Nonlinear adaptive flight control using incremental approximate dynamic programming and output feedback

Zhou, Y; van Kampen, EJ; Chu, QP

DOI
10.2514/6.2016-0360

Publication date
2016

Document Version
Accepted author manuscript

Published in
Proceedings of the AIAA guidance, navigation, and control conference

This is an Accepted Manuscript of an article published in: AIAA Guidance, Navigation, and Control Conference, 4-8 Januari 2016, San Diego, United States. ISBN: 978-1-62410-389-6
Available online: https://arc.aiaa.org/doi/10.2514/6.2016-0360

Nonlinear Adaptive Flight Control Using Incremental Approximate Dynamic Programming and Output Feedback

Ye Zhou*, Erik-Jan van Kampen† and QiPing Chu‡
Delft University of Technology, 2629HS Delft, The Netherlands

A self-learning adaptive flight control for nonlinear systems allows a reliable, fault-tolerant and effective operation of complex flight vehicles in a dynamic environment. Approximate dynamic programming provides a model-free control design for nonlinear systems with complex design processes and non-guaranteed closed-loop convergence properties. Linear approximate dynamic programming systematically applies a quadratic cost-to-go function and greatly simplifies the design process of approximate dynamic programming. This paper presents a newly developed self-learning adaptive control method called incremental approximate dynamic programming for nonlinear unknown systems. It combines the advantages of linear approximate dynamic programming methods and incremental control techniques to generate a near-optimal control without a priori knowledge of the system model. In this paper, two incremental approximate dynamic programming algorithms with the direct availability of full states and with only the availability of system outputs have been developed. Both algorithms have been applied to a nonlinear aerospace related simulation model. The simulation results demonstrate that both model-free adaptive control algorithms improve the closed-loop performance of the nonlinear system, while keeping the design process simple and systematic as compared to conventional approximate dynamic programming algorithms.

I. Introduction

Safety, which is of paramount importance in aviation, depends to a very large extent on the flight control system of contemporary air vehicles. Maintaining functionality of control law in case of unforeseen failure, damages or icing requires a high level of robustness and fault-tolerance. This challenging problem can be transformed to an equivalent situation of controlling a complex flying vehicle without sufficient knowledge of the system dynamics which may even be nonlinear. Until recent decades, adaptive control methods allow certain levels of robustness and fault-tolerance to be achieved. These model-based methods in some form or another rely on on-line identification of air vehicles' dynamic behavior and adaptation of control laws when necessary.

On-line identification of unknown dynamical systems is not a trivial task. The reasons can be roughly summarized as follows: 1) Aerodynamic models of air vehicles are complex. They are highly nonlinear and uncertain especially in failure cases. Model identification is an optimization process, through which a relevant a priori model structure with unknown parameters needs to be estimated.[1] 2) When the model structure of the damaged aircraft is highly nonlinear, the parameter estimation will be a global optimization problem.[2] 3) Model parameter identification requires system excitations. However, when an aircraft experiences failures (especially aerodynamic failures), the reduced safe flight envelop of the damaged air vehicle is unknown. Due to this, additional inputs for identifying system will probably be highly risky.[3] 4) On-line model identification has to be sufficiently quick and smooth for adaptive control. Fluctuations of identified parameters during the

*PhD student, Control and Operation Department, Aerospace Engineering, Delft University of Technology, AIAA student member.
†Assistant Professor, Control and Operation Department, Aerospace Engineering, Delft University of Technology, AIAA member.
‡Associate Professor, Control and Operation Department, Aerospace Engineering, Delft University of Technology, AIAA member.

convergence phase may not be allowed especially in the case of failure.[1] 5) Indirect and modular adaptive controls with system identification offer no guarantee of stability in general.[1,4]

Direct and integrated adaptive control laws are theoretically stable due to their Lyapunov stability analysis based designs.[5–7] These design methods need, however, function approximators for constructing the unknown dynamic model,[6,7] and optimization processes for adapting these control systems.[8,9] Sliding mode control is a method that forces the state trajectories of a dynamic system to slide along a predefined subspace of the state space, within which a sliding mode can be realized and the equilibrium can be reached.[10]

Alternatively, model-free adaptive control approaches are worth well to be investigated for fault-tolerant flight control. In recent years, Reinforcement Learning (RL) controllers have been proposed to solve nonlinear, optimal control problems. RL is learning what actions to take to affect the state and to maximize the numerical reward (or minimize the cost) signal by interacting with the environment, and to achieve a goal ultimately. This is not defined by characterizing learning methods, but by learning problems which can be described as optimal control problems of Markov Decision Processes (MDPs).[11,12] This method links bio-inspired artificial intelligence techniques to the field of control to overcome some of the limitations and problems in most control methods demanding precise models. Nevertheless, traditional RL solving the optimality problem is an off-line method by using an n-dimensional look-up table for all possible state vectors which may cause the "curse of dimensionality" issues.[11]

To tackle the "curse of dimensionality", numerical methods, such as approximate dynamic programming (ADP), have been developed to solve the optimality problems forward online,[13,14] by applying a function approximator with parameters to approximate the value/cost function. Value/cost functions are essential for RL methods.[15] In many high dimensional or continuous state space problems, the value/cost function is represented by a function approximator, such as a linear combination of the states/features, splines, or a neural network. A Universal Function Approximator (UFA) enhances the ability of generalization and can output an accurate estimate of any state value/cost in the state space to an arbitrary degree of precision. This single function exploits its structure, cache information from learning the value/cost of observed states, generalize to similar, previously unseen states, and ultimately can represent the utility of any state in the state space to achieve the agent's overall goal.

However, learning a UFA poses challenges due to its complexity, which is more considerable than conventional function approximation; representing a UFA may require a rich function approximator, such as a pretty deep, non-parameterized neural network. Searching for an applicable structure and parameters of the network is a global optimization problem as neural networks are highly nonlinear in general. For the special case when the dynamics of the system are linear, Dynamic Programming (DP) gives a complete and explicit solution to the problem, because the one-step state cost and the value/cost function in this case are quadratic.[16] For the general nonlinear control problem, DP is difficult to carry out and ADP designs are not systematic.[15]

Considering the design challenges mentioned above, trade-off solutions which may lead to simple and systematic designs are extremely attractive. In this paper, an incremental Approximate Dynamic Programming (iADP) model-free adaptive control approach is developed for nonlinear systems. This is called a model-free approach, because it does not need any a priori model information at the beginning of the algorithm nor on-line identification of nonlinear systems, but only the on-line identified linear model. This control approach was inspired by the ideas and solutions given by several articles[16–20] . It starts with the selection of the value/cost function in a systematic way,[16] and follows by the Linear Approximate Dynamic Programming (LADP) model-free adaptive control approach.[17] As the plant to be controlled in this paper is nonlinear, the iADP is therefore developed based on the linearized incremental model of the original nonlinear system.[18–20]

The incremental form of a nonlinear dynamic system is actually a linear time-varying approximation of the original system assuming sufficiently high sample rate for discretization. This form has been successfully applied to design controllers such as Incremental Nonlinear Dynamic Inversion (INDI) and Incremental Backstepping (IBS) for nonlinear systems.[18–20] Although these nonlinear control methods have greatly simplified the design process and reduced model dependence in the control system, optimization or synthesis of designed closed-loop systems has not been addressed. Combining LADP and the incremental form of the system to be controlled leads to a new nonlinear adaptive control algorithm iADP. It remains the advantages of LADP with a systematic formulation of value/cost function approximations for nonlinear systems, while keeping the closed-loop system being optimized.

Classical ADP methods assume that the system is fully observable and that the observed states obey a Markov process. The merits of model-free processes, adaptability to the environment, and efficiency of

resource usage make ADP controllers suitable and effective in the field of adaptive flight control. Unfortunately, in the real world, the agent might not have perfect perception of states of the environment.[21] In most cases, the agent "observes" the state of the environment, but these observations may be noisy and cannot provide sufficient information.

The problems of partial/imperfect information and unmeasurable state vector estimation are very challenging and demanded to be solved in numerous applications. Many researches have already taken presence of stochastic, time-varying wind disturbance into account as a general problem in practical navigation and guidance control.[22,23] Despite from that, parametrized output feedback controllers have been designed to deal with problems without full state information and to achieve finite time stability based on observers.[24–26] For example, interval observers have been introduced to cope with uncertainties that are known to characterize some classes of systems, and have been designed for both linear time-varying systems and a class of nonlinear time-varying systems.[27,28] Another recent research have proposed an angular velocity observer with a smooth structure to ensure continuity of all estimated states.[29] However, these methods still need a priori knowledge or/and assumption of the system model structure.

Other than that, output feedback approximate dynamic programming algorithms[17] have been proposed, as opposed to full state feedback, to tackle problems without direct state observation. These algorithms do not require any a priori knowledge of the system or engineering knowledge to design control parameters, or even a separate observer. However, these algorithms are derived for affine in control input linear time-invariant (LTI) deterministic systems, and require persistently exciting probing noise and a discounted cost function for convergence.

In this paper, two algorithms are presented for the proposed iADP method. First, an algorithm combining ADP and the incremental approach with direct availability of full state observation is developed.[30] Second, an iADP algorithm based on output feedback control approach is designed by applying the output and input measurement to reconstruct the full state.

II. Background

A. Reinforcement learning methods

Reinforcement learning is learning from experience denoting by a reward or a punishment, which is inspired by animal behaviors. From the standpoint of artificial intelligence, *value functions* are used and rewards are always maximized.[11,31] In control engineering and in most of this paper, *cost-to-go/ cost functions* are used, thus, a reward item diminishes the cost while a punishment increases the cost.[31,32]

The methods for solving RL problems can be classified into three categories: Dynamic Programming (DP), Monte Carlo methods (MC) and Temporal-Difference learning (TD).[11] Different methods have their advantages and disadvantages. DP methods are well developed mathematically to compute optimal policies, but require a perfect model of the systems behavior and the environment as an MDP. MC methods do not require a priori knowledge of the environment's dynamics and are conceptually simple, however, the value/cost estimates and policies are changed only upon the completion of an episode. TD methods, as a group of relatively central and novel methods of RL, require no model and are fully incremental. Actually, TD learning is a combination of MC ideas and DP ideas. Like MC methods, TD methods can learn directly from only experience without a model; and like DP, they update estimates based in part on other learned estimates without waiting for a final outcome.

RL algorithms can also be categorized by how the optimal policy is obtained.[33,34] *Policies* are what the plant/system depends on to decide what actions to take when the system is in some state. Policy iteration (PI) algorithms evaluate the current policy to obtain the value/cost function, and improve the policy accordingly. Value iteration (VI) algorithms find the optimal value/cost function and the optimal policy. Policy search (PS) algorithms search for the optimal policy directly by using optimization algorithms.

Temporal difference methods are those RL algorithms which have most impact on RL based adaptive control methods. This group of methods provides an effective way to decision making or control problems when optimal solutions are difficult to obtain or even unavailable analytically.[35] TD method can be defined basically in an iterative estimation update form as follows:

$$\widehat{J}(\mathbf{x}_t)_{new} = \widehat{J}(\mathbf{x}_t) + \alpha[r_{t+1} + \gamma\widehat{J}(\mathbf{x}_{t+1}) - \widehat{J}(\mathbf{x}_t)] \tag{1}$$

where, $\widehat{J}(\mathbf{x}_t)$ is the estimation of the cost-to-go function for state $\mathbf{x}$ at time t; α is a *step-size parameter*, which can be a constant or change as time step increases, e.g. $\alpha = 1/t$; $\gamma \in [0, 1]$ is a parameter called

the *discounted rate* or the *forgetting factor*. The target for update is $r_{t+1} + \gamma\widehat{J}(x_{t+1})$, and the *TD error* is $r_{t+1} + \gamma\widehat{J}(x_{t+1}) - \widehat{J}(x_t)$. TD methods can be classified into three most popular categories:[11]

- SARSA is an on-policy TD control method. This method learns an action-value function and considers transitions from a state-action pair to the next pair.
- Q-learning is an off-policy method. It is similar to SARSA, but the learned action-value function directly approximates the optimal action-value function, independent of the policy being followed and without taking exploration into account.
- Actor-Critic methods are on-policy TD methods that commit to always exploring and tries to find the best policy that still explores.[36] They use TD error to evaluate a selected action to strengthen or weaken the tendency of selecting the action for the future. Additionally, they have a separate memory structure to explicitly represent the policy independent of the value/cost function.

B. Approximate Dynamic Programming and Partial Observability

Traditional DP method[11] is an off-line method knowing the system model and solving the optimality problem backward by using a n-dimensional lookup table for all possible states vector in $\mathcal{R}^n$ causing "curse of dimensionality" problems. To tackle the "curse of dimensionality", numerical methods, such as approximate dynamic programming (ADP), are the best to solve the optimality problems forward online.[13,14]

The core of ADP as an adaptive optimal controller is to solve the Bellman equation or its related recurrence equations. ADP methods use a universal approximator $\widehat{J}(\mathbf{x}_t, parameters)$ instead of $\widehat{J}(\mathbf{x}_t)$ to approximate the cost-to-go function J. Besides, ADP algorithms are good for hybrid design such as problems combining continuous and discrete variables.

- Approximate value iteration (AVI) is a VI algorithm used in the situation that the number of states is too large for an exact representation. AVI algorithms use a sequence of functions J_n iterative according to $J_{n+1} = ALJ_n$, where, L denotes the Bellman operator; A denotes the operator projecting onto the space of defined approximation functions.
- Approximate policy iteration (API) generalizes the PI algorithm by using function approximation method. This algorithm is built up by iteration of two steps: approximate policy evaluation step which generates an approximated cost-to-go function J_n for a policy π_n, and policy improvement step, which generates a new policy with respect to the cost-to-go function approximation J_n greedily.[21]

ADP methods generally assume that the system is fully observable. Thus, the optimal action can be chosen in terms of the full knowledge of system states. However, the agents often try to control the systems without enough information to infer its real states.[21] If the measurement of the system is not the direct full states, e.g. some internal states are not measurable or a few states are coupled, the system is partially observable. These types of methods dealing with deterministic system are often referred to as output feedback. The system still needs to be observable, which means that the full state can be reconstructed with the observations over a long enough time horizon. With some a priori knowledge about the system, the unmeasurable internal states can be reconstructed by using output observation and system information, and the system is observable. For model-free method, the system is observable when the observability matrix has a full column rank. For the stochastic system, the observed states are assumed to obey a Markov process, which means next state, $\mathbf{x}_{t+1}$, is decided by a probability distribution depending on current state, $\mathbf{x}_t$, and action to take, $\mathbf{u}_t$. The partially observable Markov decision process (POMDP) framework can be used to decide how to act in partially observable sequential decision processes.[37–39]

III. Incremental Approximate Dynamic Programming

Incremental methods are able to deal with the nonlinearities of systems. These methods compute the required control increment at a certain moment using the conditions of the system in the instant before.[19] Aircraft models are highly nonlinear and can be generally given as follows:

$$\dot{\mathbf{x}}(t) = f[\mathbf{x}(t), \mathbf{u}(t)], \tag{2}$$

$$\mathbf{y}(t) = h[\mathbf{x}(t)], \tag{3}$$

where Eq. 2 is the *system dynamic equation*, in which $f[\mathbf{x}(t), \mathbf{u}(t)] \in \mathcal{R}^n$ provides the physical evaluation of the state vector over time, Eq. 3 is the *output (observation) equation*, which can be measured using sensors, and $h[\mathbf{x}(t)] \in \mathcal{R}^p$ is a vector denoting the measured output.

The system dynamics around the condition of the system at time t_0 can be linearized approximately by using the first-order Taylor series expansion:

$$\begin{aligned}\dot{\mathbf{x}}(t) &\simeq f[\mathbf{x}(t_0), \mathbf{u}(t_0)] + \frac{\partial f[\mathbf{x}(t), \mathbf{u}(t)]}{\partial \mathbf{x}(t)}|_{\mathbf{x}(t_0),\mathbf{u}(t_0)}[\mathbf{x}(t) - \mathbf{x}(t_0)] + \frac{\partial f[\mathbf{x}(t), \mathbf{u}(t)]}{\partial \mathbf{u}(t)}|_{\mathbf{x}(t_0),\mathbf{u}(t_0)}[\mathbf{u}(t) - \mathbf{u}(t_0)] \\ &= \dot{\mathbf{x}}(t_0) + F[\mathbf{x}(t_0), \mathbf{u}(t_0)][\mathbf{x}(t) - \mathbf{x}(t_0)] + G[\mathbf{x}(t_0), \mathbf{u}(t_0)][\mathbf{u}(t) - \mathbf{u}(t_0)],\end{aligned} \tag{4}$$

where $F[\mathbf{x}(t), \mathbf{u}(t)] = \frac{\partial f[\mathbf{x}(t),\mathbf{u}(t)]}{\partial \mathbf{x}(t)} \in \mathcal{R}^{n\times n}$ is the *system matrix* of the linearized model at time t, and $G[\mathbf{x}(t), \mathbf{u}(t)] = \frac{\partial f[\mathbf{x}(t),\mathbf{u}(t)]}{\partial \mathbf{u}(t)} \in \mathcal{R}^{n\times m}$ is the *control effectiveness matrix* of the linearized model at time t.

We assume that the control inputs, states, and state derivatives of the system are measurable. Under this assumption, the model around time t_0 can be written in the incremental form:

$$\Delta\dot{\mathbf{x}}(t) \simeq F[\mathbf{x}(t_0), \mathbf{u}(t_0)]\Delta\mathbf{x}(t) + G[\mathbf{x}(t_0), \mathbf{u}(t_0)]\Delta\mathbf{u}(t). \tag{5}$$

This current linearized incremental model is identifiable by using least squares (LS) techniques.

A. Incremental Approximate Dynamic Programming Based on Full State Feedback

The physical systems are generally continuous, but the data we collect are discrete samples. We assume that the control system has a constant high sampling frequency. With this constant data sampling rate, the nonlinear system can be written in a discrete form:

$$\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t), \tag{6}$$

$$\mathbf{y}_t = h(\mathbf{x}_t), \tag{7}$$

where $f(\mathbf{x}_t, \mathbf{u}_t) \in \mathcal{R}^n$ provides the *system dynamics*, and $h(\mathbf{x}_t) \in \mathcal{R}^p$ is a vector denoting the measuring system.

When the system has a direct availability of full state observation, the output equation can be written as

$$\mathbf{y}_t = \mathbf{x}_t. \tag{8}$$

By taking the Taylor expansion, we can get the system dynamics linearized around $\mathbf{x}_0$:

$$\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t) \simeq f(\mathbf{x}_0, \mathbf{u}_0) + \frac{\partial f(\mathbf{x}, \mathbf{u})}{\partial \mathbf{x}}|_{\mathbf{x}_0,\mathbf{u}_0}(\mathbf{x}_t - \mathbf{x}_0) + \frac{\partial f(\mathbf{x}, \mathbf{u})}{\partial \mathbf{u}}|_{\mathbf{x}_0,\mathbf{u}_0}(\mathbf{u}_t - \mathbf{u}_0). \tag{9}$$

When Δt is sufficiently small, $\mathbf{x}_{t-1}$ approximates $\mathbf{x}_t$. Thus, $\mathbf{x}_0, \mathbf{u}_0$ in Eq. 9 can be replaced by $\mathbf{x}_0 = \mathbf{x}_{t-1}$ and $\mathbf{u}_0 = \mathbf{u}_{t-1}$, and we obtain the discrete incremental form of this nonlinear system:

$$\mathbf{x}_{t+1} - \mathbf{x}_t \simeq F(\mathbf{x}_{t-1}, \mathbf{u}_{t-1})(\mathbf{x}_t - \mathbf{x}_{t-1}) + G(\mathbf{x}_{t-1}, \mathbf{u}_{t-1})(\mathbf{u}_t - \mathbf{u}_{t-1}), \tag{10}$$

$$\Delta\mathbf{x}_{t+1} \simeq F(\mathbf{x}_{t-1}, \mathbf{u}_{t-1})\Delta\mathbf{x}_t + G(\mathbf{x}_{t-1}, \mathbf{u}_{t-1})\Delta\mathbf{u}_t, \tag{11}$$

where $F(\mathbf{x}_{t-1}, \mathbf{u}_{t-1}) = \frac{\partial f(\mathbf{x},\mathbf{u})}{\partial \mathbf{x}}|_{\mathbf{x}_{t-1},\mathbf{u}_{t-1}} \in \mathcal{R}^{n\times n}$ is the *system matrix*, and $G(\mathbf{x}_{t-1}, \mathbf{u}_{t-1}) = \frac{\partial f(\mathbf{x},\mathbf{u})}{\partial \mathbf{u}}|_{\mathbf{x}_{t-1},\mathbf{u}_{t-1}} \in \mathcal{R}^{n\times m}$ is the *control effectiveness matrix* at time step $t-1$. Because of the high frequency sample data and slow-variant system, the current linearized model can be identified by using the collected data from previous M measurements.

To minimize the cost of the system approaching its goal, we first define the *one-step cost function* quadratically:

$$c_t = c(\mathbf{y}_t, \mathbf{u}_t, \mathbf{d}_t) = \widetilde{c}(\mathbf{y}_t, \mathbf{d}_t) + \mathbf{u}_t^T R \mathbf{u}_t = (\mathbf{y}_t - \mathbf{d}_t)^T Q (\mathbf{y}_t - \mathbf{d}_t) + \mathbf{u}_t^T R \mathbf{u}_t, \tag{12}$$

where $\widetilde{c}(\mathbf{y}_t, \mathbf{d}_t)$ represents a cost for the current outputs $\mathbf{y}_t$ approaching the desired outputs $\mathbf{d}_t$, Q and R are positive definite matrices.

If we only consider a stabilizing control problem, the desired outputs are zero. And the one-step cost function at time t can then be rewritten in a quadratic form, as shown below:

$$c_t = c(\mathbf{y}_t, \mathbf{u}_t) = \mathbf{y}_t^T Q \mathbf{y}_t + \mathbf{u}_t^T R \mathbf{u}_t. \tag{13}$$

For infinite horizons, the cost-to-go function is the cumulative future rewards from any initial state $\mathbf{x}_t$:

$$\begin{aligned} J^{\mu}(\mathbf{x}_t) &= \sum_{i=t}^{\infty} \gamma^{i-t} (\mathbf{y}_i^T Q \mathbf{y}_i + \mathbf{u}_i^T R \mathbf{u}_i) \\ &= (\mathbf{y}_t^T Q \mathbf{y}_t + \mathbf{u}_t^T R \mathbf{u}_t) + \gamma J^{\mu}(\mathbf{x}_{t+1}) \\ &= \mathbf{y}_t^T Q \mathbf{y}_t + (\mathbf{u}_{t-1} + \Delta \mathbf{u}_t)^T R (\mathbf{u}_{t-1} + \Delta \mathbf{u}_t) + \gamma J^{\mu}(\mathbf{x}_{t+1}), \end{aligned} \tag{14}$$

where μ is the *current policy* for this iADP algorithm. The optimal cost-to-go function for the *optimal policy* μ^* is defined as follows:

$$J^*(\mathbf{x}_t) = \min_{\Delta \mathbf{u}_t} \left[\mathbf{y}_t^T Q \mathbf{y}_t + (\mathbf{u}_{t-1} + \Delta \mathbf{u}_t)^T R (\mathbf{u}_{t-1} + \Delta \mathbf{u}_t) + \gamma J^*(\mathbf{x}_{t+1}) \right]. \tag{15}$$

And the control law (policy μ) can be defined as feedback control in an incremental form:

$$\Delta \mathbf{u}_t = \mu(\mathbf{u}_{t-1}, \mathbf{x}_t, \Delta \mathbf{x}_t). \tag{16}$$

The optimal policy at time t can be given by

$$\mu^* = \arg \min_{\Delta \mathbf{u}_t} \left[\mathbf{y}_t^T Q \mathbf{y}_t + (\mathbf{u}_{t-1} + \Delta \mathbf{u}_t)^T R (\mathbf{u}_{t-1} + \Delta \mathbf{u}_t) + \gamma J^*(\mathbf{x}_{t+1}) \right]. \tag{17}$$

When the dynamics of the system are linear, this problem is known as the linear-quadratic regulator (LQR) control problem. For this nonlinear case, the cost-to-go is the sum of quadratic values in the outputs and inputs with a forgetting factor. Thus, the state is the deterministic factor of the cost-to-go $J^{\mu}(\mathbf{x}_t)$, which should always be positive. In general, ADP uses a surrogate cost function approximating the true cost-to-go. The goal would be capturing its key attributes or features instead of accurately approximating the true cost-to-go. In many practical cases, even time-varying systems, simple quadratic cost function approximations are chosen so that the expectation step can be exactly carried out and the optimization problem evaluating the policy is reduced to be tractable.[16] A systematic cost function approximation that can be applied in our system in this paper is chosen to be quadratic in the state $\mathbf{x}_t$ for some symmetric, positive definite matrix P, as shown below:

$$\widehat{J}^{\mu}(\mathbf{x}_t) = \mathbf{x}_t^T P \mathbf{x}_t. \tag{18}$$

This quadratic cost function approximation has an additional, important benefit for this approximately convex state-cost system with a fixed minimum value. To be specific, this system has an optimal state when the state reaches the desired state (which is zero in regulator problem) and keeps it. The true cost function has many local minima elsewhere because of the nonlinearity of the system. On the other hand, this quadratic approximate cost function has only one local minimum which is the global one. Therefore, this quadratic form helps to prevent the policy from going into any other local minimum. The learned symmetric, positive definite P matrix is the guarantee of the progressive optimization of the policy.

The LQR Bellman equation for $\widehat{J}^{\mu}$ in the incremental form becomes

$$\begin{aligned} &\widehat{J}^{\mu}(\mathbf{x}_t) = \mathbf{y}_t^T Q \mathbf{y}_t + (\mathbf{u}_{t-1} + \Delta \mathbf{u}_t)^T R (\mathbf{u}_{t-1} + \Delta \mathbf{u}_t) + \gamma \mathbf{x}_{t+1}^T P \mathbf{x}_{t+1} \\ &= \mathbf{y}_t^T Q \mathbf{y}_t + (\mathbf{u}_{t-1} + \Delta \mathbf{u}_t)^T R (\mathbf{u}_{t-1} + \Delta \mathbf{u}_t) + \gamma (\mathbf{x}_t + F_{t-1} \Delta \mathbf{x}_t + G_{t-1} \Delta \mathbf{u}_t)^T P (\mathbf{x}_t + F_{t-1} \Delta \mathbf{x}_t + G_{t-1} \Delta \mathbf{u}_t). \end{aligned} \tag{19}$$

By setting the derivative with respect to $\Delta\mathbf{u}_t$ to zero, the *optimal control* can be obtained:

$$\begin{aligned}\Delta\mathbf{u}_t &= -(R+\gamma G_{t-1}^T P G_{t-1})^{-1}[R\mathbf{u}_{t-1}+\gamma G_{t-1}^T P(\mathbf{x}_t+F_{t-1}\Delta\mathbf{x}_t)] \\ &= -(R+\gamma G_{t-1}^T P G_{t-1})^{-1}[R\mathbf{u}_{t-1}+\gamma G_{t-1}^T P\mathbf{x}_t+\gamma G_{t-1}^T P F_{t-1}\Delta\mathbf{x}_t].\end{aligned} \tag{20}$$

From Eq. 20, we can conclude that the policy is in the form of system variables $(\mathbf{u}_{t-1}, \mathbf{x}_t, \Delta\mathbf{x}_t)$ feedback, and the gains are functions of the dynamics of the current linearized system (F_{t-1}, G_{t-1}).

It should be mentioned that although Eq. 20 is still depending on the model of the system, the optimal control increment is defined in a different way as compared to conventional model-based LQR designs due to the discount factor. This means that the controller only needs a rough estimate of the linearized system and input distribution matrices.

Since $\Delta\mathbf{x}(t), \Delta\mathbf{u}(t)$ are measurable as assumed, F_{t-1}, G_{t-1} may be identified by using the simple equation error method:

$$\begin{aligned}\Delta x_{i,t-k+1} &= \mathbf{f}_i\Delta\mathbf{x}_{t-k}+\mathbf{g}_i\Delta\mathbf{u}_{t-k} \\ &= \begin{bmatrix}\Delta\mathbf{x}_{t-k}^T & \Delta\mathbf{u}_{t-k}^T\end{bmatrix}\begin{bmatrix}\mathbf{f}_i^T \\ \mathbf{g}_i^T\end{bmatrix},\end{aligned} \tag{21}$$

where $\Delta x_{i,t-k+1} = x_{i,t-k+1} - x_{i,t-k}$ is the increment of ith state element, $\mathbf{f}_i$ and $\mathbf{g}_i$ are the elements of ith row vector of F_{t-1}, G_{t-1}, and $k = 1, 2...M$ denotes at which time the previously measured data is available. Because there are $n+m$ parameters in the ith row, M needs to satisfy $M \geq (n+m)$. By using piecewise sequential Least Squares (LS) method, the linearized system dynamics (ith row) can be identified from M different data points:

$$\begin{bmatrix}\widehat{\mathbf{f}}_i^T \\ \widehat{\mathbf{g}}_i^T\end{bmatrix}_{LS} = (\mathbf{A}_t^T\mathbf{A}_t)^{-1}\mathbf{A}_t^T\mathbf{b}_t, \tag{22}$$

where

$$\mathbf{A}_t = \begin{bmatrix}\Delta\mathbf{x}_{t-1}^T & \Delta\mathbf{u}_{t-1}^T \\ \vdots & \vdots \\ \Delta\mathbf{x}_{t-M}^T & \Delta\mathbf{u}_{t-M}^T\end{bmatrix},\ \mathbf{b}_t = \begin{bmatrix}\Delta x_{i,t} \\ \vdots \\ \Delta x_{i,t-M+1}\end{bmatrix}. \tag{23}$$

As opposite to the model-based control algorithms with on-line identification of nonlinear systems, the current approach needs only local linear models. Availability of these local linear models is sufficient for iADP algorithms. Furthermore, the determination of the linear model structure is much simpler than the identification of the nonlinear model structure. If the nonlinear model is unknown, while the full state is measurable, iADP algorithm (Value Iteration, VI), as shown below, can be applied to improve the policy online.

iADP algorithm based on full state feedback (iADP-FS)

Evaluation. The cost function kernel matrix P under policy μ can be evaluated and updated recursively to Bellman equation for each iteration $j = 0, 1, ...$ until convergence:

$$\mathbf{x}_t^T P^{(j+1)}\mathbf{x}_t = \mathbf{y}_t^T Q\mathbf{y}_t + \mathbf{u}_t^T R\mathbf{u}_t + \gamma\mathbf{x}_{t+1}^T P^{(j)}\mathbf{x}_{t+1}. \tag{24}$$

Policy improvement. Policy improves for the new kernel matrix $P^{(j+1)}$:

$$\Delta\mathbf{u}_t = -(R+\gamma G_{t-1}^T P^{(j+1)} G_{t-1})^{-1}[R\mathbf{u}_{t-1}+\gamma G_{t-1}^T P^{(j+1)}\mathbf{x}_t+\gamma G_{t-1}^T P^{(j+1)} F_{t-1}\Delta\mathbf{x}_t]. \tag{25}$$

Approximating Δt to 0, the policy designed above approaches the optimal policy.

B. Incremental Approximate Dynamic Programming Based on Output Feedback

The full state of a system, such as air vehicle systems, are often not available. The disturbance of sensors, such as noise, amplification, and interaction, will lead to unreadable output measurement. Here, another approach is presented using only output information instead of the full state of the system.

Considering the nonlinear system (Eq. 6, 8) again, the *output (observation)* around $\mathbf{x}_0$ can also be linearized with Taylor expansion:

$$\mathbf{y}_t = h(\mathbf{x}_t) \simeq h(\mathbf{x}_0) + \frac{\partial h(\mathbf{x})}{\partial \mathbf{x}}|_{\mathbf{x}_0}(\mathbf{x}_t - \mathbf{x}_0). \tag{26}$$

By taking $\mathbf{x}_0 = \mathbf{x}_{t-1}$, the incremental form of the output equation is written as follows:

$$\mathbf{y}_t \simeq \mathbf{y}_{t-1} + H(\mathbf{x}_{t-1})(\mathbf{x}_t - \mathbf{x}_{t-1}), \tag{27}$$

$$\Delta\mathbf{y}_t \simeq H_{t-1}\Delta\mathbf{x}_t, \tag{28}$$

where $H_{t-1} = H(\mathbf{x}_{t-1}) = \frac{\partial h(\mathbf{x})}{\partial \mathbf{x}}|_{\mathbf{x}_{t-1}} \in \mathcal{R}^{p\times n}$ is the *observation matrix* at time step $t-1$.

The nonlinear system incremental dynamics (Eq. 11, 28) at current time t can be represented by the previously measured data on time horizon [t-N, t]:

$$\Delta\mathbf{x}_t \simeq \widetilde{F}_{t-2,t-N-1}\cdot\Delta\mathbf{x}_{t-N} + U_N\cdot\overline{\Delta\mathbf{u}}_{t-1,t-N}, \tag{29}$$

$$\overline{\Delta\mathbf{y}}_{t,t-N+1} \simeq V_N\cdot\Delta\mathbf{x}_{t-N} + T_N\cdot\overline{\Delta\mathbf{u}}_{t-1,t-N}, \tag{30}$$

where symbol $\widetilde{F}_{t-a,t-b} = \prod_{i=t-a}^{t-b} F_i = F_{t-a}\cdot\ \cdots\ \cdot F_{t-b}$,

$$\overline{\Delta\mathbf{u}}_{t-1,t-N} = \begin{bmatrix}\Delta\mathbf{u}_{t-1}\\ \Delta\mathbf{u}_{t-2}\\ \vdots\\ \Delta\mathbf{u}_{t-N}\end{bmatrix} \in \mathcal{R}^{mN}, \qquad \overline{\Delta\mathbf{y}}_{t,t-N+1} = \begin{bmatrix}\Delta\mathbf{y}_{t}\\ \Delta\mathbf{y}_{t-1}\\ \vdots\\ \Delta\mathbf{y}_{t-N+1}\end{bmatrix} \in \mathcal{R}^{mN},$$

$U_N = \begin{bmatrix}G_{t-2} & F_{t-2}G_{t-3} & \ldots & \widetilde{F}_{t-2,t-N}\cdot G_{t-N-1}\end{bmatrix} \in \mathcal{R}^{n\times mN}$ is the *controllability matrix*,

$V_N = \begin{bmatrix}H_{t-1}\\ H_{t-2}\widetilde{F}_{t-3,t-N-1}\\ \vdots\\ H_{t-N}F_{t-N-1}\end{bmatrix} \in \mathcal{R}^{pN\times n}$ is the *observability matrix*,

$$T_N = \begin{bmatrix} H_{t-1}G_{t-2} & H_{t-1}F_{t-2}G_{t-3} & H_{t-1}\widetilde{F}_{t-2,t-3}G_{t-4} & \cdots & H_{t-2}\widetilde{F}_{t-3,t-N}\cdot G_{t-N-1}\\ 0 & H_{t-2}G_{t-3} & H_{t-2}F_{t-3}G_{t-4} & \cdots & H_{t-2}\widetilde{F}_{t-3,t-N}\cdot G_{t-N-1}\\ 0 & 0 & H_{t-3}G_{t-4} & \cdots & H_{t-3}\widetilde{F}_{t-4,t-N}\cdot G_{t-N-1}\\ \vdots & \vdots & \ddots & \ddots & \vdots\\ 0 & 0 & \cdots & 0 & H_{t-N}\cdot G_{t-N-1}\end{bmatrix} \in \mathcal{R}^{pN\times mN}.$$

The left inverse of V_N which has a full column rank can be obtained:

$$V_N^{left} = (V_N^T V_N)^{-1}V_N^T. \tag{31}$$

To have a full column rank for observability matrix V_N, N needs to satisfy $N \geq n/p$. Making the number of parameters to be identified as less as possible, we usually choose the smallest value for N which meets $N \geq n/p$.

By left-multiplying V_N^{left} to Eq. 30, and then substituting the equation of $\Delta\mathbf{x}_{t-N}$ into Eq. 29, the incremental state can be reconstructed uniquely as a function of the input/output data of several previous steps:

$$\begin{aligned}\Delta\mathbf{x}_t &\simeq \widetilde{F}_{t-2,t-N-1}\cdot V_N^{left}\cdot(\overline{\Delta\mathbf{y}}_{t,t-N+1} - T_N\cdot\overline{\Delta\mathbf{u}}_{t-1,t-N}) + U_N\cdot\overline{\Delta\mathbf{u}}_{t-1,t-N}\\ &= \widetilde{F}_{t-2,t-N-1}\cdot V_N^{left}\cdot\overline{\Delta\mathbf{y}}_{t,t-N+1} + (U_N - \widetilde{F}_{t-2,t-N-1}\cdot V_N^{left}\cdot T_N)\cdot\overline{\Delta\mathbf{u}}_{t-1,t-N}\\ &= \begin{bmatrix}M_{\Delta u} & M_{\Delta y}\end{bmatrix}\begin{bmatrix}\overline{\Delta\mathbf{u}}_{t-1,t-N}\\ \overline{\Delta\mathbf{y}}_{t,t-N+1}\end{bmatrix}\\ &= M_{t-1}\overline{\Delta\mathbf{z}}_{t,t-N},\end{aligned} \tag{32}$$

where $M_{\Delta y}$ denotes $M_{\Delta y}(H_{t-2},...,H_{t-N-1},F_{t-2},...,F_{t-N-1}) = \widetilde{F}_{t-2,t-N-1}\cdot V_N^{left} = \widetilde{F}_{t-2,t-N-1}\cdot(V_N^T V_N)^{-1}V_N^T \in \mathcal{R}^{n\times pN}$, $M_{\Delta u}$ denotes $M_{\Delta u}(H_{t-2},...,H_{t-N-1},F_{t-2},...,F_{t-N-1},G_{t-2},...,G_{t-N-1}) = U_N - M_{\Delta y}T_N \in \mathcal{R}^{n\times mN}$, and $M_{t-1} = [M_{\Delta u} \;\; M_{\Delta y}] \in \mathcal{R}^{n\times(m+p)N}$. The matrix M_{t-1} is identifiable by using previous $\widehat{M}$ steps with $\widehat{M} \geq (m+p)N$.

The output increment $\Delta\mathbf{y}_{t+1}$ can also be reconstructed uniquely as a function of the measured input/output data of several previous steps (see Appendix):

$$\begin{aligned}\Delta\mathbf{y}_{t+1} &\simeq \underline{F}_t \cdot \overline{\Delta\mathbf{u}}_{t,t-N+1} + \underline{G}_t \cdot \overline{\Delta\mathbf{y}}_{t,t-N+1} \\ &= \begin{bmatrix}\underline{F}_{t,11} & \underline{F}_{t,12}\end{bmatrix}\begin{bmatrix}\Delta\mathbf{u}_t \\ \overline{\Delta\mathbf{u}}_{t-1,t-N+1}\end{bmatrix} + \underline{G}_t \cdot \overline{\Delta\mathbf{y}}_{t,t-N+1} \\ &= \underline{F}_{t,11}\cdot\Delta\mathbf{u}_t + \underline{F}_{t,12}\cdot\overline{\Delta\mathbf{u}}_{t-1,t-N+1} + \underline{G}_t\cdot\overline{\Delta\mathbf{y}}_{t,t-N+1},\end{aligned} \tag{33}$$

where $\underline{F}_t \in \mathcal{R}^{p\times Nm}$, $\underline{G}_t \in \mathcal{R}^{p\times Np}$, $\underline{F}_{t,11} \in \mathcal{R}^{p\times m}$ and $\underline{F}_{t,12} \in \mathcal{R}^{p\times(N-1)m}$ are partitioned matrices from $\underline{F}_t$. $\underline{F}_t$ and $\underline{G}_t$ are identifiable by using simple equation error method as same as the illustrated method in the previous section (Eq. 21, 22, 23). In this case, there are $(m+p)N$ parameters in each row. Therefore, the number of previous data samples M needs to satisfy $M \geq (m+p)N$.

We assume that the cost-to-go of the system state at time t can be written as a function of a symmetric expended kernel matrix $\overline{P}$ in the quadratic form in terms of a history of observations vector $\overline{\mathbf{z}}_{t,t-N} = [\overline{\mathbf{u}}_{t-1,t-N}^T, \overline{\mathbf{y}}_{t,t-N+1}^T]^T$:

$$V^\mu(\mathbf{z}_{t,t-N}) = \overline{\mathbf{z}}_{t,t-N}^T \overline{P} \overline{\mathbf{z}}_{t,t-N}. \tag{34}$$

Rewrite the optimal policy under the estimation of $\overline{P}$ in terms of $\overline{\mathbf{z}}_{t,t-N}$:

$$\mu^* = \arg\min_{\Delta\mathbf{u}_t} (\mathbf{y}_t^T Q \mathbf{y}_t + \mathbf{u}_t^T R \mathbf{u}_t + \gamma \overline{\mathbf{z}}_{t+1,t-N+1}^T \; \overline{P} \; \overline{\mathbf{z}}_{t+1,t-N+1}), \tag{35}$$

where

$$\overline{\mathbf{z}}_{t+1,t-N+1}^T \; \overline{P} \; \overline{\mathbf{z}}_{t+1,t-N+1} = \begin{bmatrix}\mathbf{u}_{t-1}+\Delta\mathbf{u}_t \\ \overline{\mathbf{u}}_{t-1,t-N+1} \\ \mathbf{y}_t + \Delta\mathbf{y}_{t+1} \\ \overline{\mathbf{y}}_{t,t-N+2}\end{bmatrix}^T \begin{bmatrix}P_{11} & P_{12} & P_{13} & P_{14} \\ P_{12}^T & P_{22} & P_{23} & P_{24} \\ P_{13}^T & P_{23}^T & P_{33} & P_{34} \\ P_{14}^T & P_{24}^T & P_{34}^T & P_{44}\end{bmatrix} \begin{bmatrix}\mathbf{u}_{t-1}+\Delta\mathbf{u}_t \\ \overline{\mathbf{u}}_{t-1,t-N+1} \\ \mathbf{y}_t + \Delta\mathbf{y}_{t+1} \\ \overline{\mathbf{y}}_{t,t-N+2}\end{bmatrix}. \tag{36}$$

By differentiating with respect to $\Delta\mathbf{u}_t$, the policy improvement step can be obtained in terms of the measured data:

$$\begin{aligned}&-[R+\gamma P_{11}+\gamma(\underline{F}_{t,11})^T\cdot P_{33}\cdot\underline{F}_{t,11}+\gamma P_{13}\underline{F}_{t,11}+\gamma(P_{13}\underline{F}_{t,11})^T]\cdot\Delta\mathbf{u}_t \\ &= [R+\gamma P_{11}+\gamma(\underline{F}_{t,11})^T\cdot P_{13}^T]\mathbf{u}_{t-1} + \gamma[(\underline{F}_{t,11})^T P_{33}+P_{13}]\mathbf{y}_t \\ &\quad + \gamma[P_{12}+(\underline{F}_{t,11})^T\cdot P_{23}^T]\overline{\mathbf{u}}_{t-1,t-N+1} + \gamma[P_{14}+(\underline{F}_{t,11})^T\cdot P_{34}]\overline{\mathbf{y}}_{t,t-N+2} \\ &\quad + \gamma[(\underline{F}_{t,11})^T P_{33}+P_{13}](\underline{F}_{t,12}\cdot\overline{\Delta\mathbf{u}}_{t-1,t-N+1} + \underline{G}_t\cdot\overline{\Delta\mathbf{y}}_{t,t-N+1}).\end{aligned} \tag{37}$$

If the nonlinear model is unknown, and only partial information about the states is accessible, output feedback ADP algorithm combined with incremental method can be applied to improve the policy online.

iADP algorithm based on output feedback (iADP-OP)

Evaluation. The cost function kernel matrix $\overline{P}$ under policy μ can be evaluated and updated recursively according to Bellman equation for each iteration $j = 0, 1, ...$ until convergence:

$$\mathbf{z}_{t,t-N+1}'^T \overline{P}^{(j+1)} \mathbf{z}'_{t,t-N+1} = \mathbf{y}_t^T Q \mathbf{y}_t + \mathbf{u}_t^T R \mathbf{u}_t + \gamma \mathbf{z}_{t+1,t-N+2}'^T \overline{P}^{(j)} \mathbf{z}'_{t+1,t-N+2}. \tag{38}$$

Policy improvement. Policy improves for the new kernel matrix $\overline{P}^{(j+1)}$ according to the derived optimal control policy:

$$\begin{aligned}\Delta\mathbf{u}_t = &-[R+\gamma P_{11}+\gamma(\underline{F}_{t,11})^T\cdot P_{33}\cdot\underline{F}_{t,11}+\gamma P_{13}\underline{F}_{t,11}+\gamma(P_{13}\underline{F}_{t,11})^T]^{-1}\cdot \\ &\{[R+\gamma P_{11}+\gamma(\underline{F}_{t,11})^T\cdot P_{13}^T]\mathbf{u}_{t-1} + \gamma[(\underline{F}_{t,11})^T P_{33}+P_{13}]\mathbf{y}_t \\ &+ \gamma[P_{12}+(\underline{F}_{t,11})^T\cdot P_{23}^T]\overline{\mathbf{u}}_{t-1,t-N+1} + \gamma[P_{14}+(\underline{F}_{t,11})^T\cdot P_{34}]\overline{\mathbf{y}}_{t,t-N+2} \\ &+ \gamma[(\underline{F}_{t,11})^T P_{33}+P_{13}](\underline{F}_{t,12}\cdot\overline{\Delta\mathbf{u}}_{t-1,t-N+1} + \underline{G}_t\cdot\overline{\Delta\mathbf{y}}_{t,t-N+1})\}.\end{aligned} \tag{39}$$

Approximating Δt to 0, the policy designed above approaches the optimal policy.

IV. Experiments and Results

This section shows applications of both iADP based on full state feedback and iADP based on output feedback algorithms on a simulation model for validation.

A. Air vehicle model

A nonlinear air vehicle simulation model will be used in this section. Air vehicle models are highly nonlinear and can be generally given as follows:

$$\dot{\mathbf{x}}(t) = f[\mathbf{x}(t), \mathbf{u}(t) + \mathbf{w}(t)], \tag{40}$$

$$\mathbf{y}(t) = h[\mathbf{x}(t)], \tag{41}$$

where Eq. 40 is the kinematic state equation which provides the physical evaluation of the state vector over time, the term of $\mathbf{w}(t)$ is the external disturbance, which is set to be caused only by the input noise, Eq. 41 is the output (observation) equation which can be measured using sensors.

As an application for these control algorithms, only elevator deflection will be regulated as pitch control to stabilize the air vehicles. Thus, we are interested in two longitudinal states, *angle of attack* α and *pitch rate* q (i.e. the system variables are $\mathbf{x} = [\alpha \;\; q]$), and one control input, *elevator deflection angle* δ_e.

The nonlinear model in the pitch plane is simulated around a steady wings-level flight condition:

$$\dot{\alpha} = q + \frac{\bar{q}S}{m_a V_T}\, C_z(\alpha, q, M_a, \delta_e), \tag{42}$$

$$\dot{q} = \frac{\bar{q}Sd}{I_{yy}}\, C_m(\alpha, q, M_a, \delta_e), \tag{43}$$

where $\bar{q}$ is dynamic pressure, S is reference area, m_a is mass, V_T is speed, d is reference length, I_{yy} is pitching moment of inertia, C_z is the aerodynamic force coefficient, and C_m is the aerodynamic moment coefficient. C_z and C_m are highly nonlinear functions of angle of attack α, pitch rate q, Mach number M_a and elevator deflection δ_e.

As a preliminary test, an air vehicle model (parameter data) is taken in the pitch plane for $-10^o < \alpha < 10^o$:[40,41]

$$\begin{aligned}
&C_z(\alpha, q, M_a, \delta_e) = C_{z1}(\alpha, M_a) + B_z\delta_e,\\
&C_m(\alpha, q, M_a, \delta_e) = C_{m1}(\alpha, M_a) + B_m\delta_e,\\
&B_z = b_1 M_a + b_2,\\
&B_m = b_3 M_a + b_4,\\
&C_{z1}(\alpha, M_a) = \phi_{z1}(\alpha) + \phi_{z2} M_a,\\
&C_{z2}(\alpha, M_a) = \phi_{m1}(\alpha) + \phi_{m2} M_a,\\
&\phi_{z1}(\alpha) = h_1\alpha^3 + h_2\alpha|\alpha| + h_3\alpha,\\
&\phi_{m1}(\alpha) = h_4\alpha^3 + h_5\alpha|\alpha| + h_6\alpha,\\
&\phi_{z2} = h_7\alpha|\alpha| + h_8\alpha,\\
&\phi_{m2} = h_9\alpha|\alpha| + h_{10}\alpha,
\end{aligned} \tag{44}$$

where $b_1, ..., b_4,\ h_1, ..., h_{10}$ are identified constant coefficients in the flight envelop, and the Mach number is set to $M_a = 2.2$. The sample frequency of simulations is selected to be 100 Hz.

When the input is $\mathbf{u}(t) = 0$, $\alpha = 0$ and $q = 0$ are an equilibrium of the system. The flight control task is to stabilize the system (i.e., a regulator problem, see Eq. 13), if there is any input disturbance or any offset from this condition. Specifically, an optimal policy μ^* (optimal control $\mathbf{u}^*$) and the associated optimum performance V^* need to be found out by minimizing the state-cost function J.[42]

B. Results

1. *IADP algorithm based on full state feedback*

Since the nonlinear model is unknown, and the full state is measurable (i.e. α and q are measurable), the iADP algorithm based on full state feedback was applied. As with other ADP methods, good state-cost estimation depends heavily on the exploration of the state space, which is persistent excitation in this case. Different modes of the aircraft can be excited by using input techniques to determine identification parameters. There are many different input techniques, such as pseudo-random noise, classical sine waves, doublets, 3211 doublets, among which 3211 are one of the most commonly used maneuvers in aircraft system identification. On the other hand, disturbances are usually undesirable inputs in the real world. Fig. 1 shows the disturbance response when a 3211 input disturbance was introduced. The control system trained with iADP algorithm has a lower disturbance response and an improved performance compared to the initial one.

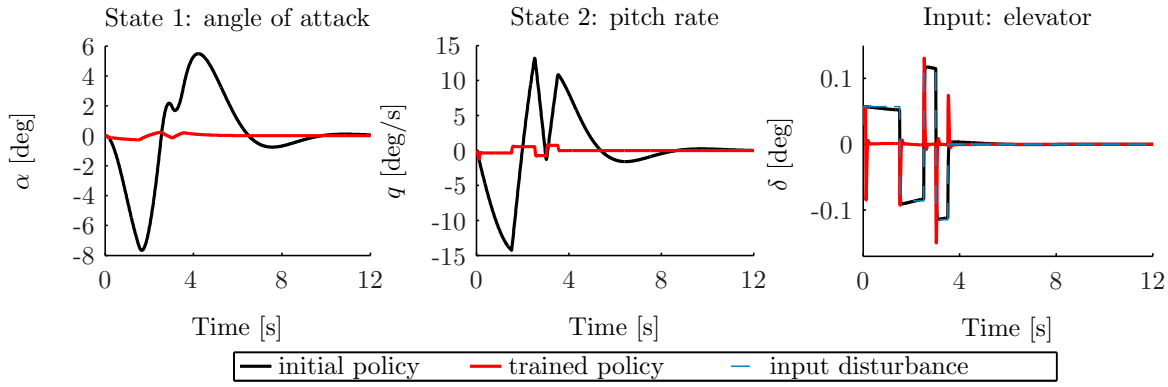

Figure 1. IADP-FS applied to nonlinear aircraft model with 3211 input disturbance

Fig. 2 shows the control performance when the initial state is an offset from stable condition after a simulated gust. Without persistent excitation, the nonlinear model cannot be identified. After training, the information of control effectiveness matrix $G(\mathbf{x}, \mathbf{u})$ and system matrix $F(\mathbf{x}, \mathbf{u})$ can be used to estimate the current linearized system when the system cannot be identified using online identification. Because this iADP method uses a very simple quadratic cost function, the policy parameters of kernel matrix P converge very quickly, after only 2 iterations (see Fig. 3).

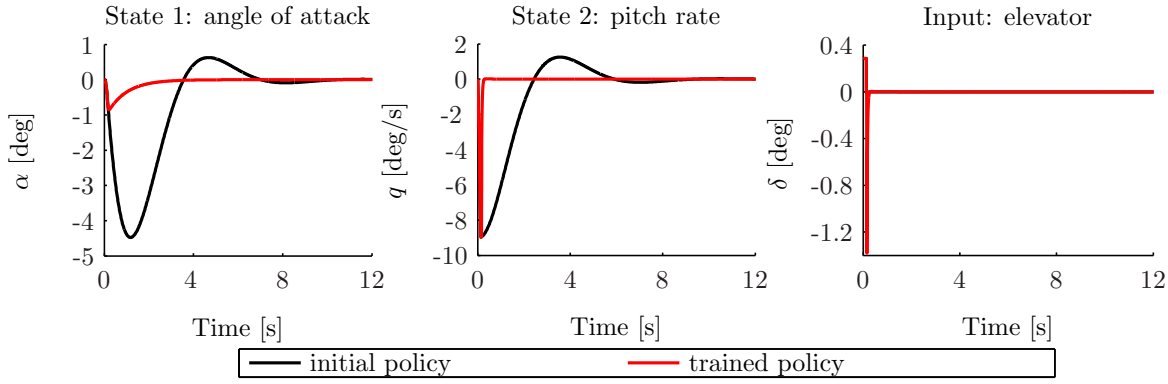

Figure 2. IADP-FS applied to nonlinear aircraft model with an initial offset

This control method does not need the model of the nonlinear system, but still need the full state to estimate the cost function and the control effectiveness matrix. If the model of the nonlinear system is unknown, and only a coupled state information (observation) can be obtained, the iADP algorithm based on output feedback can be used.

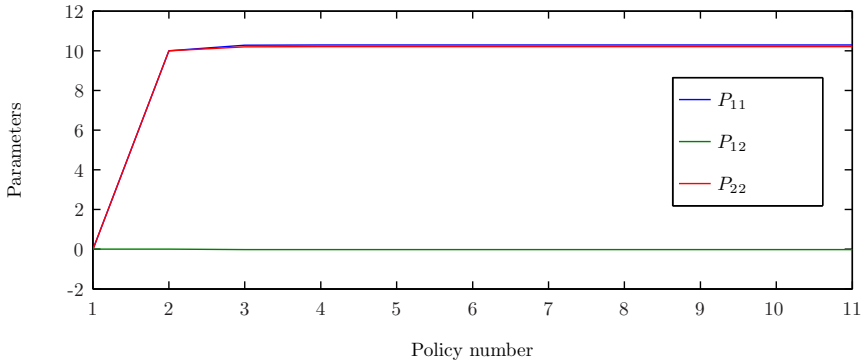

Figure 3. Kernel matrix parameters during training with IADP-FS

2. IADP algorithm based on output feedback

In practice, vane measurement techniques are found to be a cost effective way of measuring the angle of attack α.[43] The measured vane angle indicates the local airflow direction. It may considerably deviate from the direction of the freestream flow. This is partially due to flow perturbations induced by the aircraft $\Delta\alpha_{a/c\ induced}$. Thus, vanes are usually mounted on the aircraft in a location x_{vane} that allows for relatively undisturbed air flow to be measured, such as a nose boom extending forward, instead of locating in the aircraft center of gravity x_{cg}. As a consequence, another source of errors (kinematic position error) induced by angular velocities q at the vane location has to be considered:

$$\alpha_{measure} \simeq C_c(\alpha + \frac{x_{vane} - x_{cg}}{V} \cdot q), \tag{45}$$

where C_c denotes the calibration coefficient.

According to this practical case, the output/sensor measurement is set to be a combination of α and q with coefficients. Considering the practical case as well as the fact that α is what to regulate to keep stable, we choose a big portion of α (0.9) and a small portion of q (0.1). Thus, the air vehicle simulation model in Eq. 41 is rewritten as follow:

$$\mathbf{y}(t) = [c_1\ c_2] \cdot \mathbf{x}(t) = [0.9\ 0.1] \cdot \begin{bmatrix} \alpha \\ q \end{bmatrix}. \tag{46}$$

This algorithm is on the assumption that the system is controllable and observable (see Eq. 29 - 31). When the observability of the system can be provided (Eq. 44, 46), even with a small portion of q (0.1), iADP-OP algorithm works. Fig. 4 shows the disturbance response when a 3211 input disturbance was introduced; Fig. 5 shows the control performance when the initial state is an offset from stable condition after a simulated gust; and Fig. 6 shows that the policy parameters of kernel matrix converge quickly, and after 4 training iterations, the kernel matrix remains almost the same. This means after only 4 training iterations the nonlinear system can be regulated as good as shown in Fig. 4 and Fig. 5. The control system trained with iADP-OP algorithm also has a higher dynamic stiffness and a lower disturbance response compared to the initial one.

Note that when we have the information of α, we might calculate q by using the identified model and long enough previously measured observation. With some assumption, the aircraft pitch plane system, α and q, is observable with only information of α. But the iADP algorithm we use is a model-free method, that is we do not make any assumption about the model, and we use observations from only two samples. Therefore, we define observability in terms of whether V_N in Eq. 30 has a full column rank. If no information about one of the states can be provided, the iADP algorithm may not be useful.

Fig. 7 and Fig. 8 show a comparison of disturbance response and natural response, respectively, among 3 policies. The initial policy is what the original system follows. It cannot compensate the undesired inputs, such as gusts and ground effects. When full states are available, iADP-FS algorithm improves the closed-loop performance, lowers the disturbance response, and stabilizes the system from an offset much quicker. When full states are not available, but the system is observable, iADP-OP algorithm generates a policy. This policy has an almost equal ability to stabilize and regulate the system to that of iADP-FS policy.

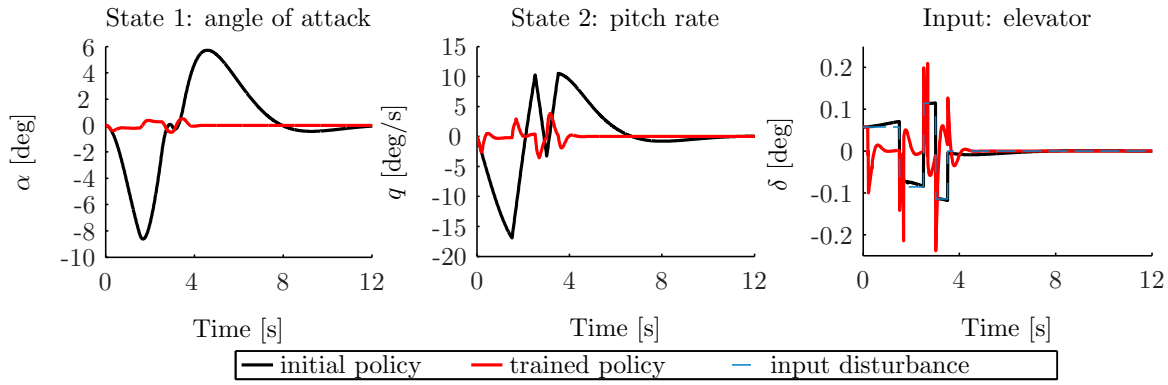

Figure 4. IADP-OP applied to nonlinear aircraft model with 3211 input disturbance ($c_1 = 0.9, c_2 = 0.1$)

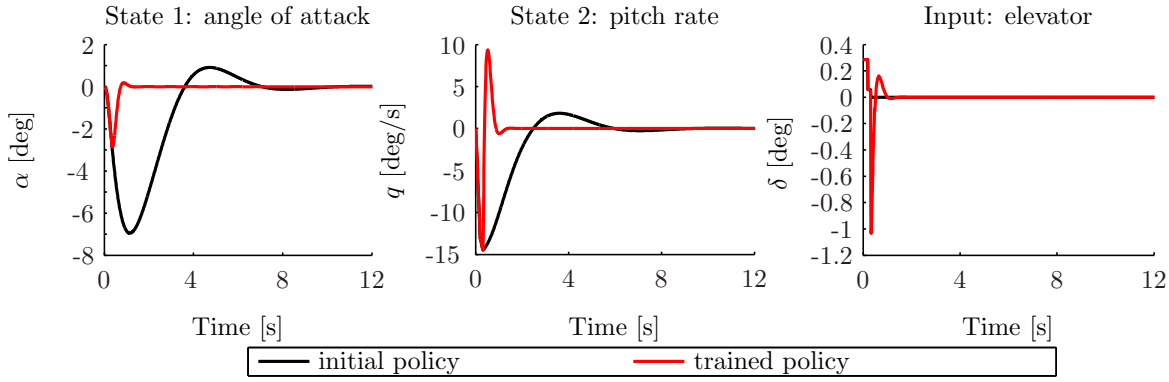

Figure 5. IADP-OP applied to nonlinear aircraft model with an initial offset ($c_1 = 0.9, c_2 = 0.1$)

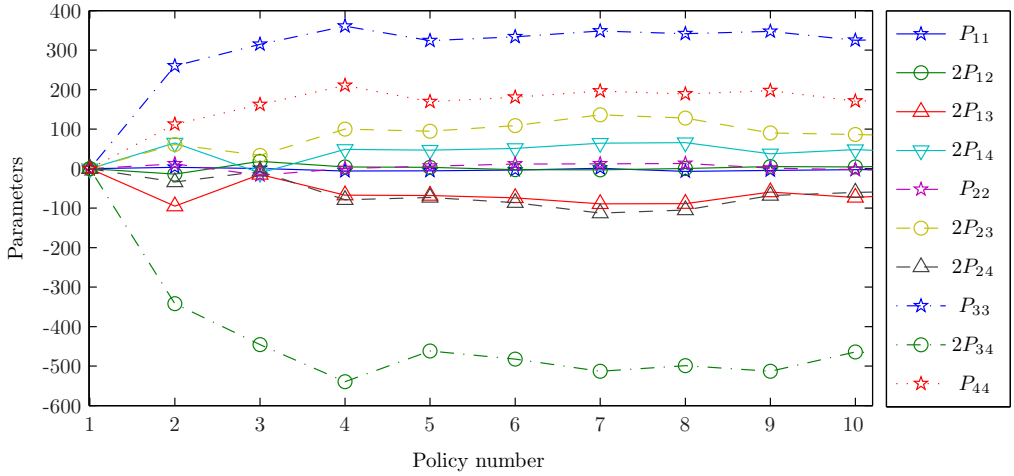

Figure 6. Kernel matrix parameters during training with IADP-OP ($c_1 = 0.9, c_2 = 0.1$)

V. Conclusion

This paper proposes a novel adaptive control method for nonlinear systems, called incremental Approximate Dynamic Programming (iADP). Approximate dynamic programming algorithms provide a linear optimal control approach to solve Linear Quadratic Regulator (LQR) problems and generate an optimal policy without knowing the system dynamics. In addition, the incremental approaches can deal with the

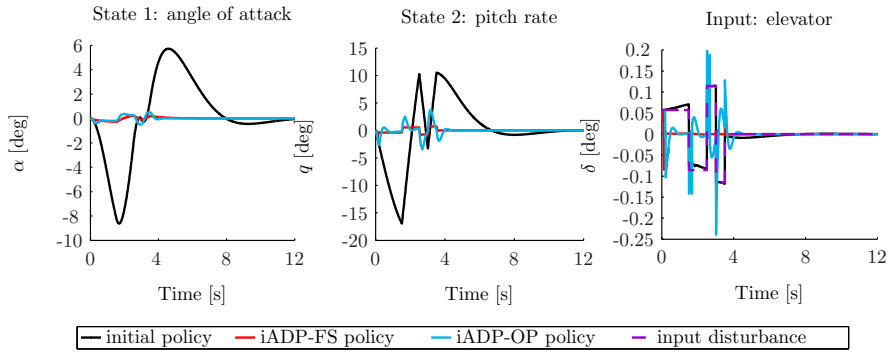

Figure 7. Comparison of policies applied to nonlinear aircraft model with 3211 input disturbance

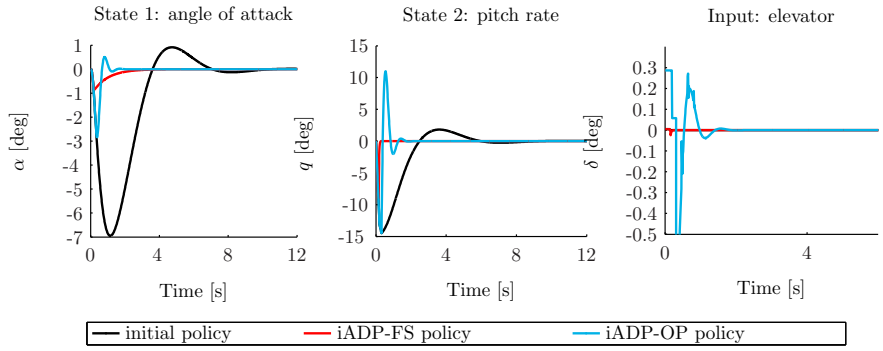

Figure 8. Comparison of policies applied to nonlinear aircraft model with an initial offset

nonlinearity of systems. The iADP method combines the advantages of both the ADP method and the incremental approach, and provides a model-free, effective adaptive flight controller for nonlinear systems. In addition to the iADP algorithm based on full state feedback (iADP-FS), an iADP algorithm based on output feedback (iADP-OP) is proposed. IADP-OP uses only a history of measured input and output data from a dynamical nonlinear system to reconstruct the local model.

Both the iADP-FS algorithm and the iADP-OP algorithm are applied to an aerospace related model. The simulation results demonstrate that the policy trained with iADP-FS has an improved performance and a lower disturbance response compared to the initial one. When the system is completely unknown but observable, the control system trained with iADP-OP is also shown to be superior to the initial policy.

The iADP method deals with nonlinearity by using an incremental approach, as opposed to increasing complexity of the cost function or on-line identification of the global nonlinear model, which maintains the efficiency of resource usage. Thus, this method can be applied to complex systems without sufficient computing power or storage capacity, such as Micro Air Vehicles. Because the iADP method still uses a very simple quadratic cost function, the policy parameters of the kernel matrix converge very quickly. This also makes the iADP method a candidate for on-line adaptive control.

This new method can potentially design a near-optimal controller for nonlinear systems without a priori knowledge nor full state measurements of the dynamic model, while keeping the design process simple and systematic as compared to conventional ADP algorithms. Although still no theoretical guarantees on the nonlinear system performance can be offered, the performance of systems with approximately convex cost functions is observed to be very promising. For general nonlinear systems and more complex tasks, real applications and other possibilities such as piecewise quadratic cost functions will be studied in the future.

Appendix

The nonlinear system incremental output equation (Eq. 28) can be represented by the a history of measured input/output data on time horizon [t-N, t-1] in another form:

$$\overline{\Delta \mathbf{y}}_{t-1,t-N} \simeq \overline{V}_N \cdot \Delta \mathbf{x}_{t-N} + \overline{T}_N \cdot \overline{\Delta \mathbf{u}}_{t-1,t-N}, \tag{47}$$

where $\overline{V}_N = \begin{bmatrix} H_{t-2}\widetilde{F}_{t-3,t-N-1} \\ H_{t-3}\widetilde{F}_{t-3,t-N-1} \\ \vdots \\ H_{t-N-1} \end{bmatrix} \in \mathcal{R}^{pN \times n}$,

$$\overline{T}_N = \begin{bmatrix} 0 & H_{t-2}G_{t-3} & H_{t-2}F_{t-3}G_{t-4} & \cdots & H_{t-2}\widetilde{F}_{t-3,t-N} \cdot G_{t-N-1} \\ 0 & 0 & H_{t-3}G_{t-4} & \cdots & H_{t-3}\widetilde{F}_{t-4,t-N} \cdot G_{t-N-1} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & H_{t-N} \cdot G_{t-N-1} \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \in \mathcal{R}^{pN \times mN}.$$

Eq. 28 and Eq. 11 is rewritten as below:

$$\Delta \mathbf{y}_t \simeq H_{t-1}\Delta \mathbf{x}_t, \tag{48}$$

$$\Delta \mathbf{x}_t \simeq \widetilde{F}_{t-2,t-N-1} \cdot \Delta \mathbf{x}_{t-N} + U_N \cdot \overline{\Delta \mathbf{u}}_{t-1,t-N}. \tag{49}$$

The left inverse of $\overline{V}_N$ which also has a full column rank can be obtained:

$$\underline{V}_N^{left} = (\overline{V}_N^T \overline{V}_N)^{-1} \overline{V}_N^T. \tag{50}$$

By left-multiplying H_{t-1} to Eq. 49 and adding to Eq. 48, term $\Delta \mathbf{x}_t$ can be eliminated. Left multiplying $\overline{V}_N^{left}$ to Eq. 47 and substituting equation of $\Delta \mathbf{x}_{t-N}$ into the new equation from previous step, the dynamics of output and previous measured data can be obtained:

$$\begin{aligned} \Delta \mathbf{y}_t &\simeq (H_{t-1}U_N - H_{t-1}\widetilde{F}_{t-2,t-N-1} \cdot \overline{V}_N^{left}\overline{T}_N) \cdot \overline{\Delta \mathbf{u}}_{t-1,t-N} \\ &+ H_{t-1}\widetilde{F}_{t-2,t-N-1}\overline{V}_N^{left} \cdot \overline{\Delta \mathbf{y}}_{t-1,t-N} \\ &= \underline{F}_{t-1} \cdot \overline{\Delta \mathbf{u}}_{t-1,t-N} + \underline{G}_{t-1} \cdot \overline{\Delta \mathbf{y}}_{t-1,t-N}. \end{aligned} \tag{51}$$

The output increment can also be reconstructed uniquely as a function of the measured input/output data of several previous steps.

Acknowledgement. The first author is financially supported for this Ph.D. research by China Scholarship Council with the project reference number of 201306290026.

References

[1]Lombaerts, T., Oort, E. V., Chu, Q., Mulder, J., and Joosten, D., "Online aerodynamic model structure selection and parameter estimation for fault tolerant control," *Journal of guidance, control, and dynamics*, Vol. 33, No. 3, 2010, pp. 707–723.

[2]De Weerdt, E., Chu, Q., and Mulder, J., "Neural network output optimization using interval analysis," *Neural Networks, IEEE Transactions on*, Vol. 20, No. 4, 2009, pp. 638–653.

[3]Tang, L., Roemer, M., Ge, J., Crassidis, A., Prasad, J., and Belcastro, C., "Methodologies for adaptive flight envelope estimation and protection," *AIAA Guidance, Navigation, and Control Conference*, 2009, p. 6260.

[4]Van Oort, E., Sonneveldt, L., Chu, Q.-P., and Mulder, J., "Full-envelope modular adaptive control of a fighter aircraft using orthogonal least squares," *Journal of guidance, control, and dynamics*, Vol. 33, No. 5, 2010, pp. 1461–1472.

[5]Sghairi, M., De Bonneval, A., Crouzet, Y., Aubert, J., and Brot, P., "Challenges in Building Fault-Tolerant Flight Control System for a Civil Aircraft," *IAENG International Journal of Computer Science*, Vol. 35, No. 4, 2008.

[6]Sonneveldt, L., Van Oort, E., Chu, Q., and Mulder, J., "Nonlinear adaptive trajectory control applied to an F-16 model," *Journal of Guidance, control, and Dynamics*, Vol. 32, No. 1, 2009, pp. 25–39.

[7]Farrell, J., Sharma, M., and Polycarpou, M., "Backstepping-based flight control with adaptive function approximation," *Journal of Guidance, Control, and Dynamics*, Vol. 28, No. 6, 2005, pp. 1089–1102.

[8]Sonneveldt, L., Van Oort, E., Chu, Q., and Mulder, J., "Comparison of inverse optimal and tuning functions designs for adaptive missile control," *Journal of guidance, control, and dynamics*, Vol. 31, No. 4, 2008, pp. 1176–1182.

[9]Sonneveldt, L., Chu, Q., and Mulder, J., "Nonlinear flight control design using constrained adaptive backstepping," *Journal of Guidance, Control, and Dynamics*, Vol. 30, No. 2, 2007, pp. 322–336.

[10]Krüger, T., Schnetter, P., Placzek, R., and Vörsmann, P., "Fault-tolerant nonlinear adaptive flight control using sliding mode online learning," *Neural Networks*, Vol. 32, 2012, pp. 267–274.

[11]Sutton, R. S. and Barto, A. G., *Introduction to reinforcement learning*, MIT Press, 1998.

[12]Bellman, R., *Dynamic Programming*, Princeton University Press, 1957.

[13]Khan, S. G., Herrmann, G., Lewis, F. L., Pipe, T., and Melhuish, C., "Reinforcement learning and optimal adaptive control: An overview and implementation examples," *Annual Reviews in Control*, Vol. 36, No. 1, 2012, pp. 42–59.

[14]Si, J., *Handbook of learning and approximate dynamic programming*, Vol. 2, John Wiley & Sons, 2004.

[15]Schaul, T., Horgan, D., Gregor, K., and Silver, D., "Universal Value Function Approximators," *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, 2015, pp. 1312–1320.

[16]Keshavarz, A. and Boyd, S., "Quadratic approximate dynamic programming for input-affine systems," *International Journal of Robust and Nonlinear Control*, Vol. 24, No. 3, 2014, pp. 432–449.

[17]Lewis, F. L. and Vamvoudakis, K. G., "Reinforcement learning for partially observable dynamic processes: Adaptive dynamic programming using measured output data," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, Vol. 41, No. 1, 2011, pp. 14–25.

[18]Sieberling, S., Chu, Q. P., and Mulder, J. A., "Robust flight control using incremental nonlinear dynamic inversion and angular acceleration prediction," *Journal of guidance, control, and dynamics*, Vol. 33, No. 6, 2010, pp. 1732–1742.

[19]Simplício, P., Pavel, M. D., van Kampen, E., and Chu, Q. P., "An acceleration measurements-based approach for helicopter nonlinear flight control using Incremental Nonlinear Dynamic Inversion," *Control Engineering Practice*, Vol. 21, No. 8, 2013, pp. 1065–1077.

[20]ACQUATELLA, P. J., van Kampen, E., and Chu, Q. P., "Incremental Backstepping for Robust Nonlienar Flight Control," *Proceedings of the EuroGNC 2013*, 2013.

[21]Sigaud, O. and Buffet, O., *Markov decision processes in artificial intelligence*, John Wiley & Sons, 2013.

[22]Bakolas, E. and Tsiotras, P., "Feedback navigation in an uncertain flowfield and connections with pursuit strategies," *Journal of Guidance, Control, and Dynamics*, Vol. 35, No. 4, 2012, pp. 1268–1279.

[23]Anderson, R. P., Bakolas, E., Milutinović, D., and Tsiotras, P., "Optimal feedback guidance of a small aerial vehicle in a stochastic wind," *Journal of Guidance, Control, and Dynamics*, Vol. 36, No. 4, 2013, pp. 975–985.

[24]Zou, A.-M. and Kumar, K. D., "Quaternion-based distributed output feedback attitude coordination control for spacecraft formation flying," *Journal of Guidance, Control, and Dynamics*, Vol. 36, No. 2, 2013, pp. 548–556.

[25]Hu, Q., Jiang, B., and Friswell, M. I., "Robust saturated finite time output feedback attitude stabilization for rigid spacecraft," *Journal of Guidance, Control, and Dynamics*, Vol. 37, No. 6, 2014, pp. 1914–1929.

[26]Ulrich, S., Sasiadek, J. Z., and Barkana, I., "Nonlinear Adaptive Output Feedback Control of Flexible-Joint Space Manipulators with Joint Stiffness Uncertainties," *Journal of Guidance, Control, and Dynamics*, Vol. 37, No. 6, 2014, pp. 1961–1975.

[27]Mazenc, F. and Bernard, O., "Interval observers for linear time-invariant systems with disturbances," *Automatica*, Vol. 47, No. 1, 2011, pp. 140–147.

[28]Efimov, D., Raïssi, T., Chebotarev, S., and Zolghadri, A., "Interval state observer for nonlinear time varying systems," *Automatica*, Vol. 49, No. 1, 2013, pp. 200–205.

[29]Akella, M. R., Thakur, D., and Mazenc, F., "Partial Lyapunov Strictification: Smooth Angular Velocity Observers for Attitude Tracking Control," *Journal of Guidance, Control, and Dynamics*, Vol. 38, No. 3, 2015, pp. 442–451.

[30]Zhou, Y., van Kampen, E., and Chu, Q. P., "Incremental Approximate Dynamic Programming for Nonlinear Flight Control Design," *Proceedings of the EuroGNC 2015*, 2015.

[31]Wiering, M. and van Otterlo, M., *Reinforcement Learning: State-of-the-art*, Vol. 12, Springer Science & Business Media, 2012.

[32]Bertsekas, D. P. and Tsitsiklis, J. N., "Neuro-dynamic programming: an overview," *Decision and Control, 1995., Proceedings of the 34th IEEE Conference on*, Vol. 1, IEEE, 1995, pp. 560–564.

[33]Busoniu, L., Babuska, R., De Schutter, B., and Ernst, D., *Reinforcement learning and dynamic programming using function approximators*, CRC Press, 2010.

[34]Busoniu, L., Ernst, D., De Schutter, B., and Babuska, R., "Online least-squares policy iteration for reinforcement learning control," *American Control Conference (ACC), 2010*, IEEE, 2010, pp. 486–491.

[35]Doya, K., "Reinforcement learning in continuous time and space," *Neural computation*, Vol. 12, No. 1, 2000, pp. 219–245.

[36]van Kampen, E., Chu, Q. P., and Mulder, J. A., "Continuous adaptive critic flight control aided with approximated plant dynamics," *Proc AIAA Guidance Navig Control Conf*, Vol. 5, 2006, pp. 2989–3016.

[37]Brooks, A., Makarenko, A., Williams, S., and Durrant-Whyte, H., "Parametric POMDPs for planning in continuous state spaces," *Robotics and Autonomous Systems*, Vol. 54, No. 11, 2006, pp. 887–897.

[38]Scott A. Miller, Z. A. H. and Chong, E. K. P., "A POMDP framework for coordinated guidance of autonomous UAVs for multitarget tracking," *EURASIP Journal on Advances in Signal Processing*, 2009.

[39]Ragi, S. and Chong, E. K. P., "UAV path planning in a dynamic environment via partially observable Markov decision process," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 49, No. 4, 2013, pp. 2397–2412.

[40]Sonneveldt, L., *Adaptive backstepping flight control for modern fighter aircraft*, TU Delft, Delft University of Technology, 2010.

[41]Kim, S.-H., Kim, Y.-S., and Song, C., "A robust adaptive nonlinear control approach to missile autopilot design," *Control engineering practice*, Vol. 12, No. 2, 2004, pp. 149–154.

[42]Anderson, B. D. and Moore, J. B., *Optimal control: linear quadratic methods*, Courier Corporation, 2007.

[43]Laban, M., *On-line aircraft aerodynamic model identification*, TU Delft, Delft University of Technology, 1994.