68th International Astronautical Congress(IAC), Adelaide, Australia, 25-29 September 2017.

IAC-17-C1.2.5

ADAPTIVE SPACECRAFT ATTITUDE CONTROL WITH INCREMENTAL APPROXIMATE DYNAMIC PROGRAMMING

Ye Zhou [a] **, Erik-Jan van Kampen** [b] **, Qi Ping Chu** [c]

[a] *Aerospace Engineering, Delft University of Technology, 2629HS Delft, The Netherlands, Y.Zhou-6@tudelft.nl.*
[b] *Aerospace Engineering, Delft University of Technology, 2629HS Delft, The Netherlands, E.vanKampen@tudelft.nl.*
[c] *Aerospace Engineering, Delft University of Technology, 2629HS Delft, The Netherlands, Q.P.Chu@tudelft.nl.*

Abstract

This paper presents an adaptive control technique to deal with spacecraft attitude tracking and disturbance rejection problems in the presence of model uncertainties. Approximate dynamic programming has been proposed to solve adaptive, optimal control problems without using accurate systems models. Within this category, linear approximate dynamic programming systematically utilizes a quadratic cost-to-go function and simplifies the design process. Although model-free and efficient, linear approximate dynamic programming methods are difficult to apply to nonlinear systems or time-varying systems, such as attitude control of spacecraft disturbed by internal liquid sloshing. To deal with this problem, this paper develops a model-free nonlinear self-learning attitude control method based on incremental Approximate Dynamic Programming to enhance the performance of the spacecraft attitude control system. This method combines the advantages of linear approximate dynamic programming and the incremental nonlinear control techniques, and generates a model-free controller for unknown, time-varying dynamical systems. In this paper, two reference tracking algorithms are developed for off-line learning and online learning, respectively. These algorithms are applied to the attitude control of a spacecraft disturbed by internal liquid sloshing. The results demonstrate that the proposed method deals with the unknown, time-varying internal dynamics adaptively while retaining accurate and efficient attitude control.

Keywords: Nonlinear control, Adaptive control, Approximate Dynamic Programming, Incremental techniques, Fuel sloshing.

1. Introduction

Spacecraft attitude control has been an active field of research for decades. Researchers have extensively studied non-adaptive nonlinear control methods for rigid satellites, or with flexible parts that can be modeled, such as solar panels. However, when unknown or uncertain dynamics are involved, the mismatch between the model and real system may degrade the performance of model-based methods. Liquid sloshing is one of the unknown and uncertain dynamics interacting with the motion of the spacecraft [1, 2, 3].

Unknown liquid sloshing causes unobservable internal dynamics. A typical case is that the liquid in the fuel tank sloshes around and changes the internal dynamics. The forces and torques acting onto the spacecraft will slosh the fuel around. The fuel will, in turn, interact with the fuel tank and thus produce additional forces and torques to the spacecraft, which degrades the performance of attitude control systems. Although the fuel sloshing has been studied for years, an accurate liquid sloshing model is extremely difficult to obtain [3, 4]. Therefore, the aim of this paper is to develop a model-free nonlinear self-learning attitude control method in order to improve the performance.

Reinforcement Learning (RL) [5, 6] controllers have been proposed to solve adaptive, optimal control problems without using accurate system models. In the control field, RL is also referred to as Approximate Dynamic Programming (ADP), which applies function approximators to solve problems with continuous state and action spaces and to tackle the curse of dimensionality. A widely used model-free ADP method for linear systems is the Linear Approximate Dynamic Programming (LADP) method [7, 8]. This method utilizes a quadratic cost-to-go function [9] and adaptively learns it online. Because of the simple quadratic function, it is efficient and gives a complete and explicit solution to optimal control problems. Although the merits of model-free processes and efficiency of resource usage make LADP controllers suitable in the field of adaptive control, LADP methods are difficult to apply to nonlinear systems or time-varying systems.

Incremental control methods, on the other hand, can deal with system nonlinearity without identifying the global system [10, 11, 12, 13, 14]. The incremental form of a nonlinear dynamic system is a linear time-varying approximation of the original system assuming a sufficiently high sample rate for discretization. This technique has been successfully applied to adaptive nonlinear controllers, such as Incremental Nonlinear Dynamic Inversion (INDI) [10, 11, 12, 13] and Incremental Backstepping (IBS) [14] for nonlinear systems. Although these nonlinear control methods have reduced model dependence in the control system, optimization or synthesis of designed closed-loop systems has not been addressed.

This paper develops an innovative model-free nonlinear self-learning attitude control method based on incremental Approximate Dynamic Programming (iADP) to enhance the performance of the spacecraft attitude control systems. This method combines the advantages of LADP methods and the incremental nonlinear control techniques and generates a model-free, effective controller for unknown, time-varying dynamical systems. A Recursive Least Square (RLS) technique is used to identify the incremental model when assuming a sufficiently high sample rate. This method belongs to model-free control because it does not need any a priori information of the system dynamics, online identification of the global nonlinear system nor even an assumption of the time scale separation, but only an identified incremental model.

The remainder of this paper is structured as follows. Section 2 starts with the formulation of the iADP algorithm. Section 3 introduces the nonlinear spacecraft model disturbed with liquid sloshing. Then, section 4 explicates the implementation of the proposed control method and applies to the spacecraft attitude tracking and disturbance rejection problem. Lastly, section 5 shows the advantages and disadvantages of using the iADP and addresses the challenges and possibilities for future research.

2. **Incremental Approximate Dynamic Programming**

The incremental Approximate Dynamic Programming (iADP) method combines the advantages of LADP, which is model-free and effective, and of incremental approach, which expands this method to nonlinear systems. It was first proposed to solve stabilization problems for nonlinear systems [15, 16]. This method, to some extend, can also be seen as an Actor-Critic method, or more specifically as Heuristic Dynamic Programming [17, 18, 19, 20]. The quadratic value function acts as the critic, the incremental model provides the model information, and thus, the actor is an explicit expression to optimal control. Therefore, this separation and adaptability of the incremental model makes this method suitable for time-varying systems. It does not need off-line learning of the global system or extensively tuning of nonlinear approximators. This section will present and elaborate the proposed iADP method for tracking problem.

2.1 Incremental approach

The incremental model at the current moment can be approximated by using the conditions of the system in the instant before [12]. This technique has been successfully applied to deal with the nonlinearities of systems. The dynamic and kinematic equations of a nonlinear system can be generally given as follows:

$$\dot{\mathbf{x}}(t) = f[\mathbf{x}(t), \mathbf{u}(t)], \tag{1}$$

where $\mathbf{x} \in \mathscr{R}^n$ is the system state vector, $\mathbf{u} \in \mathscr{R}^m$ is the control input vector, and $f[\mathbf{x}(t), \mathbf{u}(t)] \in \mathscr{R}^n$ provides the physical evaluation of the state vector over time.

When the system is first-order continuous, 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:

$$\begin{aligned}\dot{\mathbf{x}}(t) \approx \dot{\mathbf{x}}(t_0) &+ F[\mathbf{x}(t_0), \mathbf{u}(t_0)][\mathbf{x}(t) - \mathbf{x}(t_0)] \\ &+ G[\mathbf{x}(t_0), \mathbf{u}(t_0)][\mathbf{u}(t) - \mathbf{u}(t_0)],\end{aligned} \tag{2}$$

where $F[\mathbf{x}(t_0), \mathbf{u}(t_0)] = \frac{\partial f[\mathbf{x}(t), \mathbf{u}(t)]}{\partial \mathbf{x}(t)}|_{\mathbf{x}(t_0), \mathbf{u}(t_0)} \in \mathscr{R}^{n \times n}$ is the system matrix of the linearized model at time t_0, and $G[\mathbf{x}(t_0), \mathbf{u}(t_0)] = \frac{\partial f[\mathbf{x}(t), \mathbf{u}(t)]}{\partial \mathbf{u}(t)}|_{\mathbf{x}(t_0), \mathbf{u}(t_0)} \in \mathscr{R}^{n \times m}$ is the control effectiveness matrix.

Assuming that the control inputs, states, and state derivatives of the system are measurable, which means $\Delta\mathbf{u}$, $\Delta\dot{\mathbf{x}}$, and $\Delta\mathbf{x}$ are measurable, the model around time t_0 can be written in the incremental form:

$$\Delta\dot{\mathbf{x}}(t) \simeq F[\mathbf{x}(t_0), \mathbf{u}(t_0)]\Delta\mathbf{x}(t) + G[\mathbf{x}(t_0), \mathbf{u}(t_0)]\Delta\mathbf{u}(t). \tag{3}$$

Although most physical systems are continuous, their measurements are often discrete. With a sufficiently high, constant data sampling frequency, the before-mentioned nonlinear systems can also be written in a discrete form:

$$\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t). \tag{4}$$

The system dynamics around $\mathbf{x}_t$ can also be linearized by taking the Taylor expansion:

$$\mathbf{x}_{t+1} \approx \mathbf{x}_t + \mathbf{F}_{t-1} \cdot (\mathbf{x}_t - \mathbf{x}_{t-1}) + \mathbf{G}_{t-1} \cdot (\mathbf{u}_t - \mathbf{u}_{t-1}), \tag{5}$$

where $\mathbf{F}_{t-1} = \frac{\partial f(\mathbf{x}, \mathbf{u})}{\partial \mathbf{x}}|_{\mathbf{x}_{t-1}, \mathbf{u}_{t-1}} \in \mathscr{R}^{n \times n}$ is the system transition matrix, and $\mathbf{G}_{t-1} = \frac{\partial f(\mathbf{x}, \mathbf{u})}{\partial \mathbf{u}}|_{\mathbf{x}_{t-1}, \mathbf{u}_{t-1}} \in \mathscr{R}^{n \times m}$ is the input distribution matrix at time step $t-1$ for discretized systems. The incremental form of this discrete nonlinear system can be written as follows:

$$\Delta\mathbf{x}_{t+1} \approx \mathbf{F}_{t-1}\Delta\mathbf{x}_t + \mathbf{G}_{t-1}\Delta\mathbf{u}_t, \tag{6}$$

The nonlinear system can be represented as the time-varying incremental model. This linear model needs to be available online to provide the model information for the iADP algorithm instead of using a global nonlinear system model. With the high-frequency sample data and the relatively slow-varying system assumption, time-varying matrices $\widehat{\mathbf{F}}_{t-1}$ and $\widehat{\mathbf{G}}_{t-1}$ can be identified online using least squares (LS) techniques.

2.2 IADP for reference tracking

The reference tracking can be seen as a control problem for time-varying system, where the tracking error can be seen as the state. Therefore, iADP methods can be used to solve Multiple-Input Multiple-Output (MIMO) tracking problems without the assumption of time-scale separation. To minimize the cost of the system approaching its goal, we first define the one-step cost function quadratically:

$$c(t) = (\mathbf{x}_t - \mathbf{x}_{ref,t})^T Q(\mathbf{x}_t - \mathbf{x}_{ref,t}) + \mathbf{u}_t^T R\mathbf{u}_t. \tag{7}$$

where Q and R are positive definite matrices, and $\mathbf{x}_{ref,t}$ is the reference signal. The cost-to-go function from any initial state $\mathbf{x}_t$ and reference $\mathbf{x}_{ref,t}$ under current policy μ is the cumulative sum of future rewards:

$$\begin{aligned}J^\mu(t) &= \sum_{l=t}^{\infty} \gamma^{l-t}[(\mathbf{x}_l - \mathbf{x}_{ref,l})^T Q(\mathbf{x}_l - \mathbf{x}_{ref,l}) + \mathbf{u}_l^T R\mathbf{u}_l] \\ &= (\mathbf{x}_t - \mathbf{x}_{ref,t})^T Q(\mathbf{x}_t - \mathbf{x}_{ref,t}) + \mathbf{u}_t^T R\mathbf{u}_t + \gamma J^\mu(t+1),\end{aligned} \tag{8}$$

where $\gamma \in (0,1)$ is the forgetting factor.

Considering the incremental form of the system, the optimal cost-to-go function for the optimal policy μ^* is defined as follows:

$$J^*(t) = \min_{\Delta \mathbf{u}_t}[(\mathbf{x}_t - \mathbf{x}_{ref,t})^T Q(\mathbf{x}_t - \mathbf{x}_{ref,t}) \\ + (\mathbf{u}_{t-1} + \Delta \mathbf{u}_t)^T R(\mathbf{u}_{t-1} + \Delta \mathbf{u}_t) + \gamma J^*(t+1)]. \tag{9}$$

Thus, the optimal policy at time t can be given by

$$\mu^* = \arg \min_{\Delta \mathbf{u}_t} [(\mathbf{x}_t - \mathbf{x}_{ref,t})^T Q(\mathbf{x}_t - \mathbf{x}_{ref,t}) \\ + (\mathbf{u}_{t-1} + \Delta \mathbf{u}_t)^T R(\mathbf{u}_{t-1} + \Delta \mathbf{u}_t) + \gamma J^*(t+1)]. \tag{10}$$

For this nonlinear tracking problem, the cost-to-go is the sum of quadratic values in the tracking error and inputs with a forgetting factor. The cost-to-go 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 [9]. A systematic cost function approximation that can be applied in our system in this paper is chosen to be quadratic in the tracking error $\mathbf{e}_t = \mathbf{x}_t - \mathbf{x}_{ref,t}$ for some symmetric, positive definite matrix P, as shown below:

$$\widehat{J}^{\mu}(\mathbf{e}_t) = \mathbf{e}_t^T P \mathbf{e}_t. \tag{11}$$

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 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.

Assuming the reference signal is slow-variant and the sample frequency is sufficiently high, the incremental form of the tracking error can be approximate as follows:

$$\begin{aligned} \mathbf{e}_{t+1} &= \mathbf{x}_{t+1} - \mathbf{x}_{ref,t+1} \\ &\approx \mathbf{x}_t - \mathbf{x}_{ref,t+1} + \mathbf{F}_{t-1}\Delta\mathbf{x}_t + \mathbf{G}_{t-1}\Delta\mathbf{u}_t \\ &\approx \mathbf{x}_t - \mathbf{x}_{ref,t} + \mathbf{F}_{t-1}\Delta\mathbf{x}_t + \mathbf{G}_{t-1}\Delta\mathbf{u}_t \\ &= \mathbf{e}_t + \mathbf{F}_{t-1}\Delta\mathbf{x}_t + \mathbf{G}_{t-1}\Delta\mathbf{u}_t. \end{aligned} \tag{12}$$

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

$$\begin{aligned} \widehat{J}^{\mu}(\mathbf{e}_t) &= \mathbf{e}_t^T Q \mathbf{e}_t + (\mathbf{u}_{t-1} + \Delta\mathbf{u}_t)^T R(\mathbf{u}_{t-1} + \Delta\mathbf{u}_t) + \gamma \mathbf{e}_{t+1}^T P \mathbf{e}_{t+1} \\ &= \mathbf{e}_t^T Q \mathbf{e}_t + (\mathbf{u}_{t-1} + \Delta\mathbf{u}_t)^T R(\mathbf{u}_{t-1} + \Delta\mathbf{u}_t) \\ &+ \gamma(\mathbf{e}_t + F_{t-1}\Delta\mathbf{x}_t + G_{t-1}\Delta\mathbf{u}_t)^T P(\mathbf{e}_t + F_{t-1}\Delta\mathbf{x}_t + G_{t-1}\Delta\mathbf{u}_t). \end{aligned} \tag{13}$$

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

$$\begin{aligned} \Delta\mathbf{u}_t &= -(R + \gamma G_{t-1}^T P G_{t-1})^{-1}[R\mathbf{u}_{t-1} + \gamma G_{t-1}^T P(\mathbf{e}_t + F_{t-1}\Delta\mathbf{x}_t)] \\ &= -(R + \gamma G_{t-1}^T P G_{t-1})^{-1}[R\mathbf{u}_{t-1} + \gamma G_{t-1}^T P\mathbf{e}_t + \gamma G_{t-1}^T P F_{t-1}\Delta\mathbf{x}_t]. \end{aligned} \tag{14}$$

From Eq. 14, we can conclude that the policy is in the form of system variables $(\mathbf{u}_{t-1}, \mathbf{e}_t, \Delta\mathbf{x}_t)$ feedback, and the gains are functions of the dynamics of the current linearized system (F_{t-1}, G_{t-1}).

2.3 Online incremental model identification

Assuming that the incremental model is identifiable using LS techniques with measurements of proper excitation and response, this paper adopts the Recursive Least Square (RLS) approach to online identify the system transition matrix $\mathbf{F}_{t-1}$ and the input distribution matrix $\mathbf{G}_{t-1}$ of the linearized model. The incremental form of the state in Eq. (6) can be rewritten row by row as follows:

$$\Delta x_{r,t+1} \approx \begin{bmatrix}\Delta\mathbf{x}_t^T & \Delta\mathbf{u}_t^T\end{bmatrix} \cdot \begin{bmatrix}\mathbf{f}_r^T \\ \mathbf{g}_r^T\end{bmatrix}, \tag{15}$$

where $\Delta x_{r,t+1} = x_{r,t+1} - x_{r,t}$ is the increment of rth state element, and $\mathbf{f}_r$ and $\mathbf{g}_r$ are the elements of rth row vector of $\mathbf{F}_{t-1}$ and $\mathbf{G}_{t-1}$. These parameters can be identified using the RLS usually row by row. Since they share the same covariance matrix, they can also be identified together as in the parameter matrix $\Theta_{t-1} = \begin{bmatrix}\mathbf{F}_{t-1}^T \\ \mathbf{G}_{t-1}^T\end{bmatrix} \in \mathscr{R}^{(n+m)\times n}$.

The state prediction equation can be written as follows:

$$\Delta\widehat{\mathbf{x}}_{t+1}^T = X_t^T \widehat{\Theta}_{t-1}, \tag{16}$$

where $X_t = \begin{bmatrix}\Delta\mathbf{x}_t \\ \Delta\mathbf{u}_t\end{bmatrix} \in \mathscr{R}^{(n+m)\times 1}$ stands for the input information of the incremental model. The RLS approach adopted in this paper is presented as follows [21]:

$$\varepsilon_t = \Delta\mathbf{x}_{t+1}^T - \Delta\widehat{\mathbf{x}}_{t+1}^T, \tag{17}$$

$$\widehat{\Theta}_t = \widehat{\Theta}_{t-1} + \frac{Cov_{t-1}X_t}{\gamma_{RLS} + X_t^T Cov_{t-1} X_t}\varepsilon_t, \tag{18}$$

$$Cov_t = \frac{1}{\gamma_{RLS}}\left(Cov_{t-1} - \frac{Cov_{t-1}X_t X_t^T Cov_{t-1}}{\gamma_{RLS} + X_t^T Cov_{t-1} X_t}\right), \tag{19}$$

where $\varepsilon_t \in \mathscr{R}^{1\times n}$ is the prediction error, also called *innovation*, $Cov_t \in \mathscr{R}^{(n+m)\times(n+m)}$ is the estimation covariance matrix, and γ_{RLS} is the forgetting factor for this RLS approach.

The RLS approach used in this paper possesses a significant advantage over the piecewise Ordinary Least Square (OLS) method: RLS has fewer issues with persistent excitation. The OLS method needs to do a matrix inversion at each update [16, 20]. If there is not enough excitation at that moment, the matrix might not be invertible, and the parameters cannot be identified. The RLS method, on the other hand, does not need to do matrix inversions because it uses the matrix inversion lemma to update the covariance matrix, which contains the information of that inverted matrix. Therefore, it can effectively identify parameters of the time-varying system and also keep the parameters stable when the excitation is not enough. This paper initializes the **F** matrix as an identity matrix and the **G** matrix as a zero matrix and chooses the forgetting factor γ_{RLS} to be 0.8.

If the nonlinear model is unknown, while the full state is measurable, iADP algorithm can be applied to improve the policy as follows:

iADP algorithm for reference tracking

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

$$\mathbf{e}_t^T P^{(j+1)} \mathbf{e}_t = \mathbf{e}_t^T Q \mathbf{e}_t + \mathbf{u}_t^T R \mathbf{u}_t + \gamma \mathbf{e}_{t+1}^T P^{(j)} \mathbf{e}_{t+1}. \tag{20}$$

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

$$\begin{aligned}\Delta\mathbf{u}_t = &-(R + \gamma G_{t-1}^T P^{(j+1)} G_{t-1})^{-1} [R\mathbf{u}_{t-1} \\ &+ \gamma G_{t-1}^T P^{(j+1)} \mathbf{e}_t + \gamma G_{t-1}^T P^{(j+1)} F_{t-1} \Delta\mathbf{x}_t].\end{aligned} \tag{21}$$

Opposite to 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.

3. Spacecraft with liquid sloshing

The liquid sloshing is one of the most undesired, unknown, and uncertain dynamics involved in the spacecraft attitude control. This phenomenon is caused, for example, by the liquid in the fuel tank sloshing around due to the forces and torques acting on the spacecraft. In turn, the sloshing fuel will interact with the fuel tank and thus perturb the spacecraft. This paper applies the iADP method to control a satellite disturbed by the liquid sloshing. The model in Fig. 1 is used as a validation of the proposed method.

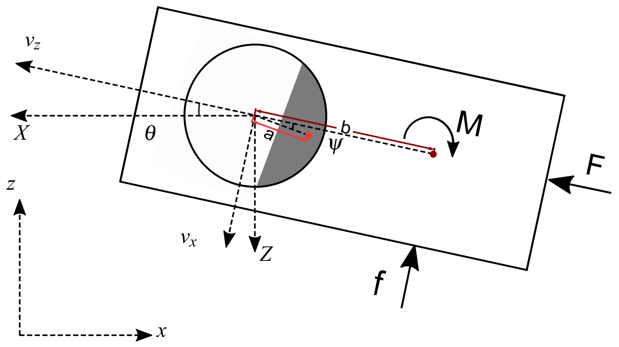

Fig. 1. A satellite model with pendulum dynamics modeling liquid sloshing.

As seen in Fig. 1, apart from the liquid the satellite can be seen as a rigid body, whose mass and moment of inertia are m and I. The liquid sloshing can be represented by a pendulum [1, 2, 3, 4], whose mass and moment of inertia are m_p and I_p, respectively. There are three control inputs: the force f and the pitch moment M are the satellite attitude control inputs, and the thrust F is constant.

Although the proposed method is model-free, the satellite model is included in this paper for the validation and reproduction. The satellite and analogous liquid sloshing dynamic and kinematic state equations are directly taken from [2, 3] and presented as follows:

$$\begin{aligned}&(m+m_p)(\dot{v}_x + v_z\dot{\theta}) + mb\dot{\theta} + m_p a(\ddot{\psi}+\ddot{\theta})sin(\psi) \\ &+m_p a(\dot{\psi}+\dot{\theta})^2 cos(\psi) = F,\end{aligned} \tag{22}$$

$$\begin{aligned}&(m+m_p)(\dot{v}_z - v_x\dot{\theta}) + mb\ddot{\theta} + m_p a(\ddot{\psi}+\ddot{\theta})cos(\psi) \\ &-m_p a(\dot{\psi}+\dot{\theta})^2 sin(\psi) = f,\end{aligned} \tag{23}$$

$$mb(\dot{v}_z - v_x\dot{\theta}) + (I_p + mb^2)\ddot{\theta} - \kappa\dot{\psi} = M + bf, \tag{24}$$

$$\begin{aligned}&(m_p a^2 + I_p)(\ddot{\psi}+\ddot{\theta}) \\ &+m_p a[(\dot{v}_x + v_z\dot{\theta})sin(\psi) + (\dot{v}_z - v_x\dot{\theta})cos(\psi)] + \kappa\dot{\psi} = 0,\end{aligned} \tag{25}$$

where the parameters used in simulations are $m = 600kg$, $I = 720kg/m^2$, $m_p = 100kg$, $I_p = 90kg/m^2$, $a = 0.3m$, $b = 0.3m$, $F = 500N$, and $\kappa = 0.19kgm^2/s$.

From Eqs. (22) and (23), the translational movement related variables can be isolated as follows:

$$\begin{aligned}&(\dot{v}_x + v_z\dot{\theta}) = \\ &\frac{F - mb\dot{\theta} - m_p a(\ddot{\psi}+\ddot{\theta})sin(\psi) - m_p a(\dot{\psi}+\dot{\theta})^2 cos(\psi)}{m+m_p},\end{aligned} \tag{26}$$

$$\begin{aligned}&(\dot{v}_z - v_x\dot{\theta}) = \\ &\frac{f - mb\ddot{\theta} - m_p a(\ddot{\psi}+\ddot{\theta})cos(\psi) + m_p a(\dot{\psi}+\dot{\theta})^2 sin(\psi)}{m+m_p}.\end{aligned} \tag{27}$$

The rotational variables can be separated from the translational variables by substituting Eqs. (26) and (27) into Eqs. (24) and (25):

$$\begin{aligned}&mb[f - mb\ddot{\theta} - m_p a(\ddot{\psi}+\ddot{\theta})cos(\psi) + m_p a(\dot{\psi}+\dot{\theta})^2 sin(\psi)] \\ &+(m+m_p)(I_p + mb^2)\ddot{\theta} - (m+m_p)\kappa\dot{\psi} \\ &= (m+m_p)(M+bf),\end{aligned} \tag{28}$$

$$\begin{aligned}&m_p a\{sin(\psi)[F - mb\dot{\theta} - m_p a(\ddot{\psi}+\ddot{\theta})sin(\psi) \\ &-m_p a(\dot{\psi}+\dot{\theta})^2 cos(\psi)] + cos(\psi)[f - mb\ddot{\theta} \\ &-m_p a(\ddot{\psi}+\ddot{\theta})cos(\psi) + m_p a(\dot{\psi}+\dot{\theta})^2 sin(\psi)]\} \\ &+(m+m_p)(m_p a^2 + I_p)(\ddot{\psi}+\ddot{\theta}) + (m+m_p)\kappa\dot{\psi} = 0.\end{aligned} \tag{29}$$

The rotational motion of the satellite and the pendulum do not contain any translational variables. Hence, the MIMO nonlinear model of the satellite attitude control problem has been generated for simulations.

4. Experiments and Results

The first part in this section explicates the implementation of the aforementioned algorithms and discusses some related issues, including the persistent excitation and the forgetting factor. The second part illustrates the results of using the iADP method. It first trains the policy off-line on the model using the before-mentioned parameters, and is then applied to a different model using online learning.

4.1 Implementation of iADP

In ADP methods, the forgetting factor γ represents the importance of the upcoming states in the future and is

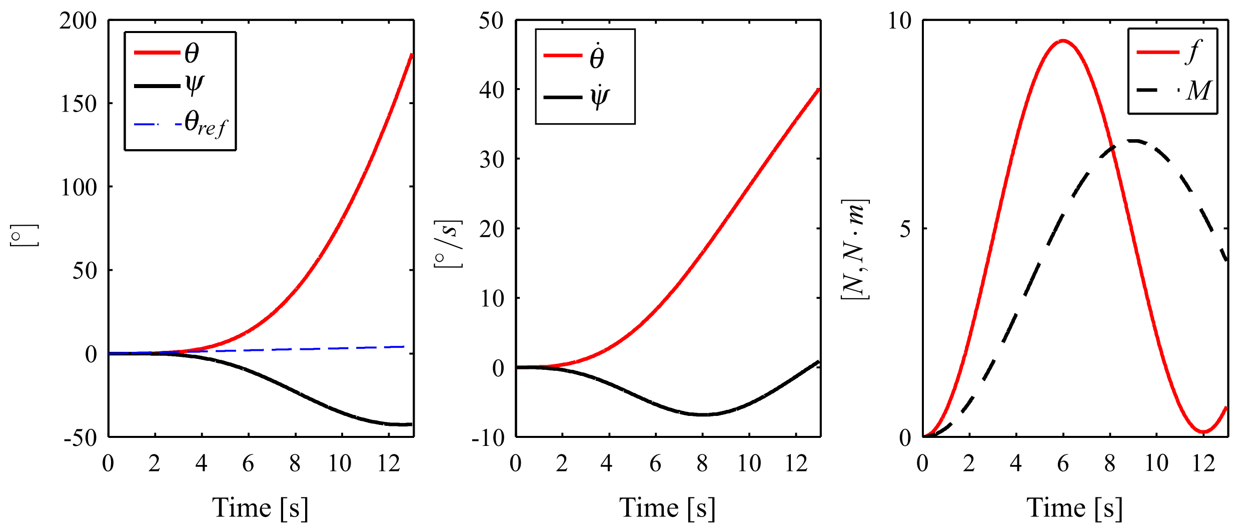

Fig. 2. The control performance using the initial policy with PE.

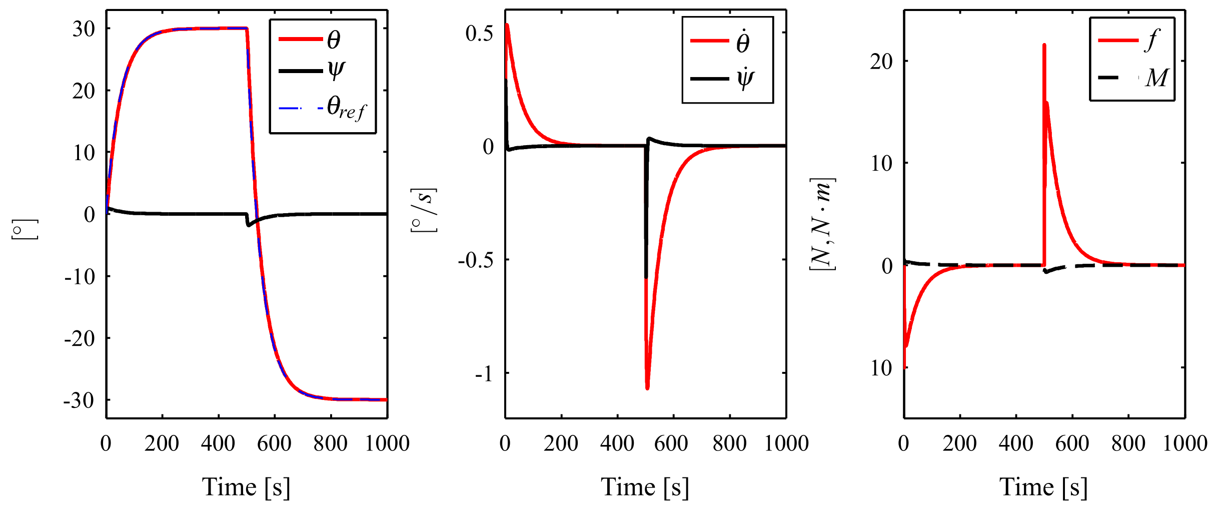

Fig. 3. The control performance using the trained policy (5th iteration) to track a filtered doublet signal.

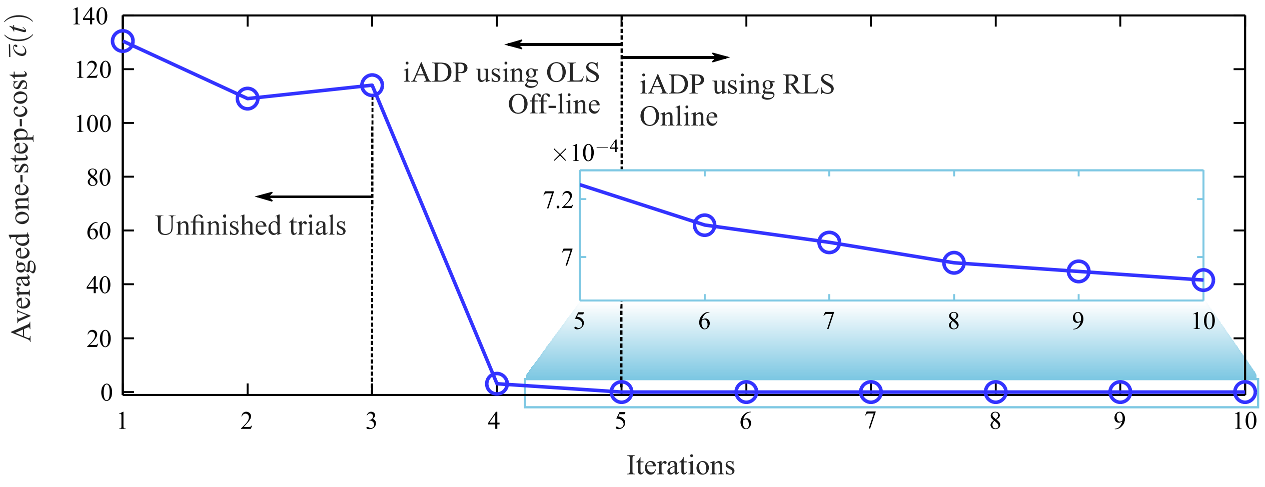

Fig. 4. The averaged one-step-cost during the training in tracking the filtered doublet signal.

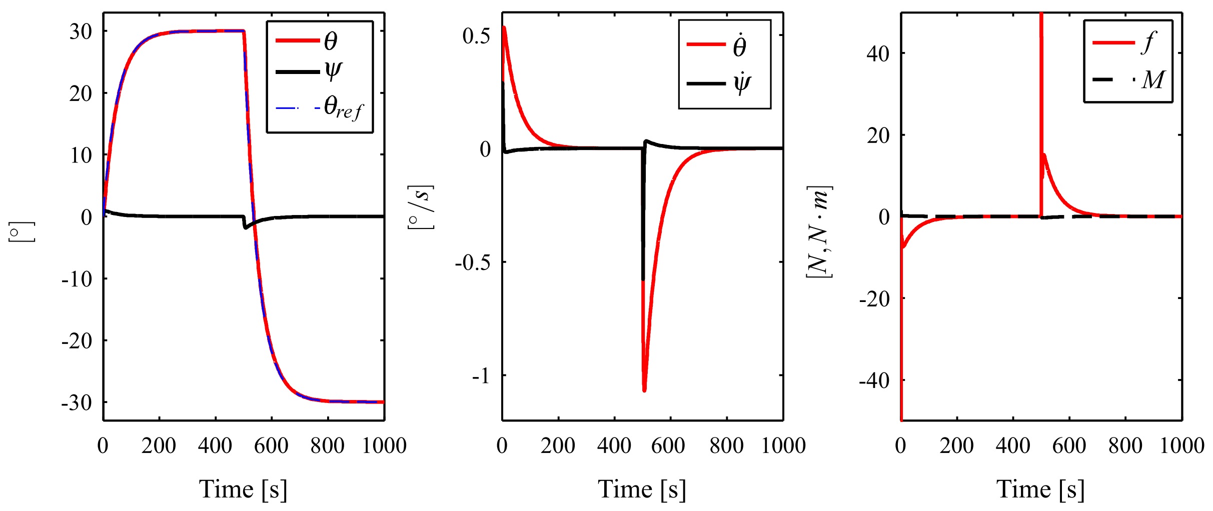

Fig. 5. The online control performance on a different model using the recursive iADP to track the filtered doublet signal.

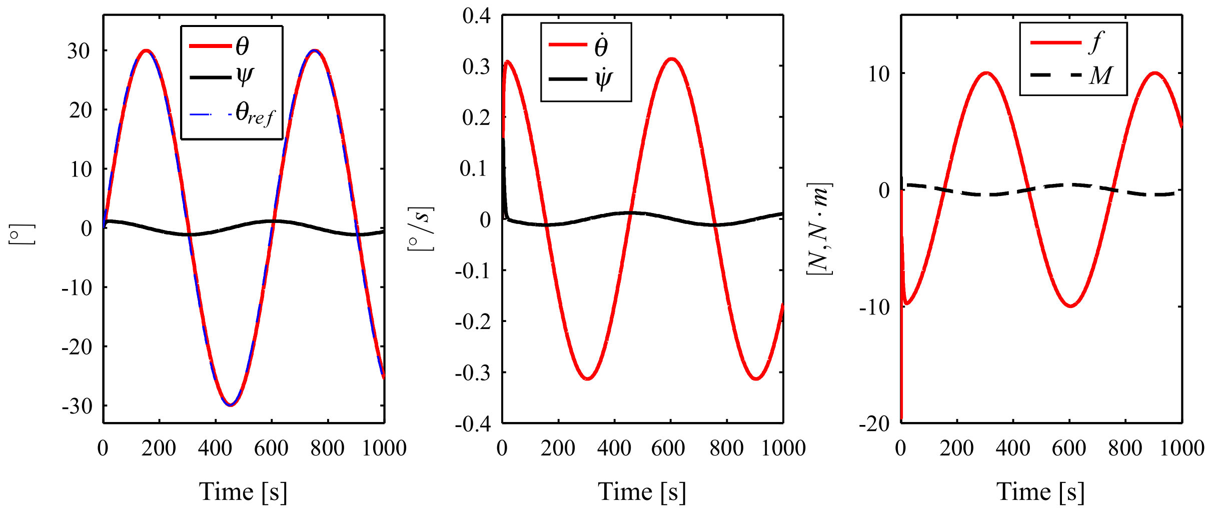

Fig. 6. The control performance using the trained policy (7th iteration) to track a sinusoidal signal.

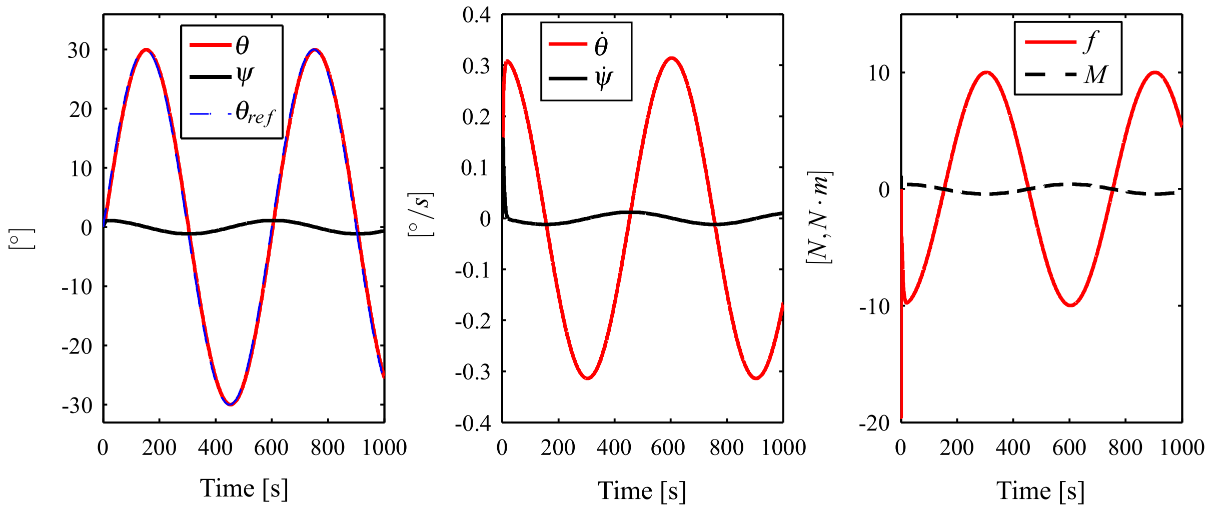

Fig. 7. The online control performance on a different model using the recursive iADP to track the sinusoidal signal.

usually chosen as $\gamma \in (0,1)$. Therefore, the infinite sum has a finite value as long as the cost sequence is bounded, and the agent is not "myopic" in being concerned only with maximizing immediate cost [5]. Compared to LADP methods, iADP methods prefer a smaller γ because the nonlinear system as well as the reference signal in tracking task has more uncertainties. In this paper, $\gamma = 0.5$. Note that the control performance is not very sensitive to γ as $\gamma = 0.2$ and $\gamma = 0.8$ have similar performance to the nominated value, but the magnitude of the value function may increase with a larger γ.

As with other ADP methods, good evaluation depends heavily on the exploration of the state space, which is represented by Persistent Excitation (PE). There are two main purposes of adding PE. The iADP method is designed based on the optimality principle, which is represented as the exploitation. Therefore, the first purpose of PE is to provide exploration so as to achieve a trade-off and to better evaluate the current policy. Second, although RLS depends less on PE, PE is still imperative for identifying the incremental model, especially when the discretized model is time-varying. This paper introduces an input disturbance, which is a sum of sinusoidal signals. This disturbance persistently excites the system for identification of the system and exploration of the state space. On the other hand, disturbances are usually undesirable inputs in the real world. Therefore, the control task is to track the reference signal as well as to reject the disturbance, which increases the difficulty of the task.

Although the rotational motion of the satellite model with liquid sloshing can be analogous to the pendulum dynamics as Eqs. (28) and (29) in previous section. The iADP model-free nonlinear method does not need any a priori knowledge about the model, but only assumes a general model, which can present any structure and parameters:

$$\dot{\mathbf{x}}(t) = f[\mathbf{x}(t), \mathbf{u}(t) + \mathbf{d}(t)], \tag{30}$$

where $\mathbf{x} = [\theta, \dot{\theta}, \psi, \dot{\psi}]^T$ is the state vector, $\mathbf{u} = [f, M]^T$ is the control input vector of the system since the thrust F is constant, and $\mathbf{d}$ is the input disturbance vector. The iADP algorithm for reference tracking in Eqs. (20) and (21) can be used to train the policy off-line. The kernel matrix P can be updated after each iteration using OLS method. The model information G_{t-1} and F_{t-1} are provided by the incremental model identified online using Eqs. (17) to (19).

However, when the trained policy P^j can track the reference very accurately and the input disturbance are rejected, the system does not have much exploration and the parameters in P^{j+1} might not be identified using OLS. Therefore, when the averaged one-step-cost in an iteration j is smaller then a threshold $\bar{c}(t)^j < c_{threshold}$, the kernel matrix P will be updated using RLS with forgetting factor $\gamma_{RLS} = 1$. The iADP algorithm in Eqs. (20) and (21) will be rewritten using the RLS as follows:

Recursive iADP algorithm for reference tracking

Evaluation. The cost function kernel matrix P within iteration $j+1$ can be evaluated and updated recursively for each time step t:

$$\mathbf{e}_t^T P_t \mathbf{e}_t = \mathbf{e}_t^T Q \mathbf{e}_t + \mathbf{u}_t^T R \mathbf{u}_t + \gamma \mathbf{e}_{t+1}^T P_{t-1} \mathbf{e}_{t+1}. \tag{31}$$

Policy improvement. The policy improves for the new kernel matrix P_t:

$$\Delta\mathbf{u}_t = -(R + \gamma G_{t-1}^T P_t G_{t-1})^{-1} [R\mathbf{u}_{t-1} + \gamma G_{t-1}^T P_t \mathbf{e}_t + \gamma G_{t-1}^T P_t F_{t-1} \Delta\mathbf{x}_t]. \tag{32}$$

In each iteration $j+1$, the initial kernel matrix P_0 is the last updated $P^{(j)}$ from previous iteration. This recursive algorithm will also be used when the off-line trained policy are applied to the real system online.

4.2 Results and Discussion

The proposed iADP method is demonstrated by a simulation experiment on satellite attitude control problems disturbed by liquid sloshing. The full states (θ, ψ, $\dot{\theta}$, and $\dot{\psi}$) and the control input (f, M) are measurable. The control target is to track a reference signal of θ while stabilizing ψ, which often happens in the initial attitude acquisition.

In the off-line learning stage, the termination of each iteration can happen 1) when it finish the whole trial and reaches the time limit 1000 seconds or 2) when any of the states reaches their limit: 180° for θ and ψ and $50^\circ/s$ for $\dot{\theta}$ and $\dot{\psi}$. The initial kernel matrix P^0 is a zero matrix. Therefore, the first iteration is an open loop system with sinusoidal PE, as shown in Fig. 2. This iteration stops at around $13s$ when θ reaches 180°.

The first reference is a filtered doublet signal. Figure 3 illustrate the control performance of the trained policy, which can track the reference of θ, stabilize ψ, and reject the input disturbance. The averaged one-step-cost after 5th iteration is below the threshold as shown in Fig. 4. The adaptation of the policy uses the RLS approach afterward. The first 3 iterations do not finish the whole trial and stop because they reach the state limit.

In practice, the really system parameters are often different from the model that is used to train the policy. The trained policy using the model in previous section will be applied to a different model with changed parameters: $m = 1200kg$, $I = 900kg/m^2$, $m_p = 50kg$, $I_p = 80kg/m^2$, $a = 0.2m$, $b = 0.5m$, $F = 600N$, and $\kappa = 0.19kgm^2/s$. As is visible in Fig. 5, the recursive iADP algorithm can be applied to the different model starting with the trained policy without loss of accuracy. The main reason is that iADP method separate the value function with an approximation and the model information with the online identified incremental model. When the system changes, the value function does not change a lot, while the incremental model changes immediately using the RLS approach.

The second reference is a sinusoidal signal. Figure 6 shows that the off-line trained policy can track the dynamical reference of θ, minimize ψ, and also reject the input disturbance. The policy applied to the original model converges from the $7th$ iteration. This trained policy is applied to the before-mentioned different model using recursive iADP online. Figure 7 depicts the online adaptive control performance of the proposed method.

5. **Conclusion**

This paper proposes an adaptive nonlinear control method, called incremental Approximate Dynamic Programming (iADP). This method systematically applies a quadratic function to approximate the cost-to-go and greatly simplifies the design process of ADP methods. In addition, this method uses an incremental model to deal with the nonlinearity and uncertainty of the system. It combines the advantages of LADP and incremental approaches to generate a model-free, adaptive controller for nonlinear systems. This method does not need any a priori knowledge of the system dynamics, online identification of the global model nor even an assumption of the time scale separation, but only an online identified incremental model.

This study expands the iADP method to tracking problems for MIMO nonlinear systems with two algorithms. The first one is an off-line learning algorithm, which uses OLS to update the policy after each iteration. After the averaged one-step-cost is below the threshold, the second algorithm will be applied online. This online recursive iADP algorithm can also be used in a real system, which is mismatched from the original model and is not easy to reset after failure. These algorithms are applied to an attitude control problem of a simulated satellite disturbed by liquid sloshing. The results show that the off-line trained policy rejects the disturbance and tracks reference signals accurately. When the trained policy is applied to a different system, the control performance remains at the same level of accuracy, and online learning using the recursive iADP ensures the adaptability of the proposed method.

Acknowledgements

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

References

[1] E. de Weerdt, Erik-Jan van Kampen, D. van Gemert, , Qi Ping Chu, and J.A. Mulder. Adaptive nonlinear dynamic inversion for spacecraft attitude control with fuel sloshing. In *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2008.

[2] Luiz Carlos Gadelha de Souza and Alain G de Souza. Satellite attitude control system design considering the fuel slosh dynamics. *Shock and Vibration*, 2014, 2014.

[3] Honghua Zhang and Zeguo Wang. Attitude control and sloshing suppression for liquid-filled spacecraft in the presence of sinusoidal disturbance. *Journal of Sound and Vibration*, 383:64–75, 2016.

[4] Paul AC Mason and Scott R Starin. Propellant slosh analysis for the solar dynamics observatory. *NASA technical report*, 2005.

[5] Richard S Sutton and Andrew G Barto. *Introduction to reinforcement learning*. MIT Press, 1998.

[6] Richard Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[7] Warren B Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality*, volume 703. John Wiley & Sons, 2007.

[8] 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.

[9] Arezou Keshavarz and Stephen Boyd. Quadratic approximate dynamic programming for input-affine systems. *International Journal of Robust and Nonlinear Control*, 24(3):432–449, 2014.

[10] S. Sieberling, Q. P. Chu, and J. A. Mulder. Robust flight control using incremental nonlinear dynamic inversion and angular acceleration prediction. *Journal of guidance, control, and dynamics*, 33(6):1732–1742, 2010.

[11] Paul Acquatella B., Wouter Falkena, Erik-Jan van Kampen, and Qi Ping Chu. Robust nonlinear spacecraft attitude control using incremental nonlinear dynamic inversion. In *AIAA Guidance, Navigation, and Control Conference*, 2012.

[12] P. Simplício, M. D. Pavel, E. van Kampen, and Q. P. Chu. An acceleration measurements-based approach for helicopter nonlinear flight control using incremental nonlinear dynamic inversion. *Control Engineering Practice*, 21(8):1065–1077, 2013.

[13] Peng Lu, ErikJan van Kampen, Cornelis de Visser, and Qi Ping Chu. Aircraft fault-tolerant trajectory control using incremental nonlinear dynamic inversion. *Control Engineering Practice*, 57:126–141, 2016.

[14] Paul Acquatella B., Erik-Jan van Kampen, and Qi Ping Chu. Incremental backstepping for robust nonlinear flight control. In *Proceedings of the EuroGNC 2013*, 2013.

[15] Y. Zhou, E. van Kampen, and Qi Ping Chu. Incremental approximate dynamic programming for nonlinear flight control design. In *Proceedings of the EuroGNC 2015*, 2015.

[16] Ye Zhou, E. van Kampen, and Q. P. Chu. Nonlinear adaptive flight control using incremental approximate dynamic programming and output feedback. *Journal of Guidance, Navigation and Dynamics*, 40:493–500, 2017. doi: http://arc.aiaa.org/doi/abs/10.2514/1.G001762.

[17] Jennie Si and Yu-Tsung Wang. Online learning control by association and reinforcement. *IEEE Transactions on Neural Networks*, 12(2):264–276, 2001.

[18] Russell Enns and Jennie Si. Helicopter trimming and tracking control using direct neural dynamic programming. *Neural Networks, IEEE Transactions on*, 14(4):929–939, 2003.

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

[20] Y Zhou, E van Kampen, and Q P Chu. Incremental model based heuristic dynamic programming for nonlinear adaptive flight control. In *Proceedings of the International Micro Air Vehicles Conference and Competition 2016, Beijing, China*, 2016.

[21] Simon Haykin. *Adaptive filter theory*. Prentice Hall, 2002.