A Reinforcement Learning Fuzzy Controller for Set-Point Regulator Problems

Augustine O. Esogbue, Warren E. Hearnes II, and Qiang Song
Intelligent Systems and Control Lab
School of Industrial and Systems Engineering
Georgia Institute of Technology
Atlanta, GA 30332-0205
e-mail: augustine.esogbue@isye.gatech.edu

Abstract— **In recent years there has been an explosion of research into *intelligent control*. Intelligent control refers to controllers that can analyze their performance and make necessary changes to their behavior in order to satisfy certain predefined control goals. This paper describes a self-learning controller model that can efficiently learn the control law for complex systems through reinforcement learning techniques and dynamic programming-like algorithms. The controller is applied to a class of problems called general set-point regulator problems in which the objective is to drive the system to the set-point while optimizing some performance objective function, making no a priori assumptions about the dynamics of the plant or its optimal trajectory. The relevant tasks for a self-learning controller are discussed. Learning is accomplished via incremental, online dynamic programming-like algorithms. Both temporal differences and Q-learning are used in the learning algorithm. Experimental results with both are reported on the inverted pendulum balancing problem, the power system stabilization problem, and the tethered satellite system retrieval problem.**

Keywords— **Fuzzy control, reinforcement learning, dynamic programming, temporal differences, Q-learning, inverted pendulum, power system stabilization, tethered satellite system retrieval.**

I. Introduction

As control problems become more complex in real-world applications, the use of traditional control techniques requiring mathematical models of the plant becomes more difficult to apply. In recent years there has been an explosion of research into controllers that can analyze their performance and make necessary changes to their behavior in order to satisfy certain predefined control goals [2], [4], [10], [19], [20]. The advantage of such a controller becomes apparent when the dynamics of the plant are either unknown or are too complex to solve analytically.

Intelligent controllers have several important advantages, such as higher robustness, shorter development time, and less assumptions about the dynamical behavior of the plant, that make them attractive for application to real-world problems [10], [11], [19]. Fuzzy set theory provides a mathematical framework for modeling vagueness and imprecision. Neural networks have the ability to learn complex mappings, generalize information, and classify inputs. Hybrid controllers utilize the advantages of each, as well as other novel techniques, creating a powerful tool for intelligent control.

The controller model described in this paper is directed toward nonlinear set-point regulator control problems that have either unknown or uncertain dynamics. The state and control spaces are assumed continuous, although application to discrete variable problems have been investigated [20]. No a priori information about the optimal control policy is assumed. The only external feedback required is the notification that the system has either reached a failure state or a success state.

II. The General Set-Point Regulator Problem

Set-point regulator problems require that the process be driven to a desired final state (the set-point) in a manner that optimizes some performance measure. The formal definition of the class of problems investigated is presented in this section [20]. Let $\mathbf{X}$ be the state space and $\mathbf{U}$ be the control space for a process and $\mathbf{x}_k \in \mathbf{X}$ and $\mathbf{u}_k \in \mathbf{U}$ be the state and control vectors at time step k, respectively. Let the initial state be defined as $\mathbf{x}_{\text{initial}}$, the set-point be $\mathbf{x}_{\text{final}}$, and T be the (possibly random) time at which the process reaches the set-point. The problem is defined as

$$\text{opt}_{(\mathbf{u}_0,\ldots,\mathbf{u}_T)\in\mathbf{U}^T}\left\{E\left[J(\mathbf{x}_{\text{initial}})\right]\right\} = \left\{E\left[\sum_{k=1}^{T} p_k\right]\right\} \tag{II.1}$$

subject to

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k, w_k) \tag{II.2}$$
$$\mathbf{x}_0 = \mathbf{x}_{\text{initial}} \tag{II.3}$$
$$\mathbf{x}_T = \mathbf{x}_{\text{final}} \tag{II.4}$$
$$p_k = p(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{x}_k) \tag{II.5}$$
$$\mathbf{x}_k \in \mathbf{X} \tag{II.6}$$

where $f(\mathbf{x}_k, \mathbf{u}_k, w_k)$ in Equation II.2 is the (possibly random) state transition function for the process and $p(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{x}_k)$ in Equation II.5 is some performance measure that is measurable at each time step k, such as the integral of the absolute error (IAE). If $\mathbf{u}^\star(\mathbf{x}_{\text{initial}}, \mathbf{x}_{\text{final}}) = (\mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_{T-1})$ is the optimal control policy for the above problem then by Bellman's

principal of optimality if state x_k occurs with positive probability, then $(\mathbf{u}_k, \mathbf{u}_{k+1}, \ldots, \mathbf{u}_{T-1})$ is the optimal policy for the subproblem where $\mathbf{x}_{\text{initial}} = \mathbf{x}_k$. This important feature of the problem allows for the use of efficient dynamic programming-like algorithms.

Formulating the above as a dynamic program, let the state be the state vector $\mathbf{x} \in \mathbf{X}$, the stage be the time step $k = \{0, 1, \ldots, T\}$, the decision variable be the control vector $\mathbf{u} \in \mathbf{U}$, the immediate reward at stage k while in state $\mathbf{x}_k$ and taking action $\mathbf{u}_k$ be the scalar function $p(\mathbf{x}_k, \mathbf{u}_k)$, and the (possibly random) transition function be $f(\mathbf{x}_k, \mathbf{u}_k, w_k)$. For a one-stage problem $(f(\mathbf{x}_0, \mathbf{u}_0, w_0) = \mathbf{x}_{\text{final}})$ the functional equation becomes

$$\begin{aligned} F_0(\mathbf{x}_0) &= \text{opt}_{\mathbf{u}_0 \in \mathbf{U}} \{E[p(\mathbf{x}_0, \mathbf{u}_0) + F_1(f(\mathbf{x}_0, \mathbf{u}_0, w_0))]\} \\ &= \text{opt}_{\mathbf{u}_0 \in \mathbf{U}} \{E[p(\mathbf{x}_k, \mathbf{u}_k)]\} + F_1(\mathbf{x}_{\text{final}}) \end{aligned} \tag{II.7}$$

where $F_1(\mathbf{x}_{\text{final}})$ is a boundary condition. Generalizing to a multistage process gives

$$F_k(\mathbf{x}_k) = \text{opt}_{\mathbf{u}_k \in \mathbf{U}} \{E[p(\mathbf{x}_k, \mathbf{u}_k) + F_{k+1}(f(\mathbf{x}_k, \mathbf{u}_k, w_k))]\} \tag{II.8}$$

Since $f(\mathbf{x}_k, \mathbf{u}_k, w_k)$ is not known, incremental online learning algorithms such as Sutton's temporal differences (TD) method [22] or Watkin's Q-Learning method [23], [24] must be used to approximate the optimal control policy.

III. The Self-Learning Fuzzy Controller

In ongoing research in our laboratory, we have developed, investigated, and applied various versions of intelligent controllers to a variety of problems of interest to our funding agencies [9], [10], [11], [12], [13]. Our approach can be broken down into a set of subtasks:

1. Develop an efficient representation of the state space $\mathbf{X}$ and control space $\mathbf{U}$.
2. Provide a "good" internal reinforcement signal $\hat{r}(\mathbf{x}, \mathbf{u}, k)$ based on the current level of learning at time step k.
3. Search the control space $\mathbf{U}$ for the "best" control action $\mathbf{u}$ for each state $\mathbf{x}$ visited during the learning phase(s).
   - Determine a logical and efficient method for assigning credit for *success* and *failure* and use this to update the internal reinforcement algorithm.
   - Determine the "best" control action $\mathbf{u}^\star$ based on $\hat{G}(\mathbf{x}, \mathbf{u}, k)$ and any possible constraints $\tilde{C}(k)$.
4. Generalize the current state of learning—i.e., the current learned control law $\hat{F}(\mathbf{x}, k)$ at time step k—to the entire state space $\mathbf{X}$.
5. Determine a method of implementing goals $\tilde{G}(k)$ in order to approach optimality during the learning phase(s).

Efficient State and Control Space Representation: If $\mathbf{X}$ and $\mathbf{U}$ are sufficiently large then the "curse of dimensionality" arises and some type of generalization is required in order to keep the learning problem tractable. Fuzzy controllers possess the ability to efficiently approximate input-output mapping functions by reducing the precision of the state and control variables and their relationships. Borrowing from cluster analysis, one can partition the state space into a predefined set of fuzzy clusters [8]. A network of nodes discretizes $\mathbf{X}$ and $\mathbf{U}$ into n and m fuzzy subsets, respectively, defined by gaussian membership functions centered around a prototype member of the fuzzy set (see Figure III.1 below) [10], [20].

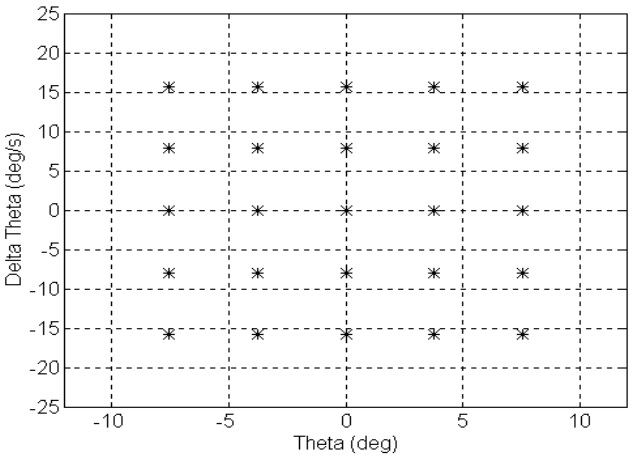

Fig. III.1. Example of the fuzzy discretization of the 2-dimensional state space for the inverted pendulum problem with 25 fuzzy subsets. The subset centers are marked with '*'.

Any state $\mathbf{x} \in \mathbf{X}$ can be described by an n-element vector of membership function values for each of the n fuzzy subsets. This approach bypasses the geometric increase in the number of rules as state variables are added since the number of nodes represent the number of fuzzy rules.

Internal Reinforcement Signal: A "good" internal reinforcement signal $\hat{r}(\mathbf{x}, \mathbf{u}, k)$ based on the current level of learning at time step k is required. Several DP-based algorithms are used to learn real-time control strategies, including Watkin's Q-learning algorithm and Sutton's temporal differences [1]. In Markovian environments, both Sutton's temporal differences (TD) and Watkin's Q-learning are approximations to dynamic programming (DP). Using stochastic approximation theory, it is shown that both of these algorithms belong to a general class of convergent algorithms.

In our TD implementation a current prediction function value $p_k(i)$ for $i \in \{1, \ldots, n\}$ is associated with each of the n fuzzy subsets of the state space that represents the prediction of the 'expected' degree of success for the controller when closest to node i at time k [10], [20]. Thus, $\hat{r}(\mathbf{x}, \mathbf{u}, k)$ is a function of the change in the prediction function values between two consecutive nodes—e.g., if the process moves from being nearest node i to being nearest node j then $\hat{r}(\mathbf{x}, \mathbf{u}, k) = h(p_k(j) - p_k(i))$. In addition, $p_k(j) = 1$ if the process moved within a spec-

ified neighborhood of the set-point and $p_k(j) = -1$ if the process moved outside the state space boundaries. Thus,

$$\hat{r}(\mathbf{x}, \mathbf{u}, k) = \begin{cases} h\left(1 - p_k(N_k)\right) & \text{if success} \\ h\left(-1 - p_k(N_k)\right) & \text{if failure} \\ h\left(p_k(N_{k+1}) - p_k(N_k)\right) & \text{otherwise} \end{cases} \tag{III.9}$$

where $N_k \in \{1, \ldots, n\}$ represents the fuzzy subset of the state space with the highest membership function value for state $\mathbf{x}_k$. The function $h(\cdot)$ may be used to relate the reinforcement signal with the membership function value of $\mathbf{x}$ in N_k.

In the Q-learning implementation a value $Q_k(\mathbf{x}, \mathbf{u})$ is associated with each of the $n \times m$ state-action pairs. These Q-values represent an estimate of the functional equation $F_k(\mathbf{x}_k)$ in Equation II.8 in the sense that $F_k(\mathbf{x}_k) = \text{opt}_{\mathbf{u}_k \in U}\{Q_k(\mathbf{x}_k, \mathbf{u}_k)\}$. The chosen performance measure is the *integral of the absolute error* (IAE) thus $\text{opt}_{\mathbf{u}_k \in U}\{Q_k(\mathbf{x}_k, \mathbf{u}_k)\}$ represents the minimal IAE when in state $\mathbf{x}_k$ and following the optimal control policy to $\mathbf{x}_{\text{final}}$. Thus, the internal reinforcement signal $\hat{r}(\mathbf{x}, \mathbf{u}, k)$ is the IAE at time step k,

$$\hat{r}(\mathbf{x}, \mathbf{u}, k) = \begin{cases} M & \text{if failure} \\ AE\left(\mathbf{x}_k, \mathbf{x}_{\text{set-point}}\right) & \text{otherwise} \end{cases} \tag{III.10}$$

where M is some sufficiently large number and $AE(\mathbf{x}_k, \mathbf{x}_{\text{set-point}})$ is a scalar function representing the instantaneous absolute error between $\mathbf{x}_k$ and the set-point.

Systematic Search of the Control Space: Searching the control space $\mathbf{U}$ for the "best" control action $\mathbf{u}$ for each state $\mathbf{x}$ visited during the learning phase(s) is vital to the overall learning efficiency of the controller algorithm. In the TD implementation the controller searches a subset of the control space $\mathbf{U}$ stochastically. During the learning phase probabilities are used to select a control action $\mathbf{u}$ whenever a new fuzzy subset of the state space has the highest membership function value—i.e., $N(\mathbf{x}_{k+1}) \neq N(\mathbf{x}_k)$. The internal reinforcement signal from Equation III.9 is used to increase or decrease the probability of a control action $\mathbf{u}_c$, $c \in \{1, \ldots, m\}$ being chosen at that node in the future. In the Q-learning implementation, the probabilities remained equal and constant for each state-action pair in order to back up the functional equation values similar to asynchronous DP [1]. Both TD and Q-learning methods implicitly learn the state transition information via the update algorithms [1], [20], [22]. The "best" control action $\mathbf{u}^\star$ for a particular fuzzy subset of the state space is identified by a high probability of being chosen when near that prototypical state in TD and by the lowest Q-value in Q-learning.

Generalized Learning: The current state of learning must be generalized to the entire state space $\mathbf{X}$. Assuming that there is some optimal control law $F^\star(\mathbf{x})$, the goal is for $\hat{F}(\mathbf{x}, k)$ to converge to $F^\star(\mathbf{x})$ as $k \to \infty$. Because of the high-dimensionality of the state and control spaces, all the learned knowledge approximating $F^\star(\mathbf{x})$ is stored in the learned membership functions for the n state nodes and m control nodes. This information is used to determine $\hat{F}(\mathbf{x}, k)$ through the use of fuzzy inference and defuzzification techniques. The use of fuzzy set theory is an efficient method, though careful consideration must be taken in the choice of the inference/defuzzification method since it has a dramatic impact on the resulting control law $\hat{F}(\mathbf{x}, k)$.

Implementing Goals: The last task is to determine a method of implementing goals $\tilde{G}(k)$ in order to approach optimality during the learning phase(s). These goals may be in the class of performance measures and give a benchmark to judge the controller by. Using dynamic programming-like algorithms such as Q-learning allows for goals such as *minimize the IAE to the set-point* to be approximated in an incremental fashion. Thus, the learning algorithm will converge to near-optimal values for each of the n nodes in $\mathbf{X}$ and the fuzzy inference/defuzzification methods generalize this near-optimal information to the rest of $\mathbf{X}$.

IV. Experimental Results

The following applications illustrate the capabilities of the controller. All experiments were run on PCs using 32-bit DOS programs developed with the Borland C/C++ compiler. We have shown in previous research [9] that the choice of language and platform have a significant effect on the resulting control law learned when using reinforcement learning techniques that learn incrementally from experience.

A. Inverted Pendulum Balancing

A.1 Model of Process

A common benchmark problem for nonlinear controllers is the inverted pendulum problem. It is one of the simplest inherently unstable systems [20] and has a broad base for comparison throughout the literature. The system is described as follows:

Inputs: State vector—$\mathbf{x} = [\theta, \Delta\theta]$.

Outputs: Force applied (in Newtons)—$\mathbf{u} = [F]$ where $F > 0$ denotes a force applied in the positive x direction.

Equations of Motion: These equations serve only to simulate the system and are not used in the derivation of the control law:

$$\ddot{\theta} = \frac{g \sin\theta + \cos\theta \left(\frac{-F - m_p l \dot{\theta}^2 \sin\theta + \mu_p \text{sgn}(\dot{x})}{m_p + m_c} \right) - \frac{\mu_p \dot{\theta}}{m_p l}}{l\left(\frac{4}{3} - \frac{m_p \cos^2\theta}{m_p + m_c}\right)} \tag{IV.11}$$

$$\ddot{x} = \frac{F + m_p l \left(\dot{\theta}^2 \sin\theta - \ddot{\theta}\cos\theta\right) - \mu_c \text{sgn}(\dot{x})}{m_p + m_c} \tag{IV.12}$$

where the variables are defined as

g	acceleration due to gravity
m_c	mass of the cart (kg)
m_p	mass of the pole (kg)
l	half-length of pole (m)
μ_c	coefficient of friction for cart (N)
μ_p	coefficient of friction for pole (N)

with values shown in Table I.

$g = 9.8\text{m/s}^2$	$m_c = 1.0\text{kg}$
$m_p = 0.1\text{kg}$	$l = 0.5\text{m}$
$\mu_c = 0.0\text{N}$	$\mu_p = 0.0\text{N}$

TABLE I
PARAMETERS OF INVERTED PENDULUM SYSTEM SIMULATION.

Success and Failure: The setpoint for the inverted pendulum problem is $[0, 0]$. Failure occurs at any of the following conditions:

$$\begin{aligned} |\,\theta\,| &> 12^\circ \\ |\,\Delta\theta\,| &> 25^\circ/\text{s} \end{aligned}$$

A.2 Controller Performance

The inverted pendulum experiments were replicated using various initial random number seeds. The learning algorithm was allowed to continue online for 50,000 time steps of 0.01 seconds each. Using the TD method for the learning algorithm, the control surface for one experiment is shown in Figure IV.2 and the resulting trajectory from initial state $[10, 0]$ is shown in Figure IV.3. Using

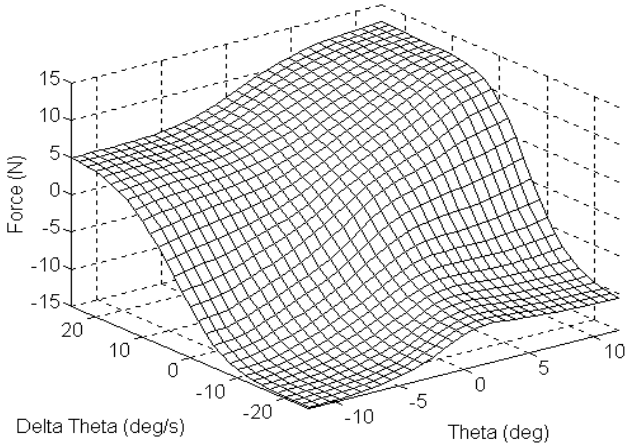

Fig. IV.2. Control surface using TD-based controller on the inverted pendulum.

the Q-learning method for the learning algorithm, the objective is to minimize IAE via Q-learning. The control surface for one experiment is shown in Figure IV.4 and the resulting trajectory from initial state $[10, 0]$ is shown in Figure IV.5.

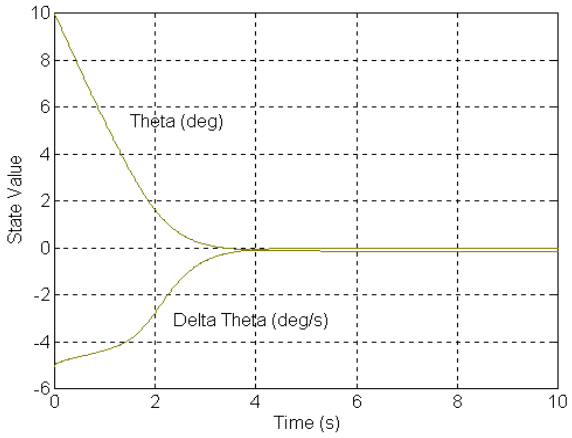

Fig. IV.3. Transient response using TD-based controller on the inverted pendulum.

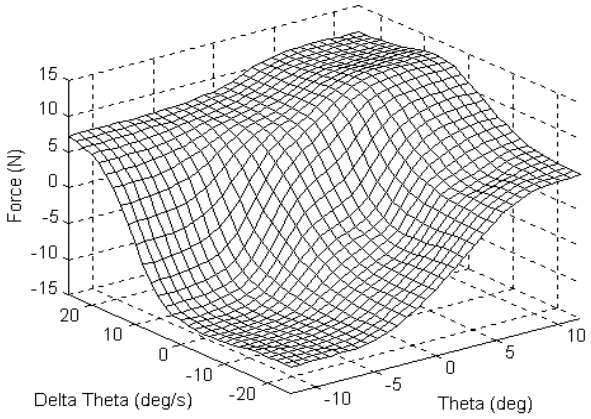

Fig. IV.4. Control surface using Q-learning-based controller on the inverted pendulum.

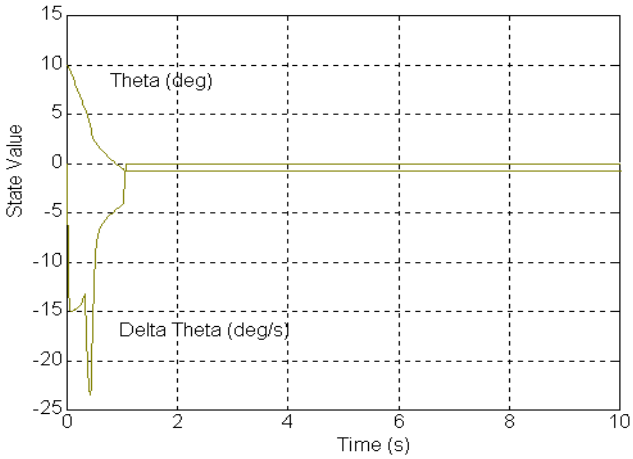

Fig. IV.5. Transient response using Q-learning-based controller on the inverted pendulum.

B. Power System Stabilization

B.1 Model of Process

The power system stabilization problem represents an underdetermined system that can be approached as either a multiple-input/single-output (MISO) or multiple-input/multiple-output (MIMO) control problem [13], [14], [15], [16], [18], [21]. The system considered here is composed of a synchronous machine with an exciter and a stabilizer connected to an infinite bus. The dynamics of the synchronous machine can be expressed as follows using the linearized incremental model [17]. The system is described as follows:

Inputs: State vector—$\mathbf{x} = [\omega, \Delta\omega]$.

Outputs: Control voltage—$\mathbf{u} = [u]$.

Equations of Motion: These equations serve only to simulate the system and are not used in the derivation of the control law:

$$\Delta\dot{\omega} = \frac{1}{M}(\Delta T_m - \Delta T_e - \Delta T_L - D\Delta\omega) \quad \text{(IV.13)}$$

$$\Delta\dot{\delta} = 377\Delta\omega \quad \text{(IV.14)}$$

$$\Delta T_e = K_e\Delta\delta + K_2\Delta e_q \quad \text{(IV.15)}$$

$$\Delta\dot{e}_q = \frac{1}{K_3T_{de}}(K_3\Delta e_{fd} - K_3K_4\Delta\delta - \Delta e_q) \quad \text{(IV.16)}$$

$$\Delta V_t = K_5\Delta\delta + K_6\Delta e_q \quad \text{(IV.17)}$$

$$\Delta\dot{V}_f = \frac{1}{T_F}(K_f\Delta\dot{e}_{fd} - \Delta V_F) \quad \text{(IV.18)}$$

$$\Delta\dot{e}_{fd} = \frac{1}{T_E}(\Delta V_A - K_E\Delta e_{fd}) \quad \text{(IV.19)}$$

$$\Delta\dot{V}_A = \frac{1}{T_A}(K_A\Delta V_{ref} - K_A\Delta V_F + K_Au - K_AK_6\Delta e_q - K_AK_5\Delta\delta - \Delta V_A) \quad \text{(IV.20)}$$

$$|\,u\,| \leq u_{\max} \quad \text{(IV.21)}$$

where

V_{ref}	constant reference input voltage
ΔV_t	terminal voltage change,
ΔV_o	infinite bus voltage change
Δe_{fd}	equivalent excitation voltage change
Δe_q	q-axis component voltage behind transient reactance change
ΔV_F	stabilizing transformer voltage change
u	stabilizer output
ΔT_m	mechanical input change
ΔT_e	energy conversion torque change
ΔT_L	load demand change
$\Delta\delta$	torque angle deviation,
$\Delta\omega$	angular velocity deviation
K_A, K_E	voltage regulator gains
T_A, T_E	voltage regulator time constants
K_F	stabilizing transformer gain
T_F	stabilizing transformer time constant
$K_1, \ldots, K_6$	constants of the linearized model of synchronous machine
T_{do}	d-axis transient open circuit time constant
M	inertia coefficient
D	damping coefficient
T_s	sampling period

with values for the above parameters given in Table II below.

$K_1 = 1.4479$	$K_2 = 1.3174$	$K_3 = 0.3072$
$K_4 = 1.8050$	$K_5 = 0.0294$	$K_6 = 0.5257$
$K_A = 400$	$T_F = 1.0$	$T_A = 0.05$
$D = 0$	$T_{do} = 5.9$	$K_E = -0.17$
$M = 4.74$	$T_E = 0.95$	$K_F = 0.025$
$\Delta T_m = 0$	$\Delta V_{ref} = 0$	$T_s = 0.01$

TABLE II
PARAMETERS OF POWER SYSTEM SIMULATION.

B.2 Controller Performance

The power system stabilization experiments were replicated using various initial random number seeds. The learning algorithm was allowed to continue online for 250,000 time steps of 0.01 seconds each. Using the TD method for the learning algorithm, the control surface for one experiment is shown in Figure IV.6 and the resulting trajectory from initial state $[0, 0]$ is shown in Figure IV.7.

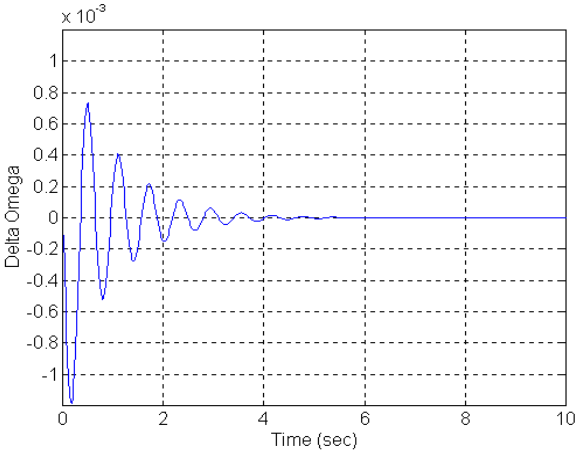

Fig. IV.6. Control surface using TD-based controller on the power system stabilization problem.

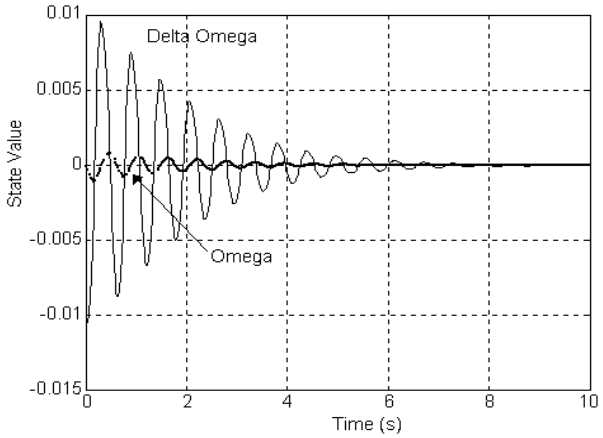

Fig. IV.7. Transient response using TD-based controller on the power system stabilization problem.

C. Tethered Satellite System Retrieval

C.1 Model of Process

The tethered satellite system problem represents a highly nonlinear control problem with a five state variables, which is considerably more than the usual test problem found in the literature. A tethered system is any two or more bodies connected by a long thin structure. The system focused on in this example is the deployment, station-keeping, and retrieval of a target satellite from the Space Shuttle. With a fixed-length tether for systems in the 'station-keeping' phase, the equations of motion are still complex. With a variable length tether—i.e., for systems in the deployment or retrieval phase—the equations of motion are further complicated by time-varying coefficients. The system is described as follows:

Inputs: State vector—$\mathbf{x} = \left[\theta, \dot{\theta}, \phi, \dot{\phi}, l\right]$.

Outputs: Tether length rate—$\mathbf{u} = \left[\dot{l}\right]$.

Equations of Motion: The model used in our simulation is a simplification of the actual dynamics [6]. There are two coordinate systems in the model—the orbital axes and the tether axes. The orbital axes, XYZ, are such that the positive Z direction points to the center of the Earth, the positive X axis points in the direction of the trajectory, and thus the Y axis is perpendicular to the XZ plane. The tether axes, xyz, are such that the z axis is aligned with the tether and have the same origin as XYZ. The system attitude is described by

- In-plane motion or pitch: Rotation θ about the Y axis.
- Out-of-plane motion or roll: Rotation ϕ about the instantaneous X axis.

These equations serve only to simulate the system and are not used in the derivation of the control law:

$$\begin{aligned} Q_\theta(t) &= \left(M_s + \frac{m_c}{3}\right) l^2 \left(\ddot{\theta} - \ddot{\theta_p}\right) c_\phi^2 \\ &\quad -2\left(M_s + \frac{m_c}{3}\right) l^2 \left(\dot{\theta} - \dot{\theta_p}\right) \dot{\phi} s_\phi c_\phi \\ &\quad +3\left(M_s + \frac{m_c}{3}\right) l^2 \left(\frac{\mu_E}{R_o^2}\right) c_\phi^2 s_\phi c_\theta \\ &\quad +2\left(M_s + \frac{m_c}{2}\right) l\dot{l} \left(\dot{\theta} - \dot{\theta_p}\right) c_\phi^2 \quad \text{(IV.22)} \\ Q_\phi(t) &= \left(M_s + \frac{m_c}{3}\right) l^2 \left(\dot{\theta} - \dot{\theta_p}\right)^2 s_\phi c_\phi \\ &\quad + \left(M_s + \frac{m_c}{3}\right) l^2 \ddot{\phi} \\ &\quad +3\left(M_s + \frac{m_c}{3}\right) l^2 \left(\frac{\mu_E}{R_o^2}\right) c_\theta^2 s_\phi c_\phi \\ &\quad +2\left(M_s + \frac{m_c}{2}\right) l\dot{l}\dot{\phi} \quad \text{(IV.23)} \end{aligned}$$

where s_ϕ, s_θ, c_ϕ and c_θ are the `sin` and `cosine` functions of the respective state variables, M_s and m_c are the sub-satellite and instantaneous tether mass, Q_ϕ and Q_θ are generalized external forces, R_o is the orbit radius, θ_p is the *true anomaly* and μ_E is the Earth gravitational constant. The values for the parameters used in the simulation are

$M_s = 150\text{kg}$	$\rho A = 0.0015\text{kg/m}$

TABLE III
Parameters of Tethered Satellite System Simulation.

The control variable is the deployed tether length rate, $\dot{l}$. The state vector is defined as

$$\vec{X} = \{x_1, x_2, x_3, x_4, x_5\}^T \equiv \left\{\theta, \dot{\theta}, \phi, \dot{\phi}, l\right\}^T. \quad \text{(IV.24)}$$

C.2 Controller Performance

The tethered satellite system experiments were replicated using various initial random number seeds. The learning algorithm was allowed to continue online for 500,000 time steps of 1 second each. The controller performed satisfactorily in these preliminary experiments in that it learned to retrieve the satellite from 100km to almost 10km before failure. The physical characteristics of the system make the retrieval phase near the Shuttle very nonlinear and dangerous to the crew. Using the TD method for the learning algorithm, the controller learned the characteristic of the optimal control, namely the 'fishing' motion of sending the satellite out and then reeling it back in. The control variable during retrieval is shown in Figure IV.8 and the resulting tether length starting at 100km is shown in Figure IV.9.

V. Discussion

This controller possesses the capability to learn the control laws to various set-point regulator problems of practical interest. The inverted pendulum provides a common benchmark and the power system stabilization problem and tethered satellite system retrieval problem are of practical interest. Both TD and Q-learning are useful tools for learning the control law online. Q-learning

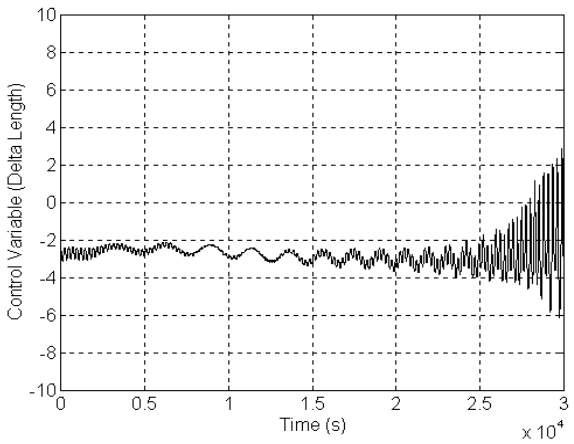

Fig. IV.8. Control variable (length) during the retrieval of the satellite.

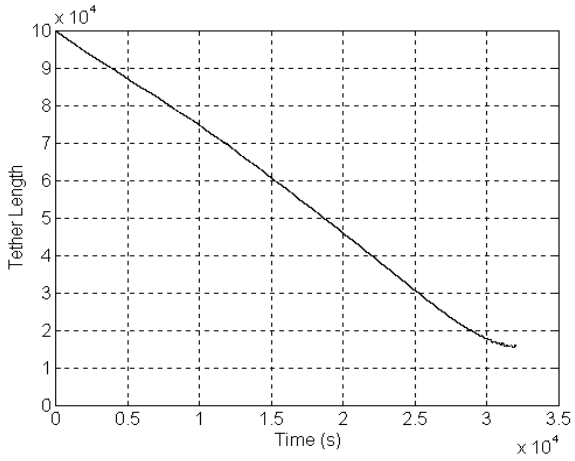

Fig. IV.9. Tether length during the retrieval of the satellite.

provides the ability to incorporate performance measures as goals (such as minimizing IAE) and thereby a method of learning some type of near-optimal control. Research into various tools for the enhancement of this controller, especially the learning phase, is in progress

VI. Acknowledgments

This research is sponsored in part by the National Science Foundation under Grant ECS-9216004, the Electric Power Research Institute under Grant RP 8030-16, and the National Aeronautics and Space Administration under Grant NAG 9-726.

References

[1] Barto, A.G., Bradtke, S.J., and Singh, S.P. Learning to act using real-time dynamic programming. *Artificial Intelligence*, **72:1-2**, 1995, 81-138.

[2] Berenji, H.R. A reinforcement learning-based architecture for fuzzy logic control. *International Journal of Approximate Reasoning*, **6:2**, 1992, 267-292.

[3] Berenji, H.R. Fuzzy *Q*-learning: a new approach for fuzzy dynamic programming. *Proceedings of the Third IEEE Conference on Fuzzy Systems*, Orlando, FL, June 26-29, 1994, 486-491.

[4] Berenji, H.R. and Khedkar, P. Learning and tuning fuzzy logic controllers through reinforcements. *IEEE Transactions on Neural Networks*, 1992, **3:5**, 724-740.

[5] Berenji, H.R. and Ralescu, A.L. Fuzzy reinforcement learning and dynamic programming. *Proceedings of the Second Workshop on Fuzzy Logic in Artificial Intelligence*, Chamberry, France, August 28, 1993, 1-9.

[6] Boschitsch, A. and Bendiksen, O.O. Nonlinear control laws for tethered satellites. *Advances in Astronautical Sciences*, **62**, 1986, 257-276.

[7] Cheng, S.-J., Malik, O.P and Hope, G.S. Self-tuning stabilizer for a multimachine power system. *IEE Proceedings*, **133-C:4**, 1986, 176-185.

[8] Esogbue, A.O. Optimal clustering of fuzzy data via fuzzy dynamic programming. *Fuzzy Sets and Systems*, **18**, 1986, 283-298.

[9] Esogbue, A.O. and Hearnes, W.E. Constructive experiments with a new fuzzy adaptive controller. *Proceedings of NAFIPS/IFIS/NASA Conference*, San Antonio, TX, December 18-21, 1994, 377-380.

[10] Esogbue, A.O. and Murrell, J.A. A fuzzy adaptive controller using reinforcement learning neural networks. *Proceedings of IEEE 2nd International Fuzzy Systems Conference*, San Francisco, CA, March 28-April 1, 1993, 178-183.

[11] Esogbue, A.O. and Murrell, J.A. Advances in fuzzy adaptive control. *Computers & Mathematics with Applications*, **27:9-10**, 1994, 29-35.

[12] Esogbue, A.O. and Song, Q. Optimal defuzzification and applications. *Proceedings of the International Joint Conference on Information Science, Fourth Annual Conference on Fuzzy Theory & Technology*, Wrightsville Beach, NC, September 28-October 1, 1995.

[13] Esogbue, A.O., Song, Q. and Hearnes, W.E. Application of a self-learning fuzzy-neuro controller to the power system stabilization problem. *Proceedings of the World Congress on Neural Networks*, Washington, DC, July 17-21, 1995, 699-702.

[14] Ghandakly, A.A. and Farhoud, A.M. A parametrically optimized self-tuning regulator for power system stabilizers. *IEEE Transactions on Power Systems*, **7:3**, 1992, 1245-1250.

[15] Hassan, M.A.M., Malik, O.P and Hope, G.S. A fuzzy logic based stabilizer for a synchronous machine. *IEEE Transactions on Energy Conversion*, **6:3**, 1991, 407-413.

[16] Hiyama, T. and Sameshima, T. Fuzzy logic control scheme for on-line stabilization of multi-machine power system. *Fuzzy Sets and Systems*, **39**, 1991, 181-194.

[17] Hsu, Y.-Y. and Hsu, C.-Y. Design of a proportional-integral power system stabilizer. *IEEE Transactions on Power Systems*, **PWRS-1:2**, 1986, 46-53.

[18] Hsu, Y.-Y. and Cheng, C.-H. A fuzzy controller for generator excitation control. *IEEE Transactions on Systems, Man and Cybernetics*, **23:2**, 1993, 532-539.

[19] Lin, C.-T. and Lee, C.S.G. Reinforcement structure/parameter learning for neural-network-based fuzzy logic control systems. *IEEE Transactions on Fuzzy Systems*, **2:1**, 1994, 46-63.

[20] Murrell, James A. A statistical fuzzy associative learning approach to intelligent control. Ph.D. Thesis, School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA, 1993.

[21] Shi, J., Herron, L.H. and Kalam, A. A fuzzy logic controller applied to power system stabilizer for a synchronous machine power system. *Proceedings of IEEE Region 10 Conference, Tencon 92*, Melbourne, Australia, November 11-13, 1992.

[22] Sutton, R.S. Learning to predict by the method of temporal differences. *Machine Learning*, **3**, 1988, 9-44.

[23] Watkins, C.J.C.H. *Learning from delayed rewards*. Ph.D. Thesis, Cambridge University, Cambridge, England, 1989.

[24] Watkins, C.J.C.H. and Dayan, P. Q-Learning. *Machine Learning*, **8**, 1992, 279-292.