Incremental Approximate Dynamic Programming for Nonlinear Flight Control Design

Y. Zhou, E. van Kampen and Q.P. Chu

Abstract A self-learning adaptive flight control design for non-linear systems allows reliable and effective operation of flight vehicles in a dynamic environment. Approximate dynamic programming (ADP) provides a model-free and computationally effective process for designing adaptive linear optimal controllers. This paper presents an incremental ADP (iADP) method which combines ADP method and incremental control techniques to design an adaptive near-optimal nonlinear controller. This nonlinear control method does not need any information of the dynamic model, but requires only the considered state (full state) and measured input and output. The iADP method was implemented on an F-16 aircraft simulation model. The results prove the success of the proposed method and show a potential approach of iADP nonlinear flight controllers without knowing full state.

1 Introduction

Reinforcement learning is learning what actions to take to affect the state and to maximize the numerical reward signal by interacting with the environment to some extent, and to achieve a goal ultimately. It is not defined by characterizing learning methods, but by learning problems which can be described as an optimal control problem of Markov Decision Processes (MDPs)[1,2]. This method links bio-inspired

Ye Zhou
Delft University of Technology, Kluyverweg 1, 2629HS Delft, the Netherlands, e-mail: Y.Zhou-6@tudelft.nl

Erik-Jan van Kampen
Delft University of Technology, Kluyverweg 1, 2629HS Delft, the Netherlands, e-mail: E.vanKampen@tudelft.nl

Qi Ping Chu
Delft University of Technology, Kluyverweg 1, 2629HS Delft, the Netherlands, e-mail: Q.P.Chu@tudelft.nl

artificial intelligence techniques to the field of control, and overcomes some of the limitations and problems, like the problem in most of the control methods of demanding precise models. The merits of model-free processes, adaptability to the environment, etc. make reinforcement learning controllers suitable and effective in the field of adaptive flight control.

In the real world, the agent might not have perfect perception of states of the environment. In most of the cases, the agent "observes" the state of the environment but these observations may be noisy and cannot provide sufficient information. UAVs (Unmanned Air Vehicles), as an example, may lose the GPS (Global Position System) signal during indoor flights or when obstructed by buildings, trees or other objects; and may obtain inaccurate altitude by using only GPS or pressure sensor in outdoor flight tasks. Manned air vehicles, such as commercial aircraft and military aircraft, may also get degraded sensor measurements or even lose the measurements for several seconds or minutes. Besides, the effect of wind disturbance on UAVs and the collisions between UAVs and obstacles and among UAVs also impede obtaining a perfect measurement of full state.

The framework dealing with Partially Observable Markov Decision Process (POMDP) problems and deciding how to act in partially observable environments has been developed especially for these situations and remains an active area of research. Nominal Belief-state Optimization (NBO)[3,4] which combines application-specific approximations and techniques within the POMDP framework produced a practical design that coordinates the UAVs in the presence of occlusions. An online, forward-search algorithm, called the Posterior Belief Distribution (PBD)[5], was proposed for calculating the posterior distribution over beliefs after a sequence of actions. This method allows evaluating the expected reward of a sequence of primitive actions, called 'macro-actions' and controlling an UAV in a target monitoring task.

Output-Feedback (OPFB) approximate dynamic programming algorithms[6] were proposed, as opposed to full state feedback, to tackle problems without direct state observation. 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. However, the control derivatives in some cases, like F-16 model, are naturally non-linear where this algorithm can not work well. On the other hand, incremental methods can be used in non-linear systems. Because it only computes the changes of the control surface deflection rather than the complete deflection, such as incremental Nonlinear Dynamic Inversion (INDI)[7,8]. However, in the current incremental approach, information about system model, aerodynamic force and states are still needed. To combine the advantages of OPFB algorithms and incremental methods, two steps should be done. First, an algorithm combining ADP and incremental approach with direct full state observation should be developed. Second, the OPFB method can be used based on the incremental ADP algorithm to reconstruct the full state from only output and input measurement.

In this paper, we are focusing on the first step of the proposed method. A new approach combining approximate dynamic programming algorithm and incremental

methods is developed for nonlinear systems. And this approach will be applied on a non-linear F-16 fighter airplane simulation model.

2 Background

2.1 Reinforcement learning methods

Reinforcement learning is learning from experience denoting by a reward or punishment, which is inspired by animal behaviors. In control engineering, control costs / cost functions are used, thus, a reward item diminishes the cost while a punishment increases the cost.

The methods for solving RL problems can be classified into three categories: Dynamic Programming (DP), Monte Carlo methods (MC) and Temporal-Difference learning (TD)[1]. 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 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[9]. Policy iteration (PI) algorithms evaluate the current policy to obtain the value function, and improve the policy accordingly. Value iteration (VI) algorithms find the optimal value function and the optimal policy. Policy search (PS) algorithms search for the optimal policy directly by using optimization algorithms.

2.1.1 Temporal difference methods

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[10]. TD method can be defined basically in an iterative estimation update form as follows:

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

where, $\widehat{V}(\mathbf{x}_t)$ is the estimation of *value 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 *forgetting factor*. The target for update is $r_{t+1} + \gamma \widehat{V}(x_{t+1})$, and the *TD error* is $r_{t+1} + \gamma \widehat{V}(x_{t+1}) - \widehat{V}(x_t)$. TD methods can be classified into three most popular categories[1]:

- 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[11]. They use TD error to evaluate a selected action to strengthen or weaken the tendency of selecting the action for the future. And they have a separate memory structure to explicitly represent the policy independent of the value function.

2.1.2 Approximate Dynamic Programming and Partial Observability

Traditional DP method illustrated in[1] 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 $\mathscr{R}^n$. To tackle with the "curse of dimensionality" problem, numerical methods, like approximate dynamic programming (ADP), are the best to solve the optimality problems forward online[12,13].

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 $\widehat{V}(\mathbf{x}_t, \mathit{parameters})$ instead of $\widehat{V}(\mathbf{x}_t)$ to approximate the cost function / value function V. 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 V_n iterative according to $V_{n+1} = ALV_n$, 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 value function V_n for a policy π_n, and policy improvement step, which generates a new policy with respect to the value function approximation V_n greedily[14].

Classical DP methods assume that the system is a fully observable system. Thus, the optimal action can be chosen in terms of the full knowledge of system states. It also assumes that the observed states 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, $\mathbf{u}_t$. However, the agents often try to control the systems without

enough information to infer its real states[14]. ADP is an extension of more general cases like POMDPs which has been active research in control and operation field[13].

2.2 Aircraft Model

A non-linear F-16 simulation model[15] will be used in this paper . This model simulates the dynamics of the real F-16 aircraft based on the description by Stevens and Lewis[16] and a NASA report[17]. The non-linear F-16 model which is constructed using Simulink allows for control over thrust, elevator, aileron and rudder. The initial states supplied from a trimming routine for a given altitude and velocity constitute the initial conditions of the F-16.

Aircraft models are highly nonlinear and can be generally given as follows:

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

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

where, Eq. 2 is the kinematic state equation which provides the physical evaluation of the state vector over time; the term of $\mathbf{w}(t)$ is the noise vector caused by input noise; eq. 3 is the observation (output) equation which can be measured using sensors; $\mathbf{v}(t)$ is the output noise vector.

As an application for this control algorithm, only elevator deflection will be regulated as pitch control to stabilize the aircraft and to remain the fight in the wings-level flight condition. 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 given based on the assumption that it remains in steady wings-level flight condition, as follows:

$$\dot{\alpha} = q + \frac{\bar{q}S}{mV_T} C_z(\alpha, q, M, \delta_e) \tag{4}$$

$$\dot{q} = \frac{\bar{q}S\bar{c}}{I_{yy}} C_m(\alpha, q, M, \delta_e) \tag{5}$$

where, C_z is the aerodynamic force coefficient, and C_m is the aerodynamic moment coefficient. They are highly nonlinear functions of *angle of attack* α, *pitch rate* q, *Mach number* M and *elevator deflection* δ_e.

In this paper, external disturbances will not be considered, i.e. $\mathbf{w}(t) = 0$; the output / sensor measurement is accurate, i.e. $\mathbf{v}(t) = 0$, but may not be sufficient; and the sample frequency of simulations is selected to be 100 Hz.

2.3 Output-feedback ADP algorithm

The OPFB algorithms are first introduced to deal with deterministic linear time-invariant (LTI) systems in a discrete-time (DT) system form[6]. It is a promising method, because the algorithms do not require any knowledge of the system dynamics neither the full state, but only the measurements of input and output data. In this section, an introduction of this ADP method and a review and discussion of OPFB VI algorithms will be given.

2.3.1 Output-feedback algorithms for an LTI system

Assumption 1. This system is an LTI system.

The sample frequency of data measurement is 100 Hz, thus the following discrete-time LTI system will be considered:

$$\mathbf{x}_{t+1} = A\mathbf{x}_t + B\mathbf{u}_t,\ \mathbf{y}_t = C\mathbf{x}_t \tag{6}$$

where, $\mathbf{x}_t \in \mathscr{R}^n$, $\mathbf{y}_t \in \mathscr{R}^p$ and $\mathbf{u}_t \in \mathscr{R}^m$ denoting the inertial states, the system outputs / observation and the control inputs, respectively. The *one-step cost function* is defined quadratically as:

$$r_t = r(\mathbf{y}_t, \mathbf{u}_t, \mathbf{d}_t) = \widetilde{r}(\mathbf{y}_t, \mathbf{d}_t) + \mathbf{u}_t^T R\mathbf{u}_t = (\mathbf{y}_t - \mathbf{d}_t)^T Q(\mathbf{y}_t - \mathbf{d}_t) + \mathbf{u}_t^T R\mathbf{u}_t \tag{7}$$

where, $\widetilde{r}(\mathbf{y}_t, \mathbf{d}_t)$ represents a *cost for the current outputs* $\mathbf{y}_t$ approaching the desired outputs $\mathbf{d}_t$, defined by a quadratic form; Q and R are positive definite matrices. When it works as a regulator, the desired outputs are zero; Eq. 7 can be rewritten as follows:

$$r_k = \mathbf{y}_k^T Q\mathbf{y}_k + \mathbf{u}_k^T R\mathbf{u}_k \tag{8}$$

Assumption 2. This system (A,B) is controllable; (A,C) and $(A,C\sqrt{Q})$ is observable.

Considering the infinite horizon, the *state-value function under certain policy* π of state $\mathbf{x}_t$ is cumulative subsequent one-step cost:

$$\begin{aligned} V^\pi(\mathbf{x}_t) = \sum_{i=t}^{\infty} \gamma^{i-t} r_i &= \sum_{i=t}^{\infty} \gamma^{i-t} (\mathbf{y}_i^T Q\mathbf{y}_i + \mathbf{u}_i^T R\mathbf{u}_i) \\ &= (\mathbf{y}_t^T Q\mathbf{y}_t + \mathbf{u}_t^T R\mathbf{u}_t) + \gamma V^\pi(\mathbf{x}_{t+1}) \end{aligned} \tag{9}$$

The Bellman optimality equation for the *optimal state-value function* V^* is shown as follows:

$$\begin{aligned} V^*(\mathbf{x}_t) &= \min_{\pi} \sum_{i=t}^{\infty} \gamma^{i-t} (\mathbf{y}_i^T Q \mathbf{y}_i + \mathbf{u}_i^T R \mathbf{u}_i) \\ &= \min_{\mathbf{u}_t} ((\mathbf{y}_t^T Q \mathbf{y}_t + \mathbf{u}_t^T R \mathbf{u}_t) + \gamma V^*(\mathbf{x}_{t+1})) \end{aligned} \tag{10}$$

This problem is known as the linear-quadratic regulator (LQR) control problem; and for this case, the value function is approximated to be quadratic in the state for some symmetric matrix P:

$$\begin{aligned} \widehat{V}^{\pi}(\mathbf{x}_t) &= \mathbf{x}_t^T P \mathbf{x}_t \\ &= (\mathbf{y}_t^T Q \mathbf{y}_t + \mathbf{u}_t^T R \mathbf{u}_t) + \gamma \mathbf{x}_{t+1}^T P \mathbf{x}_{t+1} \end{aligned} \tag{11}$$

By using linear feedback gains as shown in Eq. 12, there is a positive definite solution by solving Lyapunov equation[6], because (A, C) is assumed to be observable.

$$\mathbf{u}_t = \pi(\mathbf{x}_t) = -K\mathbf{x}_t \tag{12}$$

Both VI algorithm and PI algorithm require the knowledge of system dynamics (A, B) and the full state, while Q-learning provides online algorithms for optimal control problem without knowledge of system dynamics but still need measurement of the full state. Thus, OPFB algorithms[6] were proposed for RL that do not require knowledge of the system model neither the full system states. Instead, they require only input and output data . This method proved that the system states can be given uniquely in terms of the input and output measurement sequences on a time horizon $[t-N, t]$ under Assumption 2 as shown below .

$$\mathbf{x}_t = \begin{bmatrix} M_u & M_y \end{bmatrix} \begin{bmatrix} \overline{\mathbf{u}}_{t-1,t-N} \\ \overline{\mathbf{y}}_{t-1,t-N} \end{bmatrix} \tag{13}$$

where, M_u, M_y is obtained by theoretical derivation from system dynamics (A, B, C), controllability matrix and observability matrix[6]; N is the *time horizon of history information* which needs to be long-enough to reconstruct the full state; $\overline{\mathbf{u}}_{t-1,t-N} \in \mathscr{R}^{pN}$ and $\overline{\mathbf{y}}_{t-1,t-N} \in \mathscr{R}^{mN}$ are the *input and output historic information vectors*:

$$\overline{\mathbf{u}}_{t-1,t-N} = \begin{bmatrix} \mathbf{u}_{t-1} \\ \mathbf{u}_{t-2} \\ \cdots \\ \mathbf{u}_{t-N} \end{bmatrix}, \ \overline{\mathbf{y}}_{t-1,t-N} = \begin{bmatrix} \mathbf{y}_{t-1} \\ \mathbf{y}_{t-2} \\ \cdots \\ \mathbf{y}_{t-N} \end{bmatrix} \tag{14}$$

Thus, a new-defined vector of the observed historic information at time t is $\overline{\mathbf{z}}_{t-1,t-N} \in \mathscr{R}^{(m+p)N}$ as shown below:

$$\overline{\mathbf{z}}_{t-1,t-N} = \begin{bmatrix} \overline{\mathbf{u}}_{t-1,t-N} \\ \overline{\mathbf{y}}_{t-1,t-N} \end{bmatrix} \tag{15}$$

The quadratic cost function now can be expressed in terms of the observed historic information:

$$\begin{aligned}\widehat{V}^{\pi}(\mathbf{x}_t) &= \mathbf{x}_t^T P \mathbf{x}_t \\ &= \bar{\mathbf{z}}_{t-1,t-N}^T \begin{bmatrix} M_u^T P M_u & M_u^T P M_y \\ M_y^T P M_u & M_y^T P M_y \end{bmatrix} \bar{\mathbf{z}}_{t-1,t-N} \\ &= \bar{\mathbf{z}}_{t-1,t-N}^T \overline{P} \bar{\mathbf{z}}_{t-1,t-N}\end{aligned} \tag{16}$$

where, $\overline{P} \in \mathscr{R}^{(m+p)N\times(m+p)N}$ is the *new kernel matrix* in terms of observed historic information. And the Bellman optimality equation and TD error are then written as follows:

$$V^*(\mathbf{x}_t) = \min_{\mathbf{u}_t} ((\mathbf{y}_t^T Q \mathbf{y}_t + \mathbf{u}_t^T R \mathbf{u}_t) + \gamma \bar{\mathbf{z}}_{t,t-N+1}^T \overline{P} \bar{\mathbf{z}}_{t,t-N+1}) \tag{17}$$

$$err = \mathbf{y}_t^T Q \mathbf{y}_t + \mathbf{u}_t^T R \mathbf{u}_t + \gamma \bar{\mathbf{z}}_{t,t-N+1}^T \overline{P} \bar{\mathbf{z}}_{t,t-N+1} - \bar{\mathbf{z}}_{t-1,t-N}^T \overline{P} \bar{\mathbf{z}}_{t-1,t-N} \tag{18}$$

With TD error, learning $\overline{P}$ online is possible by using PI, VI or other TD algorithms. The cost function minimization can be achieved by differentiating with respect to $\mathbf{u}_t$, and we can get the optimal policy under the estimation of $\overline{P}$[6]:

$$\begin{aligned}\pi : \mathbf{u}_t &= -(R+\gamma P_{11})^{-1}\gamma(P_{12}\bar{\mathbf{u}}_{t-1,t-N+1} + P_{13}\bar{\mathbf{y}}_{t,t-N+1}) \\ &= K_u \bar{\mathbf{u}}_{t-1,t-N+1} + K_y \bar{\mathbf{y}}_{t,t-N+1}\end{aligned} \tag{19}$$

where, $P_{11} \in \mathscr{R}^{m\times m}$, $P_{12} \in \mathscr{R}^{m\times m(N-1)}$ and $P_{13} \in \mathscr{R}^{m\times pN}$ are block matrices partitioned from the first m row(s) of $\overline{P} \in \mathscr{R}^{(m+p)N\times(m+p)N}$.

2.3.2 An application on a linearized F-16 model

In this part, a short period reduced LTI model of F-16 fighter aircraft without actuator dynamics will be used. This model is made by trimming and linearizing the non-linear model for only one flight condition. This stable LTI system is discretized, and the discrete-time LTI system is given below:

$$\begin{aligned}\begin{bmatrix}\alpha_{t+1} \\ q_{t+1}\end{bmatrix} &= \begin{bmatrix} 0.9935 & 0.0093 \\ -0.0156 & 0.9912 \end{bmatrix} \begin{bmatrix}\alpha_t \\ q_t\end{bmatrix} + \begin{bmatrix} -1.8861e-5 \\ -0.0011 \end{bmatrix} \delta_e \\ y_t &= 57.3 \begin{bmatrix} c_1 & c_2 \end{bmatrix} \begin{bmatrix}\alpha_t \\ q_t\end{bmatrix}\end{aligned} \tag{20}$$

The open-loop poles are $0.992 \pm 0.012i$; Q and R is 10 and 1, respectively, denoting the weights on outputs and inputs; the discount factor γ is chosen to be 0.2; the historic horizon N is 2. The output matrix $C = 57.3[c_1\ c_2]$ indicates that the output is a combination of two states. This observation equation may not be truly practical,

but the purpose of this experiment set-up is to investigate the applicability of the algorithms. In this paper, we choose $C = 57.3[1\ 1]$ to regulate both α and q.

By using the OPFB VI algorithm with 3211 input signals and batch least square (batch LS) in solving $\overline{P}$, the results are shown in Fig. 1 and the policy converges (Fig. 2).

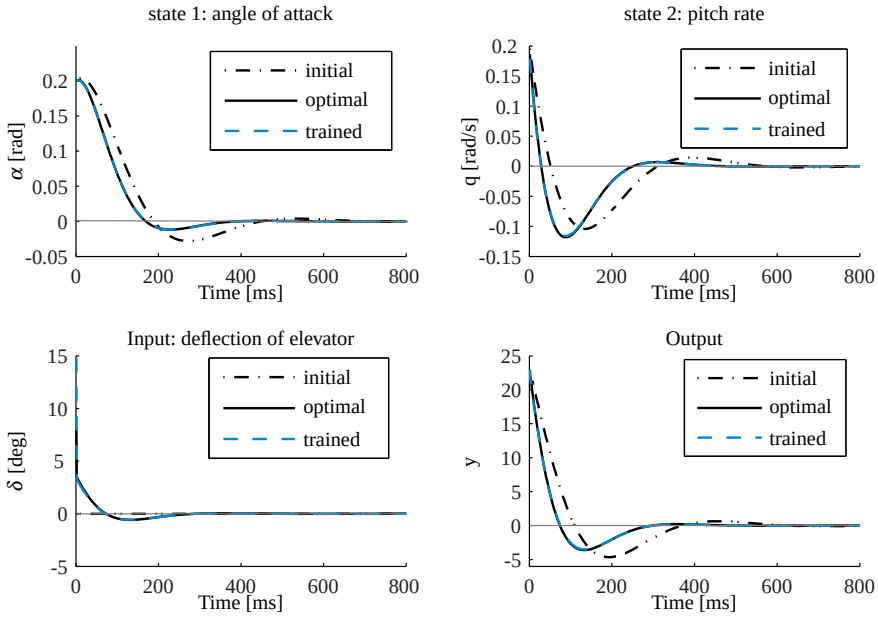

Fig. 1: Linear OPBF VI method applied to linearized F-16 model.

2.3.3 Discussion on OPFB method for F-16 LTI model

The control inputs of F-16 has maximum values which limit the simulation to only exploring part of the state space.

Input design. As with other RL methods, good value 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: pseudo-random noise, classical sine waves, doublets, 3211 doublets, etc. In this paper, 3211 maneuvers, which are one of the most commonly used maneuvers for aircraft system identification, are used for both LTI system and non-linear F-16 model.

For a closed loop system, an input noise/design will lead to a bias on value estimation in most of the cases, e.g. quadratic problems, because it is an adding term in

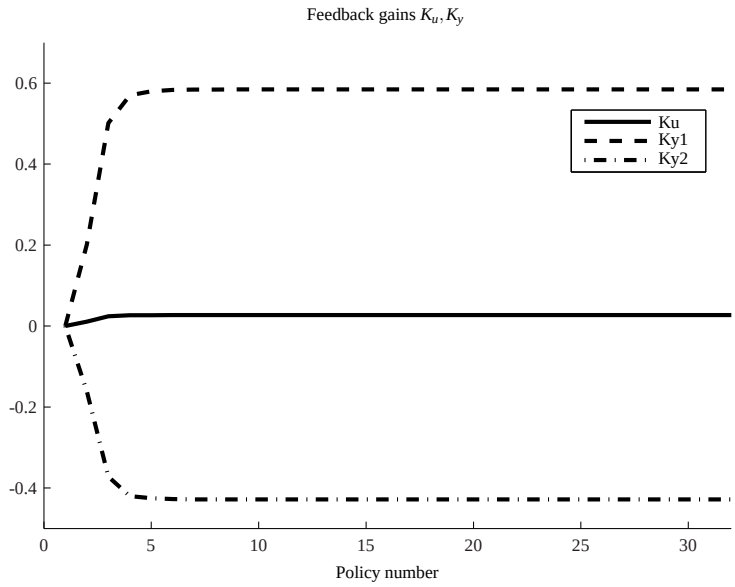

Fig. 2: Policy parameters during training with linear OPBF VI method.

quadratic form which is cumulative. That's why a discount factor was added in our algorithms.

Local optimal. Without considering the limitation of control deflections, the optimal value function approximation is possible to obtain by using a very restrict input design compensating effects of input noise. However, the optimal estimation require different well-designed inputs for different LTI systems.

By looking at Fig. 1, the trained controller performed almost the same as the optimal controller, even slightly better, because the trained controller gains converged to the local optimal solution which explored only the limited input space and often-experienced state space. In this sense, this local optimal controller is good enough for our applications.

Linear system. This OPFB algorithms above is only suitable for LTI systems. It is not functional well (more overshoot, slower convergence and non-zero static error) when applied directly to F-16 non-linear system, as shown in Fig. 3, even though the policy parameters converges (Fig. 4).

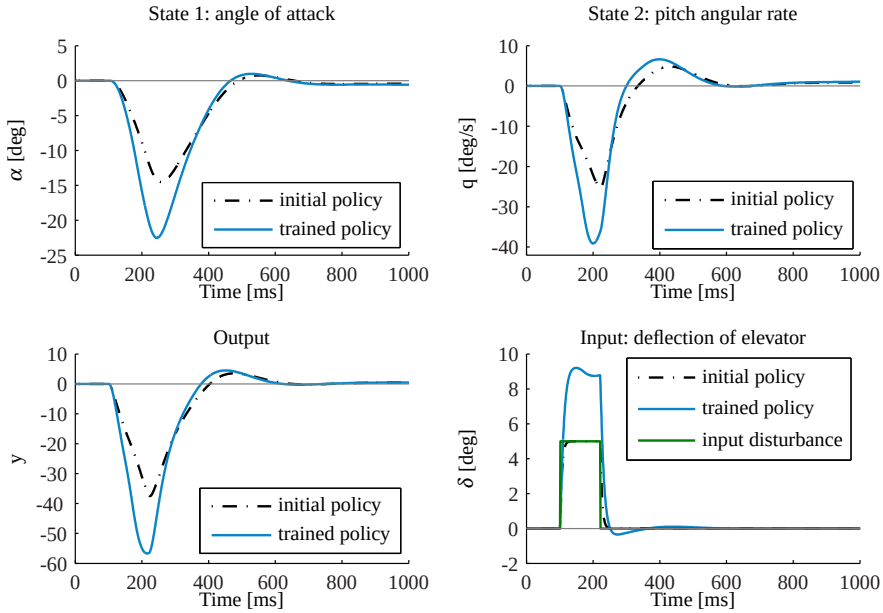

Fig. 3: Linear OPBF VI method applied to non-linear F-16 model.

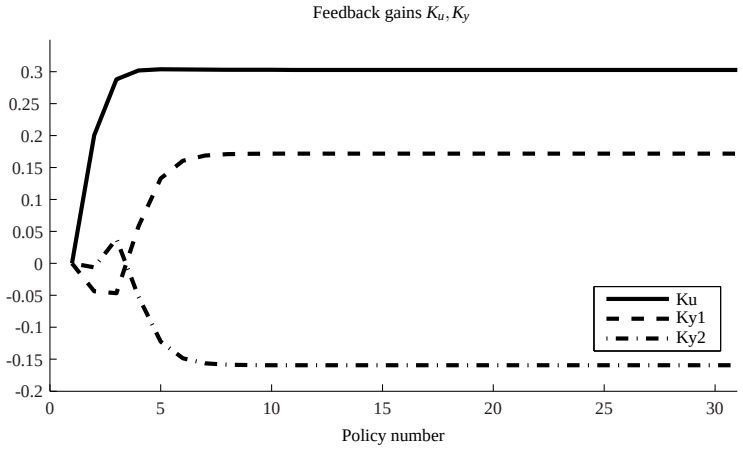

Fig. 4: Policy (feedback gains) during training with linear OPBF VI method.

3 Incremental Approximate Dynamic Programming

3.1 Incremental methods

The incremental methods have the ability to deal with the nonlinearities of systems. It computes the required control increment at a certain moment using the conditions

of the system in the instant before[8]. The main idea of incremental methods is shown as follows.

Considering a non-linear continuous system dynamics below:

$$\dot{\mathbf{x}}(t) = f[\mathbf{x}(t), \mathbf{u}(t)],\ \mathbf{y}(t) = h[\mathbf{x}(t)] \tag{21}$$

where $f(\mathbf{x}(t), \mathbf{u}(t)) \in \mathscr{R}^n$ provides the physical evaluation of the state vector over time; $h(\mathbf{x}(t)) \in \mathscr{R}^p$ is a vector denoting the measure system.

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:

$$\begin{aligned}\dot{\mathbf{x}} &\simeq f(\mathbf{x}_0, \mathbf{u}_0) + \frac{\partial f[\mathbf{x}(t), \mathbf{u}(t)]}{\partial \mathbf{x}(t)}|_{\mathbf{x}_0, \mathbf{u}_0}(\mathbf{x}(t) - \mathbf{x}_0) + \frac{\partial f[\mathbf{x}(t), \mathbf{u}(t)]}{\partial \mathbf{u}(t)}|_{\mathbf{x}_0, \mathbf{u}_0}[\mathbf{u}(t) - \mathbf{u}_0] \\ &= \dot{\mathbf{x}}(t)_0 + F(\mathbf{x}_0, \mathbf{u}_0)[\mathbf{x}(t) - \mathbf{x}_0] + G(\mathbf{x}_0, \mathbf{u}_0)[\mathbf{u}(t) - \mathbf{u}_0]\end{aligned} \tag{22}$$

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

Assumption 3. The states and state derivatives of the system are measurable, which means $\mathbf{y}(t) = h[\mathbf{x}(t)] = \mathbf{x}(t)$ and $\Delta\dot{\mathbf{x}}(t), \Delta\mathbf{x}(t), \Delta\mathbf{u}(t)$ are measurable.

The model can be written in the incremental form (Eq. 23), which is a regression model and can be identified using recursive least squares (RLS) technique.

$$\Delta\dot{\mathbf{x}}(t) \simeq F(\mathbf{x}_0, \mathbf{u}_0)\Delta\mathbf{x}(t)) + G(\mathbf{x}_0, \mathbf{u}_0)\Delta\mathbf{u}(t) \tag{23}$$

3.2 Incremental ADP algorithm based on full state feedback

Assumption 4. The control system has a constant high sampling frequency which is 100Hz.

Given a constant data sampling rate (assumption 4), the non-linear F-16 model can be written in a discrete form:

$$\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t),\ \mathbf{y}_t = h(\mathbf{x}_t) \tag{24}$$

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

Linearize the system dynamics around x^* by taking the Taylor expansion:

$$\mathbf{x}_{t+1} \simeq f(\mathbf{x}^*, \mathbf{u}^*) + \frac{\partial f(\mathbf{x}_t, \mathbf{u}_t)}{\partial \mathbf{x}_t}|_{\mathbf{x}^*, \mathbf{u}^*}(\mathbf{x}_t - \mathbf{x}^*) + \frac{\partial f(\mathbf{x}_t, \mathbf{u}_t)}{\partial \mathbf{u}_t}|_{\mathbf{x}^*, \mathbf{u}^*}(\mathbf{u}_t - \mathbf{u}^*) \tag{25}$$

When Δt is very small (assumption 4), $\mathbf{x}_{t-1}$ approximates $\mathbf{x}_t$, thus, $\mathbf{x}^*, \mathbf{u}^*$ in Eq. 25 are replaced by $\mathbf{x}^* = \mathbf{x}_{t-1}$ and $\mathbf{u}^* = \mathbf{u}_{t-1}$, and we obtain the discrete incremental form of this non-linear system:

$$\mathbf{x}_{t+1} - \mathbf{x}_t \simeq F(\mathbf{x}_{t-1}, \mathbf{u}_{t-1})(\mathbf{x}_t - \mathbf{x}_{t-1}) + G(\mathbf{x}_{t-1}, \mathbf{u}_{t-1})(\mathbf{u}_t - \mathbf{u}_{t-1}) \tag{26}$$

$$\Delta\mathbf{x}_{t+1} \simeq F(\mathbf{x}_{t-1}, \mathbf{u}_{t-1})\Delta\mathbf{x}_t + G(\mathbf{x}_{t-1}, \mathbf{u}_{t-1})\Delta\mathbf{u}_t \tag{27}$$

where, $F(\mathbf{x}_{t-1}, \mathbf{u}_{t-1}) = \frac{\partial f(\mathbf{x},\mathbf{u})}{\partial \mathbf{x}}|_{\mathbf{x}_{t-1},\mathbf{u}_{t-1}} \in \mathscr{R}^{n\times n}$ is the *system matrix*; $G(\mathbf{x}_{t-1}, \mathbf{u}_{t-1}) = \frac{\partial f(\mathbf{x}_t,\mathbf{u}_t)}{\partial \mathbf{u}_t}|_{\mathbf{x}_{t-1},\mathbf{u}_{t-1}} \in \mathscr{R}^{n\times m}$ 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 measured data in previous M steps.

The one-step cost function at time t is in a quadratic form as shown below:

$$r_t = r(\mathbf{y}_t, \mathbf{u}_t) = \mathbf{y}_t^T Q\mathbf{y}_t + \mathbf{u}_t^T R\mathbf{u}_t \tag{28}$$

For infinite horizon, the state-value function is the cumulative future rewards (Eq. 29) from any initial state $\mathbf{x}_t$.

$$\begin{aligned} V^{\mu}(\mathbf{x}_t) &= \sum_{i=t}^{\infty} \gamma^{i-t}(y_i^T Q y_i + u_i^T R u_i) \\ &= (\mathbf{y}_t^T Q\mathbf{y}_t + \mathbf{u}_t^T R\mathbf{u}_t) + \gamma V^{\mu}(\mathbf{x}_{t+1}) \\ &= \mathbf{y}_t^T Q\mathbf{y}_t + (\mathbf{u}_{t-1} + \Delta\mathbf{u}_t)^T R(\mathbf{u}_{t-1} + \Delta\mathbf{u}_t) + \gamma V^{\mu}(\mathbf{x}_{t+1}) \end{aligned} \tag{29}$$

where, μ is the *current policy* for this iADP algorithm, so as to distinguish from policy π in OPFB algorithms. The optimal value function for the *optimal policy* μ^* is defined as follows:

$$V^*(\mathbf{x}_t) = \underset{\Delta\mathbf{u}_t}{min} \left[\mathbf{y}_t^T Q\mathbf{y}_t + (\mathbf{u}_{t-1} + \Delta\mathbf{u}_t)^T R(u_{t-1} + \Delta\mathbf{u}_t) + \gamma V^*(\mathbf{x}_{t+1})\right] \tag{30}$$

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

$$\Delta\mathbf{u}_t = \mu(\mathbf{u}_{t-1}, \mathbf{x}_t, \Delta\mathbf{x}_t) \tag{31}$$

The optimal control at time t can be given:

$$\mu^*(\Delta\mathbf{x}_t) = \underset{\Delta\mathbf{u}_t}{argmin} \left[\mathbf{y}_t^T Q\mathbf{y}_t + (\mathbf{u}_{t-1} + \Delta\mathbf{u}_t)^T R(u_{t-1} + \Delta\mathbf{u}_t) + \gamma V^*(\mathbf{x}_{t+1})\right] \tag{32}$$

For this case, the state value is the sum of quadratic value in the output and input. Thus, the value function approximation is also chosen in a quadratic form:

$$\widehat{V}^{\mu}(\mathbf{x}_t) = \mathbf{x}_t^T P\mathbf{x}_t \tag{33}$$

where, P is a positive definite matrix.

Therefore, the LQR Bellman equation can be written in the incremental form:

$$\begin{aligned}\mathbf{x}_t^T P\mathbf{x}_t &= \mathbf{y}_t^T Q\mathbf{y}_t + (\mathbf{u}_{t-1}+\Delta\mathbf{u}_t)^T R(u_{t-1}+\Delta\mathbf{u}_t) + \gamma\mathbf{x}_{t+1}^T P\mathbf{x}_{t+1}\\ &= \mathbf{y}_t^T Q\mathbf{y}_t + (\mathbf{u}_{t-1}+\Delta\mathbf{u}_t)^T R(u_{t-1}+\Delta\mathbf{u}_t)\\ &+ \gamma(\mathbf{x}_t + F_{t-1}\Delta\mathbf{x}_t + G_{t-1}\Delta\mathbf{u}_t)^T P(\mathbf{x}_t + F_{t-1}\Delta\mathbf{x}_t + G_{t-1}\Delta\mathbf{u}_t)\end{aligned} \tag{34}$$

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

$$\begin{aligned}\Delta\mathbf{u}_t &= -(R+\gamma G_{t-1}^T P G_{t-1})^{-1}[R\mathbf{u}_{t-1} + \gamma G_{t-1}^T P(\mathbf{x}_t + F_{t-1}\Delta\mathbf{x}_t)]\\ &= -(R+\gamma G_{t-1}^T P G_{t-1})^{-1}[R\mathbf{u}_{t-1} + \gamma G_{t-1}^T P\mathbf{x}_t + \gamma G_{t-1}^T P F_{t-1}\Delta\mathbf{x}_t]\end{aligned} \tag{35}$$

from which, the policy is in the form of system variables $(\mathbf{u}_{t-1}, \mathbf{x}_t, \Delta\mathbf{x}_t)$ feedback, and the gain is a function of the dynamics of the current linearized system (F_{t-1}, G_{t-1}).

With assumption 3, $\Delta\mathbf{x}(t), \Delta\mathbf{u}(t)$ are measurable; F_{t-1}, G_{t-1} are identifiable by using simple equation error method:

$$\Delta x_{i,t-k+1} = \mathbf{f}_i\Delta\mathbf{x}_{t-k} + \mathbf{g}_i\Delta\mathbf{u}_{t-k} = \begin{bmatrix}\Delta\mathbf{x}_{t-k}^T & \Delta\mathbf{u}_{t-k}^T\end{bmatrix}\begin{bmatrix}\mathbf{f}_i^T\\ \mathbf{g}_i^T\end{bmatrix} \tag{36}$$

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

$$\begin{bmatrix}\mathbf{f}_i^T\\ \mathbf{g}_i^T\end{bmatrix} = (\mathbf{A}_t^T\mathbf{A}_t)^{-1}\mathbf{A}_t^T\mathbf{y}_t \tag{37}$$

where,

$$\mathbf{A}_t = \begin{bmatrix}\Delta\mathbf{x}_{t-1}^T & \Delta\mathbf{u}_{t-1}^T\\ \vdots & \vdots\\ \Delta\mathbf{x}_{t-N}^T & \Delta\mathbf{u}_{t-N}^T\end{bmatrix},\ \mathbf{y}_t = \begin{bmatrix}\Delta x_{i,t}\\ \vdots\\ \Delta x_{i,t+N-1}\end{bmatrix} \tag{38}$$

If the nonlinear system is unknown, while the full state is measurable, ADP algorithm, such as PI or VI, combined with incremental method can be applied to improve the policy online:

iADP (incremental Approximate Dynamic Programming) algorithm.

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

$$\mathbf{x}_t^T P^{j+1}\mathbf{x}_t = \mathbf{y}_t^T Q\mathbf{y}_t + \mathbf{u}_t^T R\mathbf{u}_t + \gamma\mathbf{x}_{t+1}^T P^j\mathbf{x}_{t+1} \tag{39}$$

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

$$\Delta\mathbf{u}_t = -(R+\gamma G_{t-1}^T P G_{t-1})^{-1}[R\mathbf{u}_{t-1} + \gamma G_{t-1}^T P^{j+1}\mathbf{x}_t + \gamma G_{t-1}^T P^{j+1} F_{t-1}\Delta\mathbf{x}_t] \tag{40}$$

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

4 Implementation

Considering the F-16 nonlinear model again, the iADP algorithms above can be used. Fig. 5 below shows the disturbance response when a block input disturbance is introduced. Disturbances are undesirable inputs, the control system with iADP algorithm have a higher dynamic stiffness and lower disturbance response compared to initial system and LTI OPFB controller.

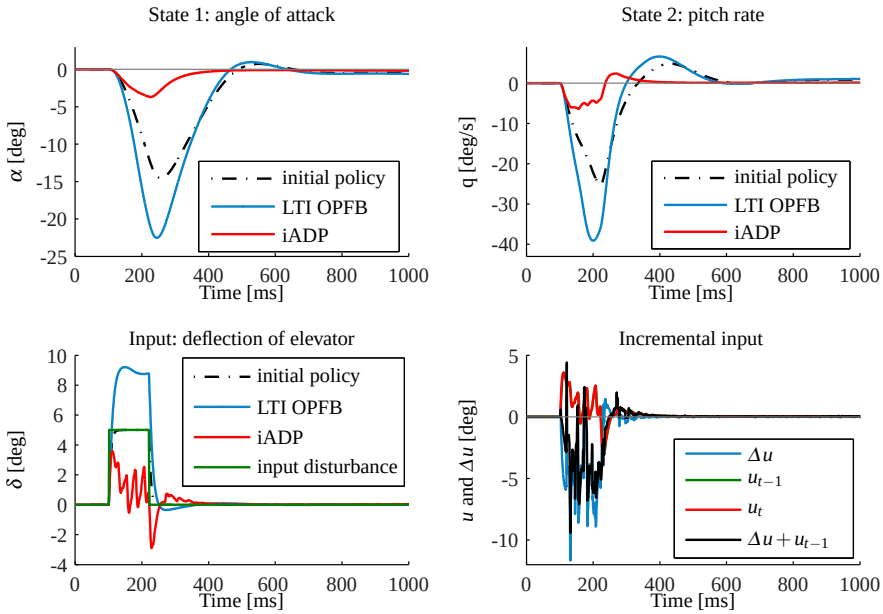

Fig. 5: IADP method applied to non-linear F-16 model with a block input disturbance.

Fig. 6 shows the control performance when the initial state is an offset from trimmed condition and compares its result with that of LTI OPFB controller. Without persistent excitation, the nonlinear system 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 method (Eq. 37). The bottom right figures in Fig. 5 and

Fig. 6 illustrate how the computed increment changes the control input (elevator deflection) and how the deflection $\mathbf{u}_t$ is filtered by the actuator.

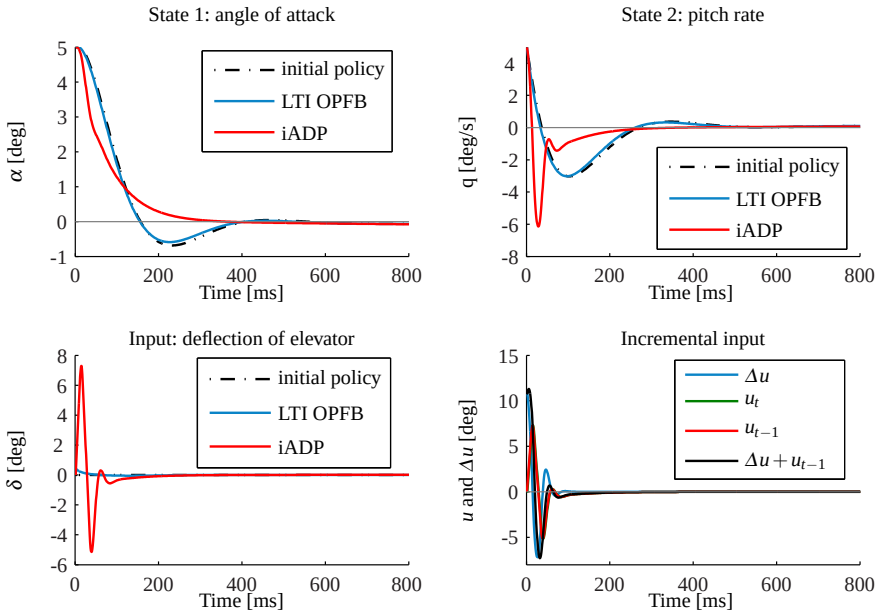

Fig. 6: IADP method applied to non-linear F-16 model with an initial offset.

This control method does not need the model of the non-linear system, but still need the full state to estimate the value function and the control effectiveness matrix.

5 Conclusion and Future Works

In this paper, an adaptive control method for non-linear system was discussed. Approximate dynamic programming algorithms provide a linear approach solving a near-optimal policy without knowing the system dynamics; while the incremental nonlinear control technique linearizes the system. An incremental approximate dynamic programming algorithm was proposed to design an adaptive near-optimal controller for non-linear system. The flight control using iADP algorithm was applied to an F-16 non-linear system. The simulation results validated that non-linear flight controller designed by using iADP method performs much better than the one using OPFB method. Furthermore, this algorithm is robust, because it doesn't need any information of the system model. But it still requires full state of the system which are often not available, such as aircraft systems. The disturbance on sensors,

e.g. noise, amplification, interaction, etc., will lead to unreadable output measurement. Under certain policy, the full state of a linear system can be reconstructed by using previous inputs and outputs information. Further improvements could be combining iADP algorithm and OPFB method and conducting experiments with different types of value functions. This proposed method can potentially design a near-optimal controller for non-linear systems without a-prior knowledge of the dynamic model.

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

References

[1] Richard S Sutton and Andrew G Barto. *Introduction to reinforcement learning*. MIT Press, 1998.
[2] Richard Bellman. *Dynamic Programming*. Princeton University Press, 1957.
[3] Zachary A. Harris Scott A. Miller and Edwin K.P. Chong. A POMDP framework for coordinated guidance of autonomous uavs for multitarget tracking. *EURASIP Journal on Advances in Signal Processing*, 2009.
[4] Shankarachary Ragi and Edwin KP Chong. UAV path planning in a dynamic environment via partially observable markov decision process. *IEEE Transactions on Aerospace and Electronic Systems*, 49(4):2397–2412, 2013.
[5] Ruijie He, Emma Brunskill, and Nicholas Roy. Efficient planning under uncertainty with macro-actions. *Journal of Artificial Intelligence Research*, 40 (1):523–570, 2011.
[6] Frank L Lewis and Kyriakos G Vamvoudakis. Reinforcement learning for partially observable dynamic processes: Adaptive dynamic programming using measured output data. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 41(1):14–25, 2011.
[7] S Sieberling, QP Chu, and JA Mulder. Robust flight control using incremental nonlinear dynamic inversion and angular acceleration prediction. *Journal of guidance, control, and dynamics*, 33(6):1732–1742, 2010.
[8] P Simplício, MD Pavel, E Van Kampen, and QP Chu. An acceleration measurements-based approach for helicopter nonlinear flight control using incremental nonlinear dynamic inversion. *Control Engineering Practice*, 21(8): 1065–1077, 2013.
[9] Lucian Busoniu, Robert Babuska, Bart De Schutter, and Damien Ernst. *Reinforcement learning and dynamic programming using function approximators*. CRC Press, 2010.
[10] Kenji Doya. Reinforcement learning in continuous time and space. *Neural computation*, 12(1):219–245, 2000.

[11] E van Kampen, QP Chu, and JA Mulder. Continuous adaptive critic flight control aided with approximated plant dynamics. In *Proc AIAA Guidance Navig Control Conf*, volume 5, pages 2989–3016, 2006.

[12] Said G Khan, Guido Herrmann, Frank L Lewis, Tony Pipe, and Chris Melhuish. Reinforcement learning and optimal adaptive control: An overview and implementation examples. *Annual Reviews in Control*, 36(1):42–59, 2012.

[13] Jennie Si. *Handbook of learning and approximate dynamic programming*, volume 2. John Wiley & Sons, 2004.

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

[15] Richard S Russell. Non-linear F-16 simulation using simulink and matlab. *University of Minnesota, Tech. paper*, 2003.

[16] Brian L Stevens and Frank L Lewis. *Aircraft control and simulation*. John Wiley & Sons, 2003.

[17] Ogburn M. E. Gilbert W. P. Kibler K. S. Brown P. W. Deal P. L. Nguyen, L. T. Simulator study of stall/post-stall characteristics of a fighter airplane with relaxed longitudinal static stability. Technical report, Tech. Rep. 1538, NASA, 1979.