{std}	CO_{max}	CO_{min}	
Uncontrolled case	24.48	2.23	29.71	19.78	3750
Controller 1	24.34	0.95	27.42	21.87	3617
Controller 2	24.99	0.73	27.15	22.42	3445
Controller 3	25.03	0.59	25.45	23.54	3428

4. Concluding remarks

In this paper, we studied on a new type of actor-critic strategy in which the critic part is trained by kernel-based least-squares estimation and the actor part is trained via policy gradient, and applied the strategy to two control problems.

Updating the approximate state value function via kernel methods is the key ingredient of the studied strategy, and it was efficiently implemented utilizing the concept of sliding windows. Algorithms resultant from the strategy consist of the main approach based on the exact solution and the alternative one based on an approximate recursive solution. The applicability of the considered algorithms was illustrated with good performance via tunnel ventilation control and locomotion of a two-linked robot arms. One of the future topics that may be considered along the line of extending this paper is to address the robustness issue for the stochastic and/or uncertain systems.

5. Acknowledgment

This research was supported in part by the MKE(The Ministry of Knowledge Economy), Korea, under the Human Resources Development Program for Convergence Robot Specialists support program supervised by the NIPA(National IT Industry Promotion Agency) (NIPA-2011-C7000-1001-0005)

References

[1] A. G. Barto, R. S. Sutton, C. W. Anderson, "Neuronlike elements that can solve difficult learning control problems," *IEEE Transactions on Systems Man and Cybernetics*, vol. 13, pp. 835-846, 1983,.

[2] H. R. Berenji, D. Vengerov, "A convergent actor-critic-based FRL algorithm with application to power management of wireless transmitters," *IEEE Tranactions on Fuzzy Systems*, vol. 11, pp. 478-485, 2003,.

[3] H. Kimura, S. Kobayashi, "An analysis of actor/critic algorithms using eligibility traces: Reinforcement learning with imperfect value function," In *Proceedings of the Fifteenth International Conference on Machine Learning*, pp. 111-116, 1998.

[4] J. Park, J. Kim, D. Kang, "An RLS-based natural actor-critic algorithm for locomotion of a two-linked robot arm," *Lecture Notes in Artificial Intelligence*, vol. 3801, pp. 65-72, 2005.

[5] J. Peters, S. Vijayakumar, S. Schaal, "Reinforcement learning for humanoid robotics," In *Proceedings of the Third IEEE-RAS International Conference on Humanoid Robots (Humanoids2003)*, 2003.

[6] P. Thomas, M. Branicky, N. Kobori, K. Suzuki, P. Hartono, S. Hashimoto, "Learning to control a joint driven double inverted pendulum using nested actor/critic algorithm," In *Proceedings of the 9th International Conference on Neural Information Processing*, 2002.

[7] J. Park, D. Kang, J. Lee, D. Nam, "An actor-critic algorithm using kernel-based least-squares estimation: An application to robot locomotion," In *Proceedings of 2009 CACS International Automatic Control Conference*, 2009.

[8] R. S. Sutton, A. G. Barto, *Reinforcement Learning: an Introduction*, MIT Press, Cambridge, 1998.

[9] B. Schölkopf, A. J. Smola, *Learning with Kernels*, MIT Press, Cambridge, 2002.

[10] J. Park, D. Nam, J. Lee, "Some observations on kernel-based function approximation steps for actor-critic methods," In *Proceedings of KIIS Fall Conference*, vol. 19, no. 2, pp. 79-82, 2009.

[11] J. A. Boyan, "Technical update: Least-squares temporal difference learning," *Machine Learning*, vol. 49, pp. 233-246, 2002.

[12] S. V. Vaerenbergh, J. Vıa, I. Santamarıa, "Nonlinear system identification using a new sliding-window kernel RLS algorithm," *Journal of Communications*, vol. 2, no. 3, pp. 1-8, 2007.

[13] R. S. Sutton, D. McAllester, S. Singh, Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," *Advances in Neural Information Processing Systems*, vol. 12, pp. 1057–1063, 1999.

[14] H. Kimura, K. Miyazaki, S. Kobayashi, "Reinforcement learning in POMDPs with function approximation," In *Proceedings of the Fourteenth International Conference on Machine Learning*, pp. 152-160, 1997.

[15] B. Chu, D. Kim, D. Hong, J. Park, J. T. Chung, T.-H. Kim, "Tunnel ventilation control using reinforcement learning methodology," *JSME International Series C*, vol. 47, no. 4, pp. 939–945, 2006.

[16] D. Hong, B. Chu, W. D. Kim, J. T. Chung, T.-H. Kim, "Pollution level estimation for tunnel ventilation," JSME International Series B, vol. 46, no. 2, pp. 278–286, 2003.

[17] D. Kim, B. Chu, D. Hong, J. T. Chung, T.-H. Kim, "Design of alternating operation algorithm for tunnel ventilation systems," In *Proceedings of the Society of Air-conditioning and Refrigerating Engineering of Korea 2005 Summer Conference*, pp. 872–877, 2005.

Baeksuk Chu received his B.S., M.S., and Ph.D. degrees in Mechanical Engineering from Korea University in 1999, 2001 and 2006, respectively. He joined Kumoh National University in 2001, where he is currently a full-time lecturer in the Department of Intelligent Mechanical Engineering. His current research interests include in the area of robotics, mechatronics, intelligent control and reinforcement learning. (E-mail: bschu@kumoh.ac.kr)

Keunwoo Jung received his B.S. degree in Control and Instrumentation Engineering from Korea University in 2010. He is currently a M.S. candidate of Department of Control and Instrumentation Engineering in Korea University. His current research interests include intelligent control and

reinforcement learning. (E-mail: rmsdn@korea.ac.kr)

Jooyoung Park received his B.S. degree in Electrical Engineering from Seoul National University in 1983 and his Ph.D. degree in Electrical and Computer Engineering from the University of Texas at Austin in 1992. He joined Korea University in 1993, where he is currently a Professor in the Department of Control and Instrumentation Engineering. His current research interests are in the area of kernel methods, reinforcement learning, and control theory. (E-mail: parkj@korea.ac.kr)