Optimal Control Using Adaptive Resonance Theory and Q-Learning

Bahare Kiumarsi [a] , Bakur AlQaudi [b,c], Hamidreza Modares [a] , Frank L. Lewis [b,d], Daniel S. Levine [e]

[a] Michigan State University, East Lansing, MI 48824 USA

[b] The UTA Research Institute UTARI, University of Texas at Arlington, Arlington, TX 76118 USA

[c] Electronics and Instrumentation Engineering Technology, Yanbu Industrial College, 41912, Yanbu AlSinaiyah, K.S.A.

[d] Consulting Professor, State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang, China

[e] Department of Psychology, University of Texas at Arlington, Arlington, TX 76019, USA

Abstract

Motivated by recent advancement in neurocognitive in brain modeling research, a multiple model-based Q-learning structure is proposed for optimal tracking control problem of time-varying discrete-time systems. This is achieved by utilizing a multiple-model scheme combined with adaptive resonance theory (ART). The ART algorithm generates sub-models based on the match-based clustering method. A responsibility signal governs the likelihood of contribution of each sub-model to the Q-function. The Q-function is learned using the batch least-square algorithm. Simulation results are added to show the performance and the effectiveness of the overall proposed control method.

Keywords: Neural Network, Reinforcement learning, Adaptive algorithms, Multiple Model Control.

1. Introduction

The tracking control problem has gained significant attention in the control system community, due to its numerous applications. The objective is to design a control law to ensure stability of the control systems, as well as track a desired reference trajectory in an optimal fashion, by minimizing a performance function. Traditional solutions to the optimal tracking problem aim at finding two components [1], namely, a feedback term obtained by solving a Hamilton-Jacobi-Bellman (HJB) equation and a feedforward term obtained by solving a noncausal difference equation [2]-[3]. The feedback and feedforward terms are mainly found separately, and the solution is commonly obtained in an offline fashion, which requires complete knowledge of the system dynamics. Reinforcement learning (RL) [4-7], as a class of machine learning methods, has been widely used to find the online solution to the optimal tracking problem of time-invariant discrete-time systems. RL algorithms mainly use neural networks (NNs) to approximate the value function and consequently find the optimal control solution. This application of NNs essentially extends traditional adaptive control capabilities to more advanced optimal adaptive learning feedback controllers. There is a large gap between such neural adaptive learning feedback controllers and the manner in which the human brain functions. In neuro-cognitive psychology, it is observed that the human makes quick decisions based on association of existing external variables or cues with responses learned and stored based on previous experiences [8]-[9]. If there is a match between current observed circumstances and previously stored responses, the human executes the previously stored response that most closely corresponds to current observations. Otherwise, the human generates a new response to the new conditions.

Likewise, in many applications, the system operates in different environments, with each environment requiring a dynamic description of the system. Such is the case in multiple-model adaptive control [10]-[11], fault tolerant control, and elsewhere. In these applications, neural networks must provide high-level functions such as classification, clustering, and so on. There are similarities between such applications in different environments and the operation of the brain in using previously stored responses.

Learning networks can be used to encapsulate previous experiences into categories of stored responses that correspond to system response in different environments. Motivated by the sensory information handling in parts of the cerebral cortex in the human brain, numerous methods of data clustering have been developed to match current experiences to previously stored experiences [8]. In self-organized clustering, the categories are determined automatically based on various criteria for determining similarity, and new categories may be created if current data does not match with the previous experience [9]. In k-mean clustering [13]-[14], the goal is to partition the inputs into a predetermined k clusters. In the self-organizing map [15-17], previously experienced data is stored as representative categories in the interconnection weights of a neural network, which is trained to produce a low-dimensional discretized mapping of the input space to achieve clustering with dimensionality reduction. Such methods of self-organized clustering are subject to unstable categorization when the distances between categories are too large, and to temporal instability in stored memory when categories are updated and do not retain their distinct characteristics. Methods for adding or resetting categories based on new information in the incoming data and for pruning or removing categories that are not used are generally ad hoc and do not have performance guarantees. An adaptive self-organizing map [2] was applied to feedback control applications in by sorting observed data into previously defined categories, within each of which a feedback controller based on prior experiences is stored. Nevertheless, this method can exhibit temporal instability in stored memory since representatives of several categories can be tuned simultaneously. Therefore, improved methods of self-organized clustering [17] are needed that have more stable categories and temporal behaviors for applications in automatic feedback control for different environments.

Adaptive Resonance Theory (ART) is a match-based clustering approach that provides an explanation of human cognitive information processing [17]. It represents a number of NN models which use supervised and unsupervised learning methods to address problems such as pattern recognition and prediction in the human brain. The basic concept is to categorize the input data into categories, based on a vigilance criterion. Once some categories are established, the new input data is matched with existing categories, which is called the internal memory of an active code. ART matching leads either to a resonant state or a rest state. This matching state is established if the vigilance criterion is met. If the resonance is not achieved, the learning process takes place and an alternative category search is to be established. If the search ends, i.e. no resonance with all the categories, a new category, active code, is created. This match-based learning process is the foundation of ART code stability.

In this paper, motivated by recent neurocognitive models of mechanisms in the brain, a model-free Q-learning-based algorithm is presented to find the optimal solution to the tracking problem of time-varying discrete-time systems. It is assumed that the number of sub-models is not fixed and updated adaptively once a mismatch between the stored sub-models and the new data is detected. The proposed algorithm combines RL with ART algorithms to monitor changes in the dynamics of the environment. The system starts with a number of sub-models based on a prior knowledge. A new sub-model is then added due to the vigilance once a mismatch, no resonance, is established in ART. A responsibility signal is defined using ART algorithm which indicates the likelihood of the current input belonging to the existing sub-models in the time space. Then, an overall Q-function is defined which is the linear combination of all sub-models' Q-functions. Each sub-model contributes to overall Q-function based on the responsibility signal. Based on the presented Q-function structure, a model free algorithm is presented to find the optimal control input for discrete time-varying systems. The proposed method does not require system identification or any knowledge about the system dynamics. The optimal solution is found using only measured data.

This paper is organized as follows. The optimal control problem is discussed in Section 2. Adaptive resonance theory is presented in Section 3. A new formulation for the optimal tracking problem of time-varying discrete-time systems using Q-learning is presented in Section 4. Simulation and conclusion are presented in Sections 5 and 6, respectively.

2. Optimal Tracking Control Problem

Consider the linear system dynamics as

$$x_{k+1} = A_j x_k + B_j u_k \tag{1}$$

where $x_k \in \mathbb{R}^n$ is the state vector of the system, and $u_k \in \mathbb{R}^m$ is the control input. It is assumed that the system operates in multiple environments which may change abruptly from one sub-model to another. That is, $(A_j, B_j) \subset M = \{(A_1, B_1), (A_2, B_2), ..., (A_N, B_N)\}$ with N is the number of sub-models which is generally unknown.

The goal is to design the control input u_k to assure that the states of the system track the reference trajectory r_k in an optimal manner by minimizing a predefined performance function as

$$J(x_k, r_k) = \sum_{i=k}^{\infty} \gamma^{i-k} \left[(x_i - r_i)^T S (x_i - r_i) + u_i^T R u_i \right] \tag{2}$$

with $S \geq 0$ and $R = R^T > 0$. $0 < \gamma \leq 1$ is a discount factor.

The desired reference trajectory is defined as

$$r_{k+1} = F r_k \tag{3}$$

with $r_k \in \mathbb{R}^n$.

The rest of this section assumes there is only one sub-model. This assumption is relaxed in the next section. Based on the system dynamics (1) and the reference trajectory dynamics (3), construct the augmented system as

$$X_{k+1} = \begin{bmatrix} x_{k+1} \\ r_{k+1} \end{bmatrix} = \begin{bmatrix} A_j & \mathbf{0} \\ \mathbf{0} & F \end{bmatrix} \begin{bmatrix} x_k \\ r_k \end{bmatrix} + \begin{bmatrix} B_j \\ \mathbf{0} \end{bmatrix} u_k \equiv T_j X_k + B_{1j} u_k \tag{4}$$

where the augmented state is $X_k = \begin{bmatrix} x_k^T & r_k^T \end{bmatrix}^T$.

The performance function (2) in terms of the state of the augmented system for the sub-model j can be written as

$$\mathrm{V}_j(X_k) = \sum_{i=k}^{\infty} \gamma^{i-k} \left[X_i^T \overline{S}_j X_i + u_i^T R_j u_i \right] \tag{5}$$

where $\overline{S}_j = \begin{bmatrix} S_j & -S_j \\ -S_j & S_j \end{bmatrix}$.

It is shown in [18] that the value function is quadratic in terms of the states of the system (4) as

$$V_j(X_k) = X_k^T P_j X_k \tag{6}$$

Substituting (6) into (5) yields the Bellman equation corresponding to $j-th$ sub-model as

$$X_k^T P_j X_k = X_k^T \overline{S}_j X_k + u_k^T R_j u_k + \gamma X_{k+1}^T P_j X_{k+1} \tag{7}$$

Then, the optimal control input corresponding to sub-model j is given as

$$u_k^* = -\gamma (R_j + \gamma B_{1j}^T P_j B_{1j})^{-1} B_{1j}^T P_j T_j X_k \tag{8}$$

where P_j is obtained by solving the following algebraic Riccati equation (ARE)

$$\overline{S}_j - P_j + \gamma T_j^T P_j T_j - \gamma^2 T_j^T P_j B_{1j} (R_j + \gamma B_{1j}^T P_j B_{1j})^{-1} B_{1j}^T P_j T_j = 0 \tag{9}$$

Eq. (8) is the optimal control input when one has just one model to show the behavior of the system. However, the time-varying systems have different dynamics for each kind of fault or change in the environment. Each type of these dynamics is a sub-model and we should take into account all these sub-models in the value function and control input. In the next section, adaptive resonance theory (ART) for self-organized clustering is proposed to determine the contribution of each sub-model in the general value function using a signal extracted by ART. The overall system is highlighted in Fig.1.

3. Adaptive Resonance Theory and Value Function for Multiple-model Systems

Adaptive resonance theory (ART) is a cognitive and neural theory of how the brain autonomously learns to categorize inputs in a changing world based on a vigilance criterion. Once some categories are

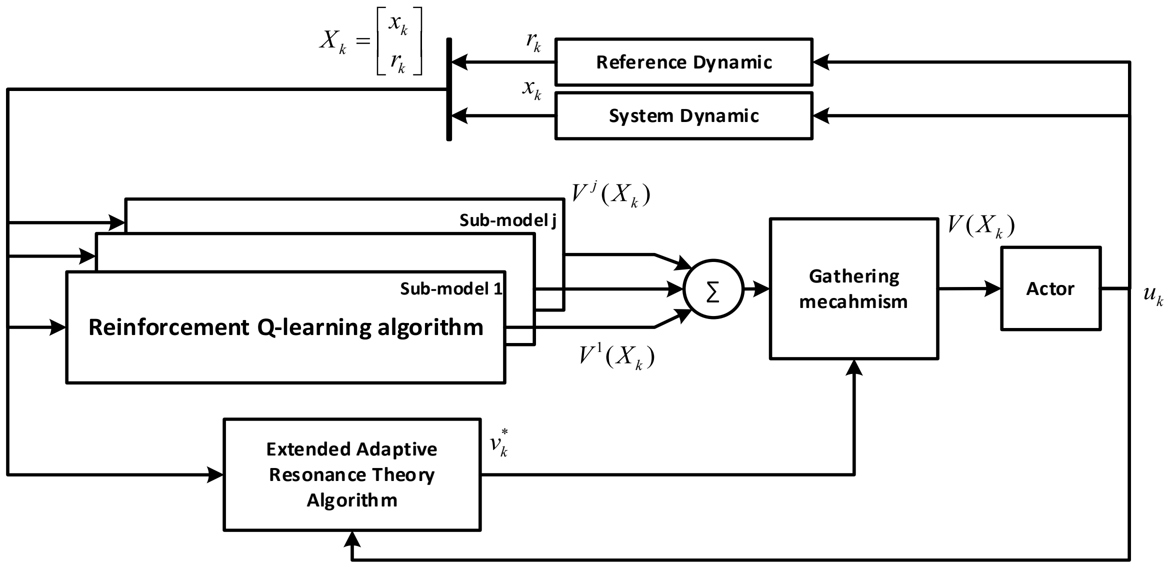

Fig.1 : Overall system

established, the new input data is matched with existing categories if the vigilance criterion is met. The learning process takes place and an alternative category is created if vigilance criterion is not met. This match-based learning process is the foundation of ART code stability. These properties can be used to control of time-varying systems. To do so, ART starts with a number of sub-models based on a prior knowledge. A new sub-model is then added due to the vigilance once a mismatch, no resonance, is established in ART. A responsibility signal is defined using ART algorithm which indicates the likelihood of the current input belonging to the existing sub-models in the time space. Then, an overall Q-function is defined which is the linear combination of all sub-models' Q-functions. Each sub-model contributes to overall Q-function based on the responsibility signal. The fundamentals of ART are first presented and it then is combined with multiple model and Q-learning to approximate the optimal value function and consequently the optimal solution for uncertain time-varying systems.

3.1. Adaptive Resonance Theory

ART remedies many of these defects noted above by employing two maps between two spaces [9][12][19]. An input data space F1 is called the short-term memory, and a category feature space F2 is called the long-term memory. One map is from the input data space F1 to the category feature space F2. This is termed the bottom-up map. A second dual map is from the category feature space F2 back to the input space F1, known as the top-down map. The process of matching the observed data to a stored category occurs as a result of the interaction of these two maps. The model postulates that 'top-down' expectations mapped back to the input space take the form of a category prototype which is then compared, using certain metrics, with the actual input data as detected. When the difference between the actual input and the category prototype does not exceed a threshold called the vigilance parameter, the input is considered a member of the expected class. ART thus offers a solution to the plasticity versus stability paradox [19][20]. Furthermore, in the ART network, pruning and resetting of categories is furthermore formally defined using certain parameters. Furthermore, the number of categories is not predefined. There are many different forms of the ART theory which utilize different methods or metrics for determining the match between the input signal and the category prototype.

These notions are captured mathematically as follows. The complete ART algorithm is detailed in Table 1. The situation of our concern is when the observed input data for the ART are the inputs and outputs of a dynamical system (1). The input data for the ART are defined as the vector of past controls and states

$$d_k = [x_k^T \quad x_{k-1}^T \quad \dots \quad \dots \quad x_{k-\delta_x}^T \mid \quad u_{k-1}^T \quad u_{k-2}^T \quad \dots \quad u_{k-\delta_u}^T]^T \tag{10}$$

where δ_x, δ_u depend on the problem in hand and are determined by the user based on experience. On one hand, the greater δ_x, δ_u are, the higher dimension of the clustering space, which makes the clustering more complicated, on the other hand, accurate clustering requires to take into account enough number of past input and the state to distinguish different model of the system. These input data are viewed as residing in the F1 layer, and a map to the category feature space, or F2 layer, is provided by

$$v_k = \frac{W^T d_k}{\left\| W^T d_k \right\|} = \begin{bmatrix} v_k^1 & \dots & v_k^j & \dots & v_k^N \end{bmatrix}^T \tag{11}$$

where matrix W is interpreted as the weight matrix of a neural network. Define $W = \begin{bmatrix} W_1 & W_2 & \cdots & W_J \end{bmatrix}$ where the columns $W_j, j = 1,...,N$ are the category representatives currently stored in the ART network. As such, (11) is an inner product that compares the input data d_k to each category column of W, with elements v_k^j larger for categories W_j that are closer to d_k in Euclidean norm.

The next step in ART is to use competitive learning to choose a winning category to which v_k most closely belongs. To accomplish this, sort the entries v_k^j in F2 in descending order of magnitude to define the ordered list $j_1, j_2, ..., j_N$. Define $j^* = j_1$, the largest index as the chosen category F2 winner, Define the F2 vector v_k^* as a vector of zeros with an entry of 1 in position $j^* = j_1$. Define the dual map from F2 to F1 as

$$d_k^* = W v_k^* \tag{12}$$

Then, $d_k^* = W_{j^*}$ is the expected or hypothesized category prototype in F1 space. That is, d_k^* is the ideal data signal that would produce category representative W_{j^*} using the map (11).

The key step in ART now occurs, namely matching the input data to a stored category. To accomplish this, compare the input data to the category prototype $d_k^* = W_{j^*}$ in F1 space. Many norms have been proposed for this matching test, including techniques from fuzzy logic. We use simply the Euclidean norm condition

$$\left\| d_k - d_k^* \right\| < \nu \tag{13}$$

where ν is known as the vigilance parameter, specified by the user. If this condition is satisfied, then column $W_{j^*} = W_{j_1}$ is declared the winning category and resonance is said to occur. Then, the weights in column W_{j^*} are updated to more closely fit the observed input data. This is accomplished by using the adaptive learning algorithm

$$W_{k+1}^{j^*} = \alpha d_k + (1-\alpha) W_k^{j^*} \tag{14}$$

To avoid unstable categorization and temporal instability in the stored categories, only one step of this update is performed. Note that non-winning category representatives are not updated. This preserves stability of categorization in ART. If condition (13) does not hold, then no winning category is declared, and the next closest category to v_k is selected. That is, one sets the trial value $j^* = j_2$. Then the map from F2 to F1 (12) and the matching test (13) are repeated. If match occurs, then column $W_{j^*} = W_{j_2}$ is declared the winning category and resonance is said to occur. Then, the weights in column W_{j^*} are updated by one step of adaptive learning algorithm (14). Again, if condition (13) does not hold, then no winning category is declared, and the next closest category to v_k is selected, namely $j^* = j_3$. The steps (12), (13) are repeated until a winning category is found. If no winning category is found, then a new category is declared, and the input data itself is added as the last column of NN weight matrix W, so that the number of categories increases to $J+1$. Thus, adding new categories is automatic in ART, whereas in other clustering methods, it is usually ad hoc.

In summary, the ART network consists of four stages. The first stage is the preprocessing stage wherein the input data d_k is mapped to a vector v_k in category space F2. The second stage is the choice stage of selecting a winning category in F2, namely j^*, which is the category to which the data most closely belongs. Third, the match stage, where the winning category is mapped back to F1 to obtain the hypothesized category prototype d_k^*, which is compared to the current observed data d_k. If there is a match between and d_k and d_k^*, resonance is said to occur. Then, in a fourth adaptation stage, the winning category representative is update to more closely match the current observed data.

The complete ART algorithm is detailed in Table 1. Several details remain about the functioning of ART.

Table.1. Extended ART Algorithm

	Select λ, ρ, $0 \le \kappa \le J$,
	Initialize $k=0$, x_0, $\tau_0=0$, $f_0=\lambda$
(1)	$k=k+1$
	$x_{k+1}=Ax_k+Bu_k$ $x_k \in \mathbb{R}^n$
	Form d_k $d_k \in \mathbb{R}^{n(\delta x+1)m\delta u}$ $d_k=\dfrac{d_k}{\lVert d_k \rVert}$
	$\tau_{k+1}=\max(\tau_k-1,0)$, Where τ Is Refractory Index $f_{k+1}=\max(f_k-1,0)$, And f Fading Index
	If $f_j(k)=0$ Remove Column W_j
	$v_k=\dfrac{W^T d_k}{\lVert W^T d_k \rVert}=\begin{bmatrix} v_1(k) & \dots & v_j(k) & \dots & v_J(k) \end{bmatrix}^T$
	Order $v_j(k)$ In Descending Order 1 To -1, Call the Ordered Indices $j_1, j_2 \dots j_J$
	Set $l=1$
(2)	Set $j^*=j_l$
	Define $v(k)^*=\begin{bmatrix} 0 & 0 & 1 & 0 & 0 \end{bmatrix}^T$ As A Zero Vector With 1 In Position j^*
	If $\tau_{j^*}(k)=0$ Then $d_k^*=W_{j^*}$
	Else Go To (3)
	If $\lVert d_k-d^*_k \rVert < \nu$ Then
	Train W_{j^*} As ($W_{j^*}(k+1)=\alpha d_k+(1-\alpha)W_{j^*}(k)$) Set $f_{j^*}(k+1)=\lambda$
	Set $\tau_{j_i}(k+1)=\rho$, $\forall j_i < l$ Start Refractory
	Else Go To (3)
(3)	Set $l=l+1$
	If $l>\kappa$ Go To (2)
	Set $W_{J+1}=d(k)$
	Go To (1)

The first is the specification of how many categories to try before the condition of 'no winning category' is declared. This number is called κ in Table 1. Next, if a category is selected as the winning category $j^*=j_l$ in F2, and if this choice fails the match test (13) in F1, then a refractory time period ρ is initiated for the category j_l and it cannot be used again until the refractory period is over. This is kept track using a refractory index $\tau_{j_l}(k)$ in Table 1. Finally, ART includes a formal method for deleting categories that are never used. This is accomplished using a fading index J-vector f_k in Table 1, where the j-th entry of f_k is the fading index for category j. If category j is not used during the fading period λ, then category column W_j of the NN weight matrix is removed, and the number of categories J is set to $J-1$

3.2. New Value Function Structure Using ART

In this subsection, a general value function approximation for multiple-model linear systems using ART is presented. In the proposed value function approximation scheme, each sub-model contributes to the value function using a responsibility signal. In fact, the general value function is given by

$$V(X_k)=\sum_{j=1}^{N} v_k^j V_j(X_k)=\sum_{j=1}^{N} v_k^j X_k^T P_j X_k \tag{15}$$

where $v_k^j \quad j=1,...,N$ are the responsibility signals which determine the contribution of each sub-model to the general value function.

Considering (4) and (15) in (2), yields the Bellman equation for time-varying systems

$$\sum_{j=1}^{N} v_k^j X_k^T P_j X_k = X_k^T \bar{S} X_k + u_k^T R u_k + \gamma \sum_{j=1}^{N} v_k^j X_{k+1}^T P_j X_{k+1} \tag{16}$$

and the Hamiltonian is defined as

$$H(X_k,u_k)=X_k^T \bar{S} X_k + u_k^T R u_k + \gamma \sum_{j=1}^{N} v_k^j X_{k+1}^T P_j X_{k+1} - \sum_{j=1}^{N} v_k^j X_k^T P_j X_k \tag{17}$$

Applying the stationarity condition $\partial H(X_k,u_k)/\partial u_k = 0$ yields the optimal control input as

$$u_k^* = \gamma \left(\sum_{j=1}^{J} v_k^j (R+\gamma B_{1j}^T P_j B_{1j}) \right)^{-1} \left(\sum_{j=1}^{J} v_k^j B_{1j}^T P_j T_j \right) X_k \tag{18}$$

where $P_j \quad j=1,...N$ are obtained by solving a set of AREs (9).

Remark 1. Note that complete knowledge about the augmented system dynamics is required to find the optimal control input (18). In the next section, RL is used to find the solution to the optimal tracking problem without requiring any knowledge about the system dynamics.

4. Q-learning to Solve Optimal Tracking Problem of Multiple-model Systems

The solution to the optimal multiple-model tracking control problem needs complete knowledge about the system dynamics and reference trajectory dynamics. In this section a Q-learning algorithm is developed that solves this problem online without requiring any knowledge of the augmented system dynamics.

Based on the Bellman equation (7), the discrete-time Q-function for j-th sub-system is defined as

$$Q_j(k)=X_k{}^T \bar{S}_j X_k + u_k{}^T R_j u_k + \gamma X_{k+1}{}^T P_j X_{k+1} \tag{19}$$

Substituting the augmented system (4) in (19) yields,

$$\begin{aligned} Q_j(X_k,u_k) &= X_k{}^T \bar{S}_j X_k + u_k{}^T R_j u_k + \gamma (T_i X_k + \mathrm{B}_1 u_k)^T P_j (T_i X_k + \mathrm{B}_1 u_k) \\ &= \begin{bmatrix} X_k \\ u_k \end{bmatrix}^T \begin{bmatrix} \bar{S}_j + \gamma T_j^T P_j T_j & \gamma T_j^T P_j B_1 \\ \gamma B_1^T P_j T_j & R_j + \gamma B_1^T P_j B_1 \end{bmatrix} \begin{bmatrix} X_k \\ u_k \end{bmatrix} \\ &= \begin{bmatrix} X_k \\ u_k \end{bmatrix}^T \begin{bmatrix} H_j^{xx} & H_j^{xu} \\ H_j^{ux} & H_j^{uu} \end{bmatrix} \begin{bmatrix} X_k \\ u_k \end{bmatrix} \\ &= Z_k{}^T H_j Z_k \end{aligned} \tag{20}$$

For the multiple-model systems, the general Q function is defined as

$$Q(k)=\sum_{j=1}^{N} v_k^j Q_j(X_k) \tag{21}$$

By substituting the quadratic form (20) in (21), one has

$$\begin{aligned}
Q(k) &= \sum_{j=1}^{N} v_k^j Q_j(X_k) = v_{k1}^1 Z_k^T H_1 Z_k + v_k^2 Z_k^T H_2 Z_k + \ldots + v_k^N Z_k^T H_J Z_k \\
&= Z_k^T \Big(\sum_{j=1}^{N} v_k^j H_j\Big) Z_k \\
&= Z_k^T \begin{bmatrix} \sum_{j=1}^{N} v_k^j H_j^{xx} & \sum_{j=1}^{N} v_k^j H_j^{xu} \\ \sum_{j=1}^{N} v_k^j H_j^{ux} & \sum_{j=1}^{N} v_k^j H_j^{uu} \end{bmatrix} Z_k \\
&= Z_k^T \mathrm{H} Z_k
\end{aligned} \tag{22}$$

Eq. (22) shows that the general Q-function for multiple-model systems is quadratic in terms of the states of the augmented system and control input.

Applying the stationarity condition $dQ(k)/du_k = 0$ yields,

$$u_k^* = \left(\sum_{j=1}^{N} v_k^j H_j^{uu}\right)^{-1} \left(\sum_{j=1}^{N} v_k^j H_j^{ux}\right) X_k \tag{23}$$

Now, we can present a Q-learning algorithm to solve the optimal tracking control problem of multiple-model systems online without knowing the augmented system dynamics (T_j, B_{1j}).

The Bellman equation (16) in terms of Q-function is given as

$$Q(X_k, u_k) = X_k^T \bar{S} X_k + u_k^T R u_k + \gamma Q(X_{k+1}, u_{k+1}) \tag{24}$$

Substituting (22) into (24), the Q-function Bellman equation (24) becomes

$$Z_k^T \mathrm{H} Z_k = X_k^T S_1 X_k + u_k^T R u_k + \gamma Z_{k+1}^T \mathrm{H} Z_{k+1} \tag{25}$$

Policy iteration is especially easy to implement in terms of the Q-function, as follows.

Algorithm 1. Policy Iteration using Q-function

1. Policy evaluation

$$Z_k^T \mathrm{H}^{i+1} Z_k = X_k^T \bar{S} X_k + (u_k^i)^T R(u_k^i) + \gamma Z_{k+1}^T \mathrm{H}^{i+1} Z_{k+1} \tag{26}$$

2. Policy improvement

$$u_k^{i+1} = -\left(\sum_{j=1}^{N} v_k^j (\mathrm{H}_j^{uu})^{i+1}\right)^{-1} \left(\sum_{j=1}^{N} v_k^j (\mathrm{H}_j^{ux})^{i+1}\right) X_k \tag{27}$$

Q-Learning algorithm attempts to learn the cost of the current category state, and taking a specific action toward minimizing the performance index. The advantage of Q-learning algorithm is that convergence guarantees can be given even when function approximation is used to estimate the action values.

Remark 2. Note that the policy improvement step (27) in Algorithm 1, which is given by minimizing the Q-function (24) with respect to the control input, can be carried out in terms of the learned kernel matrix H^{i+1} without resorting to the system dynamics.

Remark 3. The computation complexity of the proposed algorithm depends on the number of the states of the systems and also the number of sub-models. If we have more states and sub-models, it takes more time for the algorithm to learn the optimal control input.

5. Simulation

To show the effectiveness of the proposed method, simulations have been carried out on a mass-spring-damper system. The system dynamics is

$$\begin{aligned} x_{1,k+1} &= x_{1,k} + x_{2,k} \\ x_{2,k+1} &= -\frac{k}{m}x_{1,k} + (1-\frac{b}{m})x_{2,k} + \frac{1}{m}u_k \end{aligned} \tag{28}$$

Three different system behaviors are considered for this simulation. The parameters set of each interval are provided in Table 2. These parameters change the system dynamics (28). For each time interval, a system is activated for the corresponding parameters.

Table 2: The parameters of three system dynamics for three-time intervals

Time Interval	System Parameters
$0 < t \le 300$	$k_1 = 10 \quad b_1 = 10 \quad m_1 = 90$
$300 < t \le 600$	$k_2 = 30 \quad b_2 = 15 \quad m_2 = 90$
$600 < t \le 1000$	$k_3 = 50 \quad b_3 = 50 \quad m_3 = 90$

The extended ART parameters are chosen as $\lambda = 100$, $\rho = 0.78$, $\kappa = 20$, and $\delta_x = \delta_u = 4$. Fig.2 shows the norm of the difference between the optimal control gain and the computed gain. It is obvious from the figure that the gain converges to the optimal value. Spikes are seen in the first iterations with system 1, a smaller spike is seen when the dynamic changes at 300 points, and tiny spike at 600. This is because the extended ART sub-models are carried through the three systems, and the change in the system dynamic is not very huge. Fig 3 shows the states of the system after applying the computed control input using the proposed approach. The optimal gain and the computed gain are shown in table 3.

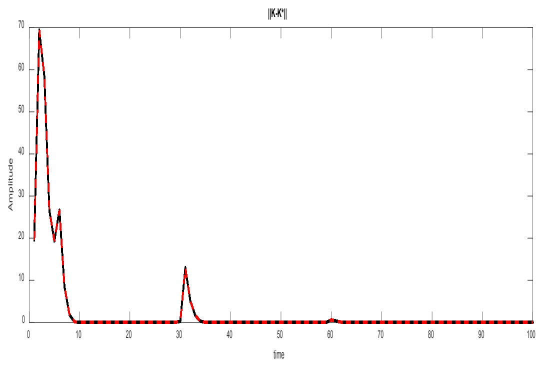

Fig. 2. The norm of the difference between the optimal control gain and the computed gain

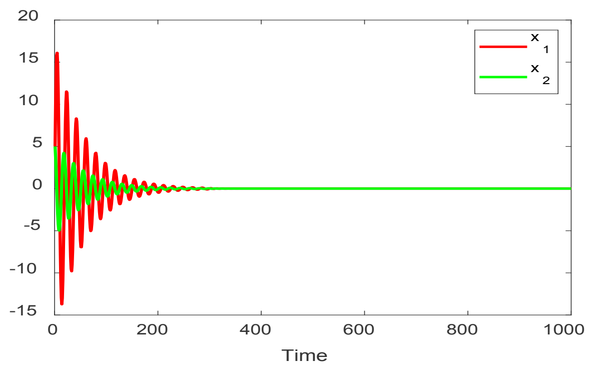

Fig. 3. The states of the system

Table 3: The optimal gains vs the Computed gains

	Optimal Gains	Computed Gains

System 1	$K^*_{Sys1} = [-0.127 \quad 3.019]$	$K_{Sys1} = [-0.127 \quad 3.019]$
System 2	$K^*_{Sys2} = [-4.292 \quad 23.686]$	$K_{Sys2} = [-4.292 \quad 23.686]$
System 3	$K^*_{Sys3} = [-0.489 \quad 1.297]$	$K_{Sys3} = [-0.489 \quad 1.297]$

6. Conclusion

In this paper, ART clustering algorithm is combined with RL to find the optimal solution to the tracking problem of time-varying discrete-time systems. The changes in the system behavior is taken into account using multiple-model approach. ART algorithm generates sub-models based on the clustering match-based method. A Q-learning based algorithm is then used to find the optimal solution online and without requiring any knowledge of the system dynamics. Each sub-model contributes into Q-function through a responsibility signal generated by ART.

A possible extension of this work can be sample-data control systems with noisy sampling interval [21] [22] as they are time-varying systems.

Acknowledgment

This work is supported by ONR grant N00014-17-1-2239, ONR Grant N00014-18-1-2221, NSF Grant ECCS-1839804.

References

[1] F. L. Lewis, K. Liu, and A. Yesildirek, Neural net robot controller with guaranteed tracking performance, IEEE Trans. Neural Networks, 6 (3) (1995) 703-715.

[2] B. Kiumarsi, F. L. Lewis, D. S. Levine, Optimal control of nonlinear discrete time-varying systems using a new neural network approximation structure, Neurocomputing, 156 (2015) 157-165

[3] S. J. Bradtke, B. E. Ydstie, and A. G. Barto, A. G, Adaptive linear quadratic control using policy iteration, Proceedings of IEEE American control conference, Baltimore, Maryland, (1994) 3475-3479.

[4] A. Al-Tamimi, F.L. Lewis, M. Abu-Khalaf, Model-free Q-learning designs for linear discrete-time zero-sum games with application to HH-infinity control, Automatica, 43 (3) (2007) 473–481.

[5] Q. Wei, R. Song, Q. Sun, Nonlinear neuro-optimal tracking control via stable iterative Q-learning algorithm, Neurocomputing, 168 (2015) 520-528.

[6] F.L. Lewis, D. Vrabie, K.G. Vamvoudakis, Reinforcement learning and feedback control using natural decision methods to design optimal adaptive controllers, IEEE Systems Magazine, 32 (6) (2012) 76–105

[7] K.G. Vamvoudakis, F.L. Lewis, Online actor–critic algorithm to solve the continuous-time infinite horizon optimal control problem, Automatica 46 (5) (2010) 878–888.

[8] P.J. Werbos, A menu of designs for reinforcement learning over time, Neural networks for control, MIT Press, Cambridge, MA (1991), pp. 67–95.

[9] D. S. Levine, Neural dynamics of affect, gist, probability, and choice, Cognitive System Research 16 (2012) 57-72.

[10] K.S. Narendra, J. Balakrishnan, Improving transient response of adaptive control systems using multiple models and switching, IEEE Trans. Automatic Control, 39 (9) (1994) 1861–1866.

[11] K. Pawelzik, J. Kohlmorge, K.R Muller, Annealed competition of experts for a segmentation and classification of switching dynamics, Neural Computation, 8 (1996) 340–356.

[12] S. Grossberg, Competitive learning: From interactive activation to adaptive resonance. Cognitive science, 11 (1) (1987) 23-63.

[13] J. A. Hartigan, M. A. Wong, Algorithm AS 136: A K-Means Clustering Algorithm, Journal of the Royal Statistical Society. Series C (Applied Statistics), 28 (1) (1979), 100–108.

[14] M.C. Naldi, R.J.G.B. Campello, Comparison of distributed evolutionary k-means clustering algorithms, Neurocomputing, 163 (2015) 78-93.

[15] T. Kohonen, Self-organizing Maps, Springer, Berlin and Hiidelberg, 1995.

[16] F. Coleca, A.State, S.Klement, E.Barth, T.Martinetz, Self-organizing maps for hand and full body tracking, Neurocomputing, 47 (2015) 174-184.

[17] S. Grossberg, Adaptive pattern classification and universal recoding, I: Parallel development and coding of neural feature detectors & II: Feedback, expectation, olfaction, and illusions, Biological Cybernetics, (1976) 187-202.

[18] B. Kiumarsi, F. L. Lewis, H. Modares, A. Karimpour, and M-B. Naghibi, Reinforcement Q-learning for optimal tracking control of linear discrete-time systems with unknown dynamics. Automatica, 50 (4) (2014) 1167-1175.

[19] T. Frank, K. F. Kraiss and T. Kuhlen, Comparative analysis of fuzzy ART and ART-2A network clustering performance, IEEE Transactions on Neural Networks, 9 (1998) 544-559.

[20] K. Oweiss, R. Jin, Y. Suhail, Identifying neuronal assemblies with local and global connectivity with scale space spectral clustering, Neurocomputing, 70 (2007) 1728-1734.

[21] B. Shen, H. Tan, Z. Wang, and T. Huang, Quantized/saturated control for sample-data systems under noisy sampling intervals: a confluent vandermonde matrix approach, IEEE Transactions on Automatic Control, 62 (9) (2017) 4753-4759.

[22] B. Shen, Z. Wang, and T. Huang, Stabilization for sampled-data systems under noisy sampling interval, Automatica, 63 (2016) 162-166.