Proceedings of the EuroGNC 2013, 2nd CEAS Specialist Conference on
Guidance, Navigation & Control, Delft University of Technology, Delft,
The Netherlands, April 10-12, 2013

Adaptive Trajectory Controller for Generic Fixed-Wing Unmanned Aircraft

Maximilian Mühlegg, Johann C. Dauer, Jörg Dittrich and Florian Holzapfel

Abstract This work deals with the construction of a nonlinear adaptive trajectory controller, which is easily applicable to a multitude of fixed wing unmanned aircraft. Given a common signal interface, the adaptive trajectory controller is divided into a generic part, which is common for each vehicle, and into a part, which is unique. The generic part of the control architecture bases on a common inversion model which is used for feedback linearization. However, the dynamics of the aircraft and the inversion model differ, thus introducing model uncertainties to the feedback linearized system. The effect of modeling uncertainties is reduced by the application of a concurrent learning model reference adaptive controller, which uses neural networks in order to approximate the uncertainty. Leveraging instantaneous as well as stored data concurrently for adaptation ensures convergence of the adaptive parameters to a set of optimal weights, which minimize the approximation error. Performance and robustness against certain model uncertainties is shown through numerical simulation for two significantly different unmanned aircraft.

Dipl.-Ing. Maximilian Mühlegg
Institute of Flight System Dynamics, Technische Universität München, Boltzmannstrae 15, 85748 Garching bei München, Germany, e-mail: maximilian.muehlegg@tum.de

Dipl.-Ing. Johann C. Dauer
Institute of Flight Systems, German Aerospace Center, Lilienthalplatz 7, 38108 Braunschweig, e-mail: johann.dauer@dlr.de

MSc. Jörg Dittrich
Institute of Flight Systems, German Aerospace Center, Lilienthalplatz 7, 38108 Braunschweig, e-mail: joerg.dittrich@dlr.de

Prof. Dr.-Ing. Florian Holzapfel
Institute of Flight System Dynamics, Technische Universität München, Boltzmannstrae 15, 85748 Garching bei München, Germany, e-mail: florian.holzapfel@tum.de

1 Introduction

Unmanned Aircraft (UA) have attracted a high amount of attention in recent years. Research institutes encounter a lot of challenging tasks, while handling an increasing number of aerial vehicles. In order to perform flight tests in different research areas, reliable flight control systems need to be available for each aircraft. In UA projects a considerable amount of time and effort is spent creating reliable control architectures. In an attempt to decrease the effort for the controller design the Institute of Flight Systems at the German Aerospace Center (DLR) analyzed in collaboration with the Institute of Flight System Dynamics (FSD) at the Technische Universität München, to which extent such a control architecture can be generalized. As a basis for this the two significantly different aircraft Prometheus (DLR) and ExtremeStar (FSD) serve as testing platforms. The goal of this paper is to develop a flight control architecture, which can easily be applied to different fixed-wing aircraft at both institutes, thus aiming to decrease the time and effort needed in order to set up a flight control system. The underlying assumption in this connection is that the fixed wing UA share the same dynamical structure.

In classical linear control theory the parameters of the aircraft and controller gains satisfying desired performance and robustness requirements have to be found for a set of trimming points throughout the flight envelope. While this approach has shown to be reliable to control an aircraft, parameter identification and the selection of gains has to be performed for each UA individually ([6, 22]). Furthermore, the received database might become unreliable once the configuration of the aircraft changes, further increasing the amount of time and effort which needs to be put into the design of the flight control system. This disadvantage especially applies to (fixed wing) UA, since they operate in a huge range of altitude and velocity.

Using nonlinear control methods, trimming points and their blending can be avoided. One such technique is feedback linearization, also called dynamic inversion. The idea is to transform the nonlinear system into an equivalent linear form ([11]). For the resulting system linear control methods such as a linear tracking control design can be applied. The benefit compared to the classic linearization is that the transformed system is valid throughout a wider part of the flight envelope and not limited to the close vicinity of selected trimming points. However, feedback linearization requires accurate knowledge of the dynamics and parameters of the nonlinear system. In general, nonlinear systems are of infinite order and only estimates of the real parameters are available. Therefore, feedback linearization can only be performed with respect to a model of the nonlinear dynamical system, inevitably introducing uncertainties to the system.

Model Reference Adaptive Control (MRAC) is concerned with reducing the impact of such modeling uncertainties ([1, 18]). In the framework of a feedback linearized system, the idea is to make the real dynamics behave like the model chosen for the inversion. If the structure of the uncertainty is known, it can be linearly parameterized by a weighted combination of its (known) basis ([1, 18, 26]). If the structure is unknown universal approximators can be employed to reduce the impact of the uncertainty ([4, 11, 13, 14, 19, 23]). Therefore a-priori chosen regressor func-

tions are weighted by a set of adaptive parameters, which are updated based on the instantaneous tracking error. The underlying assumption is that there exists an ideal set of weights which result in the approximation with the smallest error. In classical adaptive control the convergence of the adaptive parameters to their optimal values is only achieved if the regressor vector is persistently excited (PE) ([26]). To ensure PE on the regressor vector is often operationally undesirable and might not even be possible.

Concurrent learning uses instantaneous information concurrently with specifically stored data for future updates of the parameters ([5, 7]): The idea is, that if data points are stored at a time when the regressor vector was excited, this information can be used for future updates to ensure parameter convergence. For this purpose the stored data only has to meet a requirement of linearly independency on the stored data, the regressor vector is not required to be PE.

In order to construct a trajectory controller based on feedback linearization and concurrent learning MRAC for a multitude of UA, a common signal interface needs to be defined. Especially, the states chosen for the feedback linearization approach need to be available through a suitable set of sensors and sensor fusion algorithms. The adaptive controller is then used to drastically reduce the effects of the uncertainty resulting from a deviation between the plant dynamics and the inversion model. In fact, the concurrent learning adaptive controller ensures that the same inversion model can be used for a multitude of fixed wing UA. In order to show that the control architecture is applicable to a multitude of UA and in order to show robustness against uncertainties, the controller is tested in numerical simulation for two significantly different fixed wing UA, namely Prometheus (DLR) and ExtremeStar (FSD).

In this paper, for a given vector a, $(a)_*$ denotes the frame in which a is notated and a_* denotes the physical type of a. For example $(V_K)_B$ denotes the kinematic velocity notated in the body-fixed frame. Furthermore, (ω^{xy}) describes the rotational rates of the y system relative to the x system. All signals are notated with respect to the center of gravity if not stated otherwise. The outline of this paper is as follows: In Section 2 the control strategy based on feedback linearization and an adaptive element is described. In Section 3 we present the results from numerical simulation in a Software- and Hardware-in-the-Loop framework. The paper is concluded in Section 4.

2 Control Architecture

The concept of the proposed control strategy is presented in Figure 1. The goal of the paper is to define a control architecture which is simply applicable to a variety of fixed wing UA. The underlying assumption is that multiple fixed wing UA share the same dynamic structure. However, it is not yet possible to construct an inversion controller which is simply applicable to every aircraft. Instead, the proposed non-

linear trajectory controller is divided into two parts: A generic inversion controller, including reference models, and the control allocation.

The generic inversion controller is independent of the individual fixed wing UA and is constructed based on a shared inversion model. Still, the parameters in this block, in particular proportional and integral controller gains as well as the parameter of the reference model dynamics, have to be adjusted in order to suit each individual platform. Another part of this block is the concurrent learning adaptive controller, which minimizes the deviation between the plant dynamics and the chosen inversion model.

In order for this block to be valid for several platforms, a common signal interface has to be defined. This interface has to meet two requirements. On the one hand, the model of the plant has to be feedback linearizable with the chosen set of states provided by the interface. On the other hand, the chosen signals have to be measurable or at least computable from measurable data by the application of appropriate sensor fusion algorithms.

Finally, the control allocation maps the desired outputs of the generic inversion controller onto the control devices of the aircraft. Since the number and type of control devices can vary strongly between UA, this part has to be constructed individually for each aircraft.

The outline of this section is as follows: Section 2.1 defines a common signal interface for the proposed generic inversion controller. In section 2.2 dynamic inversion is performed with respect to a selected inversion model. Section 2.3 introduces the adaptive component which reduces the effects of modeling uncertainties. Finally, Section 2.4 briefly discusses the control allocation.

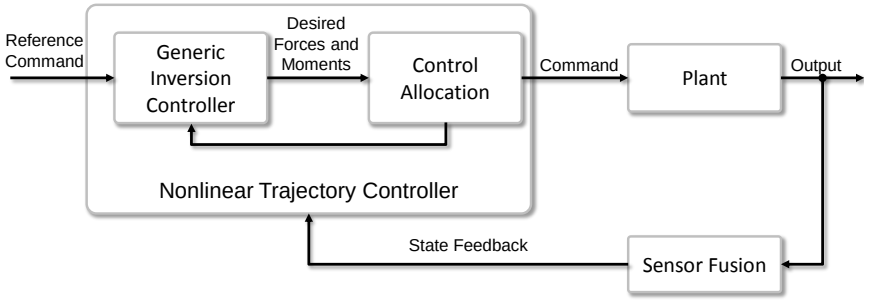


Fig. 1 General control strategy

2.1 Signal Interface Definition

Unmanned aircraft can be equipped with a huge variety of sensors each relaying on different information about the states of an aircraft. Furthermore, it is possible

to perform feedback linearization of a dynamic system relying on different sets of states, the attitude dynamics can, for example, be inverted by using either Euler angles or kinematic attitude angles. Table 1 gives a summary of the specific signal interface definitions used in this work. All signals which are required as an input are labeled with 'in'. The commands generated by the inversion controller are labeled with 'out'. A feasible set of sensors which ensures that the required signals in Table 1 can be obtained and the proposed control architecture can be used is given by a pitot tube, an Inertial Measurement Unit, a Magnetometer and a GPS module. Furthermore, the body-fixed moments and the force which acts in the direction of the velocity vector need to be estimated. For this purpose the chosen inversion model can be used together with a model of the actuators ([11]).

Character	Context	Symbol	Unit	Explanation
Reference Command (In)	Translation	$V_{K,CMD}$	$[\frac{m}{s}]$	Kinematic velocity
		$\gamma_{K,CMD}$	$[rad]$	Flight-path azimuth angle
		$\chi_{K,CMD}$	$[rad]$	Flight-path inclination angle
Sensor Fusion (In)	Position	h	$[m]$	Altitude
	Translation	$\begin{pmatrix} u \\ v \\ w \end{pmatrix}_B$	$[\frac{m}{s}]$	Body fixed velocity vector
	Attitude	$\begin{pmatrix} \Phi \\ \Theta \\ \Psi \end{pmatrix}$	$[rad]$	Euler angles
	Rotation	$\begin{pmatrix} p \\ q \\ r \end{pmatrix}_B$	$[\frac{rad}{s}]$	Body-fixed rotational rates
	Air Data	$\bar{q}$	$[\frac{N}{m^2}]$	Dynamic pressure
Estimated quantities (In)	Force	$(F_x)_K$	$[N]$	Estimated force in kinematic x-direction
	Moment	$\begin{pmatrix} L \\ M \\ N \end{pmatrix}_B$	$[Nm]$	Estimated moments
Control variable (Out)	Force	$(F_{x,CMD})_K$	$[N]$	Force in kinematic x-direction
	Moment	$\begin{pmatrix} L_{CMD} \\ M_{CMD} \\ N_{CMD} \end{pmatrix}_B$	$[Nm]$	Commanded moments

Table 1 Interface definitions for the proposed generic inversion controller

2.2 Approximate Model Inversion for Fixed Wing Aircraft

This section discusses the fundamentals of approximate model inversion and how this technique can be used to invert the dynamics of a fixed wing UA. Let $x(t) \in \mathbb{R}^n$ be the known state vector and let $\delta(t) \in \mathbb{R}^m$ denote the control input.

The general nonlinear dynamics of the aircraft can be written as

$$\dot{x}(t) = f(x(t), \delta(t)), \tag{1}$$

where the function f is assumed to be unknown yet sufficiently smooth. That is, its partial derivatives up to the required order are defined and continuous. Additionally, the control input $\delta(t)$ is assumed to be bounded and piecewise continuous. Since the exact model in (1) is usually neither available nor invertible, we introduce an approximate inversion model $\hat{f}(x(t), \delta(t))$. The inversion model $\hat{f}$ needs to be continuous and invertible with respect to $\delta(t)$. Given a pseudo-control input $\nu(t) = \hat{f}(x(t), \delta(t))$ these requirements need to be fulfilled in order to be able to find a control command $\delta(t)$ by dynamic inversion such that:

$$\delta(t) = \hat{f}^{-1}(x(t), \nu(t)) \tag{2}$$

This approximation results in a model error $\Delta \in \mathbb{R}^n$, which can be formulated in additive form:

$$\dot{x}(t) = \nu(x(t), \delta(t)) + \Delta(x(t), \delta(t)) \tag{3}$$

The feedback linearization approach used in this work is based on the dynamics of a fixed wing aircraft, which are not derived in detail here but can be found among others in [3]. The validity of the differential equations is connected to a set of assumptions. The earth is assumed to be flat and non-rotating. Hence, the transport rate and the angular velocity of the earth are neglected. These assumptions are valid, since the presented UA only fly short distances and operate for a limited time only. The aircraft is seen to be a rigid body, the relative motion of aircraft mass elements is thus considered to be zero. Furthermore, the mass as well as mass distribution are considered to be quasi stationary. For the inversion model the atmosphere is assumed to be static, that is, there is no wind. As a result the intermediate kinematic frame as well as the kinematic attitude angles equal the aerodynamic frame and the aerodynamic attitude angles. Furthermore, the kinematic attitude angles can be used in order to invert the attitude dynamics. However, in reality the atmosphere is not static. The forces and moments resulting from wind are therefore considered as disturbances.

Figure 2 shows the general concept for the feedback linearization approach. The system is cascaded along the dynamical chain of the aircraft, each loop having a relative degree of one. For each loop a separate reference model and tracking controller is constructed. The advantage of such a cascaded system is that the resulting analytical terms are easy to handle. A drawback of the cascaded approach is that the bandwidth of the overall system is reduced compared to an inversion with relative degree three. However, it is argued in [11] that in the second case signals can be required, which are heavily corrupted by measurement noise, thus favoring a cascaded approach. In the following the feedback linearization approach based on [11] for a fixed wing UA is depicted.

The actuator dynamics are neglected in the proposed inversion model. Since the input exhibits its own dynamics and limitations, an additional uncertainty is added

to the system. Especially if the inverted system is combined with an adaptive element, actuator saturation can cause instability of the closed loop system through unbounded parameter growth during saturation. Pseudo Control Hedging (PCH) is a method which slows down the reference model dynamics by a measure of the expected reaction deficit of the plant, thus hiding actuator dynamics from the error dynamics and allowing adaptation even in the presence of saturation ([13]). Note, that from any loop the next inner loop can be viewed as a kind of actuator dynamics. Hence, PCH can be added separately to each loop and allows the simultaneous application of an adaptive element and artificial saturations in inner loops.

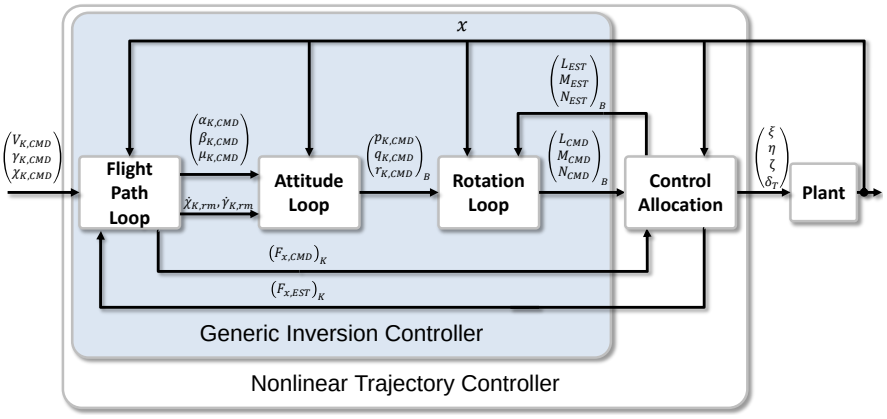


Fig. 2 Cascaded structure of the feedback linearization approach for the generic inversion controller

2.2.1 Inversion of the Path Dynamics

The outer loop is concerned with finding the proper kinematic attitude angles and thrust of the aircraft in order to follow a desired flight path. A nonzero angle of sideslip hereby leads to increased drag and an increase in the required thrust, reducing the efficiency of the flight. In order to preserve an aerodynamically efficient flight, the kinematic angle of sideslip β_K is commanded to be zero. Basis for the inversion of the path dynamics are the following equations of motion derived with Newtons Second Law:

$$\begin{aligned} \dot{V}_K &= \frac{T-D}{m} - gsin(\gamma_K) \\ \dot{\chi}_K &= \frac{Lsin\mu_K}{mV_K cos(\gamma_K)} \\ \dot{\gamma}_K &= \frac{Lcos\mu_K}{mV_K} - \frac{g}{V_K} cos(\gamma_K) \end{aligned} \tag{4}$$

Here T, D and L denote the thrust, the drag and the lift respectively. In the equations above it is assumed that the thrust only acts in the direction of the velocity

vector. The path dynamics can be inverted by applying the dynamic inversion concept to equation (4). However, Holzapfel argues in [11] that especially if the input of a dynamical system is described by an irrational function, a pure mathematical inversion can result in a multitude of solutions, which are often physically meaningless. Alternatively, a physically and geometrically motivated approach is used. For feedback linearization the forces in a plane perpendicular to the current velocity vector are considered. For the inversion model, assume that the lift coefficient C_L is only dependent on the zero-lift derivative $C_{L,0}$ and the derivative resulting from a change in the angle of attack $C_{L,\alpha}$. With $C_L = C_{L,0} + C_{L,\alpha}\alpha_K$ and $L = \frac{1}{2}\rho V_K^2 S C_L$ the linearizing feedback for the kinematic attitude angles results in

$$\begin{aligned} \alpha_{K,CMD} &= \frac{2L_{CMD}}{\rho V_K^2 S C_{L,\alpha}} - \frac{C_{L,0}}{C_{L,\alpha}} \\ \beta_{K,CMD} &= 0 \\ \mu_{K,CMD} &= arctan\left(\frac{\nu_\chi}{\nu_\gamma + \frac{g}{V_K} cos\gamma_K}\right), \end{aligned} \tag{5}$$

where $\nu_{OL} = [\nu_V, \nu_\gamma, \nu_\chi]^T$ denotes the pseudo control variables in the outer loop and $L_{CMD} = mV_K\sqrt{(\nu_\gamma + \frac{g}{V_K} cos(\gamma_K))^2 + \nu_\chi^2}$. The thrust equation is inverted by

$$T_{CMD} = F_{x,CMD} = m(\nu_V + gsin(\gamma_K)) + D. \tag{6}$$

For each state of the path dynamics a separate first order reference model can be constructed. A linear feedback with proportional and integral components is used in order to construct the pseudo control variable ν_{OL}. Finally, PCH is used in order to slow down the reference model dynamics of the path states by a measure of the reaction deficit of the plant.

2.2.2 Inversion of the Attitude Dynamics

The middle loop is concerned with finding the angular rates in order to realize the desired kinematic attitude angles α_K and μ_K as well as keeping the angle of sideslip β_K equal to zero in order to preserve an aerodynamically efficient flight. The basis for the inversion of the attitude dynamics is a purely kinematic relation between the angular rates and the kinematic attitude angles. The equations of motion can be derived from the strapdown equation ([25]):

$$\begin{pmatrix} \dot{\alpha}_K \\ \dot{\beta}_K \\ \dot{\mu}_K \end{pmatrix} = M\left[M_{KB}(\omega_K^{OB})_B - (\omega_K^{OK})_K\right], \tag{7}$$

where $M_{KB} = M_{BK}^T$ denotes the transformation matrix between the body-fixed and the kinematic frame and

$$M = \begin{bmatrix} 0 & \frac{cos(\mu_K)}{cos(\beta_K)} & \frac{sin(\mu_K)}{cos(\beta_K)} \\ 0 & sin(\mu_K) & -cos(\beta_K) \\ 1 & -tan(\beta_K)cos(\mu_K) & tan(\beta_K)sin(\mu_K) \end{bmatrix}. \tag{8}$$

The linearizing feedback for the attitude dynamics is given by

$$(\omega_{K,CMD}^{OK})_B = M_{BK}[(\omega_K^{OK})_K + M^{-1}\nu_{ML}], \tag{9}$$

where ν_{ML} denotes the pseudo control variables for the middle loop and

$$(\omega_K^{OK})_K = \begin{pmatrix} -\dot{\chi}_K sin(\gamma_K) \\ \dot{\gamma}_K \\ \dot{\chi}_K cos(\gamma_K) \end{pmatrix}. \tag{10}$$

The time derivatives $\dot{\gamma}_K$ and $\dot{\chi}_K$ are not measurable. Instead they can be computed from the load factors, which in turn are dependent on the accelerations and are therefore highly susceptible to measurement noise. As an alternative the states of the reference model from the outer loop $\dot{\gamma}_{K,rm}$ and $\dot{\chi}_{K,rm}$ can be used to construct the linearizing feedback. The pseudo-control input ν_{ML} is constructed using a proportional feedback controller. If the attitude dynamics are to be analyzed separately, an integrator can be added to the linear error controller in order to ensure steady state accuracy.

2.2.3 Inversion of the Rotational Dynamics

The basis for the inversion of the rotational dynamics are the equations of angular momentum. They are derived applying the law of conservation of angular momentum with respect to the center of gravity:

$$(\dot{\omega}_K^{OB})_B^B = I_{BB}^{-1}\{(M)_B - (\omega_K^{OB})_B \times \left[I_{BB}(\omega_K^{OB})_B\right]\} \tag{11}$$

Here $(M)_B$ denotes the moments about the aircraft and I_{BB} denotes the mass moment of inertia. The linearizing feedback for the inner loop is given with

$$M_{CMD})_B = (\omega_K^{OB})_B \times \left[I_{BB}(\omega_K^{OB}))_B\right] + I_{BB}\nu_{IL}, \tag{12}$$

where ν_{IL} denotes the pseudo control variable for the inner loop. For each angular rate a separate reference model is constructed. For the pseudo-control ν_{IL} a proportional feedback is used. Finally, PCH is applied to the inner loop, thus allowing the addition of an adaptive element in the presence of actuator dynamics.

2.3 Adaptive Controller

Feedback linearization is concerned with transforming a nonlinear system such that it exhibits linear input-output behavior. A major drawback of this approach is that this method is highly susceptible to parameter errors and unmodeled dynamics. In order to decrease the impact of model uncertainties an adaptive element is added to the feedback linearized system. Model Reference Adaptive Control for feedback linearized systems aims to make the uncertain system behave like the underlying inversion model. MRAC achieves this goal by utilizing a model of the uncertainty. If the uncertainty can be parameterized linearly a weighted combination of the basis of the uncertainty can be employed as an adaptive element ([2, 15, 17]). The assumption on structural knowledge of the uncertainty can be relaxed by requiring it to be at least continuous and defined over a compact domain. In this case neural networks have been repeatedly used as adaptive elements ([4, 11, 13, 14, 19, 23]). Instead of the basis of the uncertainty, neuro-adaptive control utilizes a set of chosen basis functions. In particular, a Gaussian Network with radial basis functions ([21]) is employed in this work. However, also other activation functions such as Sigmoids or B-splines are imaginable ([11, 24]). The output of the adaptive element $\nu_{ad} \in \mathbb{R}^m$ is given by

$$\nu_{ad} = W^T(t)\sigma(x(t)). \tag{13}$$

Here $W(t) \in \mathbb{R}^{(n_2+1)\times m}$ denotes the adaptive weights and $\sigma(x(t)) \in \mathbb{R}^{(n_2+1)}$ denotes the regressor vector containing n_2 radial basis functions and a constant bias. According to the universal approximation property of Radial Basis Function Neural Networks ([20]) we have, that given a fixed number of radial basis functions n_2, there exist ideal weights $W^* \in \mathbb{R}^{(n_2+1)\times m}$ and a vector $\varepsilon \in \mathbb{R}^m$ such that given a compact domain $D \subset \mathbb{R}^n$ the following approximation holds for all $x \in D$:

$$\Delta(x,\delta) = {W^*}^T\sigma(x) + \tilde{\varepsilon}(x) \tag{14}$$

The functional approximation error $\bar{\varepsilon} = sup_{x\in D}\|\tilde{\varepsilon}(x)\|$ can be made arbitrarily small by increasing the number of radial basis functions ([20]). Define the tracking error as $e(t) = x(t) - x_{rm}(t)$, where $x_{rm}(t) \in \mathbb{R}^n$ represents the states of a reference model. For a positive definite matrix $Q \in \mathbb{R}^{n\times n}$ there exists a positive definite matrix $P \in \mathbb{R}^{n\times n}$ which satisfies the Lyapunov equation $A_e^T P + PA_e + Q = 0$, where $A_e \in \mathbb{R}^{n\times n}$ is Hurwitz and denotes the state of the error dynamics formed by the respective pseudo control variables. A common update law ([1, 11, 18]) for the adaptive weights is then given by

$$\dot{W}(t) = -\Gamma\sigma(x)e^T PB - m(x,W). \tag{15}$$

Here $m(x,W) \in \mathbb{R}^{m\times(n_2+1)}$ denotes a modification term, which is required in order to guarantee boundedness of the adaptive weights and therefor stability of the closed loop system. These include among others σ - Modification ([12]), e - Modification ([17]) or Q-Modification ([28]).

The linear control design based on the chosen inversion model is only valid if the effect of the uncertainty on the feedback linearized system is minimized. This in turn requires the convergence of the adaptive parameters to a set of weights which approximates the uncertainty best and therefore minimizes the functional approximation error. In classical adaptive control parameter convergence is subject to a condition of PE on the regressor vector. To ensure PE regressor vectors in neuro-adaptive control is in most cases neither possible nor operationally desirable.

In this work, instead of one of the previously mentioned modification terms, concurrent learning adaptive control ([5, 7]) is used. Concurrent learning uses online recorded information concurrently with current data in order to update the adaptive parameters. The key idea is storing information at a time when the regressor vector was exciting, and using this data for future updates. This allows the parameters to converge to a set of weights which minimize the functional approximation error.

Concurrent learning achieves parameter convergence by comparing the current estimation of the uncertainty ν_{ad} with the stored one and updating the adaptive parameters based on this deviation. Therefore, the regressor vector $\sigma_j(x)$ at certain time instants t_j is stored in a matrix $\sigma_H = [\sigma_1, \sigma_2, ..., \sigma_p]$, where $\sigma_H \in \mathbb{R}^{(n_2+1)\times p}$ is called history stack in the following and $p \geq (n_2+1)$. Only regressor vectors are stored which are linearly independent to the already stored data points. Once the history stack is full, methods such as the minimum singular value maximization approach ([8]) exist to include additional points by exchanging them with older data. Apart from the regressor vectors, also the model uncertainty Δ_j has to be determined. With regard to equation (3) the model uncertainty Δ_j at a time instant t_j is calculated by

$$\Delta_j = \dot{x}_j - \nu_j. \tag{16}$$

In order to solve equation (16), knowledge about the first state derivative $\dot{x}_j$ is required. In most cases, state derivatives cannot be measured directly and have to be estimated. For concurrent learning, the estimates of $\dot{x}_j$ do not have to be available instantaneously. Rather estimation methods can be applied which require an amount of time in order to arrive at a good estimate. One such technique is optimal fixed point smoothing ([10]). Optimal fixed point smoothing arrives at a state estimate at time t by using all available data in a time frame $0 \leq t \leq T$. In particular, the smoother combines a Forward Kalman Filter up to time t with a backwards iterated Kalman Filter which uses all data from t up to T (see e.g. [16] for smoother equations). After estimation is finished, the model uncertainty Δ_j is stored along with the respective regressor vector σ_j. For each point the training signal based on stored data can be calculated as follows:

$$\varepsilon_j = W^T(t)\sigma_j - \Delta_j \tag{17}$$

The modified update law of equation (15) becomes

$$\dot{W}(t) = -\Gamma\sigma(x)e^T PB - \Gamma\sum_{j=1}^{p}\sigma_j\varepsilon_j^T. \tag{18}$$

If the history stack contains at least $n_2 + 1$ linearly independent data points, the update law in (18) ensures robustness of the closed loop system and convergence of the adaptive parameters to a set of weights, which minimize the functional approximation error. For reasons of brevity a proof is omitted here. For further information refer to [5] and [7].

2.4 Control Allocation

The following section gives an overview of the component of the controller which remains vehicle specific, the control allocation (CA). Every vehicle type is uniquely designed in respect to position, size and number of the control surfaces. The calculation of the actual surface deflections based on the forces and moment requirements is thus treated vehicle dependent. Control allocation has been extensively studied in literature with respect to nominal flight behavior and as active methods to increase fault tolerance ([27]).

In general CA is treated differently depending on the number of actuators in relation to degrees of freedom. For over- and under actuated systems, CA becomes an optimization problem, while for an equal number of independent actuators to degrees of freedom CA falls back to an algebraic relation. In this work an incremental CA was chosen, which is often also referred to as direct CA. Incremental refers to the fact that a global relation between moment or forces to deflection is not available. Rather, a change in moment implies a change of the deflection using linear relation based on the effectiveness matrix B_{eff} ([9]). In this work, force to thrust calculation and moment to deflections of control surfaces is done separately. A changes of the Moments ΔM is hence given by

$$\Delta M = B_{eff} \Delta \delta_{rot}, \tag{19}$$

where $\Delta \delta_{rot}$ is the change in rotational control surfaces. The control input $\delta_r ot$ is calculated using an estimate of the current control input $\delta_{est,rot}$,

$$\delta_{rot} = \delta_{est,rot} + B_{eff}^{-1} \Delta M. \tag{20}$$

An equivalent, scalar relation yields the thrust. For the two fixed-wing UA of this paper (see below for further information) two different control allocation approaches have to be considered.

Prometheus is equipped with pairs of elevators, aileron and rudder surfaces. Both elevators and rudder are deflected symmetrically while ailerons are deflected antisymmetrically thus resulting in a set of three independent control variables. The resulting effectiveness matrix is quadratic in nature. In contrast, Extreme Star offers a total of 16 control devices which can be actuated independently. However, for the purpose of this paper only eleven control inputs are considered. These include both canards, both ailerons, both flaps, both elevators, the rudder and the throttle for each main wing motor. As a result the control effectiveness matrix is non-quadratic.

Optimization methods such as constraint minimization can optionally be applied to find an adequate solution.

3 Numerical Simulation

This section outlines numerical simulations in order to assess the performance of the presented control architecture for different UA. Furthermore, the robustness of the nonlinear controller against disturbances is evaluated.

3.1 Test-Beds and Simulation Environment

Fig. 3 Extreme Star (left) and Prometheus (right)

The fixed-wing unmanned aircraft *Prometheus* is a development by the German Aerospace Center, Braunschweig, Germany, see Figure 3. The high-wing in pusher configuration has a MTOW of $25\,kg$ and is equipped with a PC104 based avionics system containing flight control computer, vision computer and the common set of sensors required for automated flight. Flight control and guidance algorithm use a QNX operating system while vision application run on a Linux based system. Aims of the project are the evaluation of methods and algorithms of unmanned aircraft. These include path-planning, mission management, flight control, sensor fusion and environmental awareness. An additional goal is the operation of multiple UA of different type within a unique software framework. Especially for the Prometheus project, also fixed-wing specific research from operation perspective, such as aerial refueling, are of interest.

The *FSD ExtremeStar* bases on the off-the-shelf polystyrene model airplane Multiplex TwinStar II and was modified by the AkaModell Munich on behalf of the Institute of Flight System Dynamics of the Technische Universität München. In particular these modifications include the addition of canards with variable incidence, the extension of the fuselage, the conversion of the trailing edge of the inner wing part into flaps and the replacement of the existing motors with higher performance

motors with pitch axis thrust vectoring. Furthermore, an additional tiltable third propeller was attached to the tail of the aircraft. An independent control of left and right side control devices is possible, therefore offering a total of 16 actuators. The main purpose of the airplane is to analyze new and existing control methods for an UA with a large number of control devices. For the purpose of this work only eleven control inputs are considered.

Algorithms normally run through two specific simulation stages before going into flight test as depicted in Figure 4. First stage is the software-in-the-loop (SIL) simulation, which is used for development of the algorithms themselves. They are tested in a single computer setup, simulating all aspects of the unmanned aircraft system (UAS). The SIL typically consists of the mission manager, which handles the mission components and provides an interface to the ground control station. The flight controller receives commands of the mission management and the flight dynamics are used to simulate the UA flight mechanics. Optionally, a realistic sensor fusion can be used, which contains sensor emulation as well as an algorithm like a Kalman filter for state estimation ($\hat{x}$). The advantages of the SIL simulation are a fast development cycle, reduced recourse requirements and modular level of abstraction and thus simulation complexity.

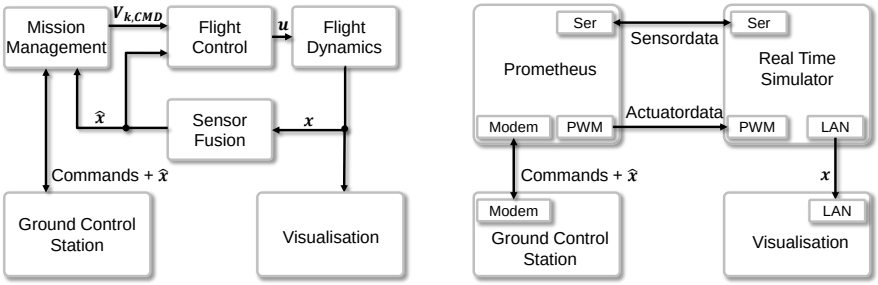


Fig. 4 Software-in-the-loop (left) and hardware-in-the-loop (right) simulations

After integration of the algorithm into the flight software, it is tested within the hardware-in-the-loop (HIL) simulation. For this work a HIL framework was only available for Prometheus. For the simulation setup the complete vehicle is integrated into the simulation including the processing units of the avionics. A real time simulator based on dSpace simulates the flight dynamics as well as sensor emulation. The emulated sensor data is supplied via serial (Ser) connection to the avionics. The UA is controlled via the GCS in the same manner as it is in flight test. In both simulation cases, SIL and HIL, the same visualization is used to give a user feedback of the real state of the UA.

3.2 Test Scenario and Results from Numerical Simulation

At first, the control architecture is tested in the SIL framework. For robustness considerations parameter uncertainties are artificially introduced to the system. In particular, the mass moment of inertia I_{BB} is increased by 15%, the aerodynamic derivative of the lift due to an angle of attack $C_{L,\alpha}$ is decreased by 10% and the mass of the aircraft m is increased by 10%. Additionally, a constant wind from northeast disturbs the aircraft with $V_W = [-5, -2.5, 0]^T \frac{m}{s}$.

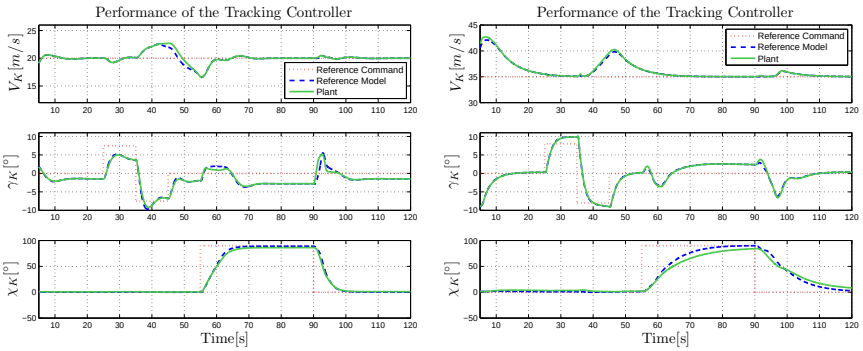
During the numerical simulation, the aircraft shall track a series of path commands represented by step inputs. The simulation runs a total of $120s$ with a time step of $0.01s$. The initial conditions for the path states are given with $V_{K,0} = 40\,\frac{m}{s}$, $\gamma_{K,0} = -10^\circ$ and $\chi_{K,0} = 0^\circ$ for FSD ExtremeStar and $V_{K,0} = 40\,\frac{m}{s}$, $\gamma_{K,0} = -10^\circ$ and $\chi_{K,0} = 0^\circ$ for Prometheus. In order to compare the results each aircraft receives the same commands. The reference signals are comprised of several step inputs. After $25\,s$ the climb angle is commanded to be $\gamma_K = 7.5^\circ$ for $10\,s$ and $\gamma_K = -7.5^\circ$ for $10\,s$ after that. After $55\,s$ the aircraft are commanded to perform a 90° right turn followed by a 90° left turn after $90\,s$. During the simulation the velocity is held constant at $V_K = 20\,\frac{m}{s}$ for FSD ExtremeStar and $V_K = 35\,\frac{m}{s}$ for Prometheus, respectively.

Figures 5(a) and 5(b) show the tracking performance of Prometheus and ExtremeStar respectively if no adaptive controller is used in any loop. For Prometheus in Figure 5(a) it can be seen that the plant follows the reference trajectory without major deviations even in the presence of parameter errors and the external disturbance. However, Pseudo Control Hedging in the respective loops alters the reference trajectory of the longitudinal axis in the outer loop such that it deviates from the commanded signal significantly. This is due to the presence of modeling uncertainties in the inner loop and outer loop. In order to achieve steady state accuracy in the longitudinal axis, the effect of modeling uncertainties needs to be reduced, which can be achieved by applying adaptive elements to the system.

Similar to Prometheus, ExtremeStar in Figure 5(b) is able to track the reference model accurately, but PCH prevents the reference model from tracking the commanded signal. In particular, while the performance of the lateral motion increases compared to Prometheus, the longitudinal performance worsens, thus prohibiting a successful operation of the aircraft.

It can be observed that the two aircraft exhibit different performance properties. This deviation is attributed to the fact that both aircraft significantly differ in architecture and parameters, in particular size, mass and number of control surfaces.

In the following the same simulations are performed while adaptive elements are added to the system. Since the relations in the attitude loop are purely kinematic, no model uncertainty are expected here. Hence, adaptive elements are only added to the inner and outer loop. In particular a radial basis function neural network with concurrent learning update laws augments the inner loop, consisting of 125 neurons, which are evenly distributed in the state space. The RBF receives only the rotational rates as inputs. The learning rate is set to $\Gamma = 1.5$. For concurrent learning a total of 130 points are stored in a static history stack. Additionally for testing purposes a neural network with sigmoid activation functions is added to the outer loop, con-



(a) (b)

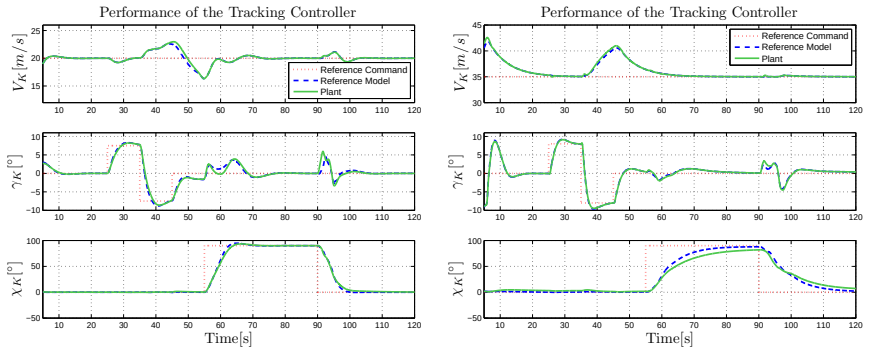
Fig. 5 Simulation of ExtremeStar (5(a)) and Prometheus (5(b)) in the SIL framework with wind and uncertainties but without adaptive controller.

sisting of a total of nine neurons. Note, that in this case also the input weights are updated, thus the neurons do not have to be distributed in the state space a priori. Hence, the number of neurons needed for adaptation is significantly lower than in the case of radial basis functions neural network. The input vector to the outer loop network consists of the path, attitude and rotational states. The learning rates are set to $\Gamma_V = 2$ for the input weights and $\Gamma_W = 0.1$ for the output weights. The latter are chosen to be small in order to prevent the propagation of errors resulting from fast adaptation to the middle loop. For the concurrent learning update law a total of 30 points are stored.

Figures 6(a) and 6(b) show the tracking performance of Prometheus and Extreme Star with adaptive elements. It can be seen that especially the performance in the longitudinal motion drastically increases in both cases. This improvement leads to the conclusion that the neural networks approximate the uncertainty such that its effects on the plant dynamics are significantly reduced. Furthermore, the control architecture is seen to be robust not only against the parameter uncertainties but also against the disturbance by wind.

In addition to the SIL simulation, a HIL framework based on dSpace is available for Prometheus. The simulation runs for a total of $175\,s$ with a time step of $0.01\,s$. The initial values for the desired path are given with $V_{K,0} = 30\,\frac{m}{s}$, $\gamma_{K,0} = 0^\circ$ and $\chi_{K,0} = 0^\circ$. The reference signals are comprised of several step inputs. Prometheus shall slow to $V_K = 20\,\frac{m}{s}$ after $80\,s$ and accelerate to $V_K = 40\,\frac{m}{s}$ after $138\,s$. During the different velocity phases climb anlge and course angle are varied. After $70\,s$ the climb angle is commanded to be $\gamma_K = 7.5^\circ$ for $5\,s$ and $\gamma_K = -7.5^\circ$ for $5\,s$ after that. The same maneuver is initiated after $125\,s$ and $165\,s$. In the second maneuver, the duration of the steps is extended to $8\,s$. Finally, after $41\,s$, $95\,s$ and $150\,s$ the UA is commanded to perform three consecutive 90° right turns.

Figure 7 shows the tracking performance of the nonlinear adaptive path controller for the HIL simulation. It can be seen that the controller is able to follow the desired



(a) (b)

Fig. 6 Simulation of ExtremeStar (6(a)) and Prometheus (6(b)) in the SIL framework with wind and uncertainties; inner loop augmented by radial basis function neural network, outer loop augmented by single hidden layer neural network with sigmoid activation functions.

path if the real Flight Control System Hardware is used. The deviation between the reference model output and the measured states is minor. The jump in the reference model of the course angle after $130\,s$ is attributed to the fact that χ_K is limited to $(-\pi...\pi]$. However, it can also be seen that for higher velocities the trajectory generated by the reference model deviates from the commanded signal. This is attributed to the fact that the uncertainties in the inner loop and outer loop, especially the generated lift and the gradients in the control allocation, are dependent on the velocity. Concurrent learning is formulated for constant optimal weights. Hence, the change in velocity and consequently a change of the optimal weights lead to a drop in performance. However, the control architecture is still seen to be robust.

4 Conclusion

In this paper we showed how an adaptive trajectory control architecture can be set up in order to be simply applicable to multiple fixed-wing UA. Key part of the controller is a generic inversion part, which feedback linearizes a chosen inversion model. Adaptive elements, in the form of concurrent learning neural networks, significantly decrease the effect of model uncertainties between the plant dynamics and the inversion model, therefore increasing the applicability of the control architecture to a broad spectrum of fixed-wing aircraft. While the former can be simply applied to any fixed-wing UA which shares the same signal interface, the gains of the linear controller have to be selected and a control allocation has to be constructed for each aircraft individually. Still, by generalizing the feedback linearization, the amount of time required to set up a reliable flight control system decreases significantly. Results from numerical simulation showed that the exclusive use of feedback linearizaa-

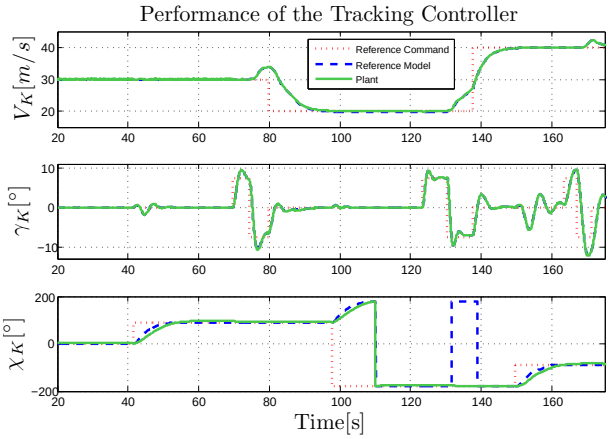


Fig. 7 Hardware-in-the-Loop simulation of Prometheus

tion doesn't result in acceptable performance. Hence, adaptive elements have to be used in order to decrease the effects of the uncertainty. By using concurrent learning adaptive control the adaptive parameters are driven to a set of weights which minimize the approximation error, thus increasing the performance significantly compared to instantaneous learning laws. Further improvement of the proposed control architecture can be achieved by e.g. finding methods to generalize the construction of the control allocation. Furthermore, the performance and limitations of the proposed architecture need to be evaluated in flight tests.

5 Acknowledgment

The authors gratefully acknowledge the support of both, the Unmanned Aircraft Department of DLR and the Institute of Flight System Dynamics at the Technische Universität München. For the valuable discussions we especially thank Fabian Klüssendorf, Leonhard Höcht and Thomas Bierling.

References

1. Karl J. Åström and Björn Wittenmark. *Adaptive control*. Dover Publications, Mineola and N.Y, 2 edition, 2008.
2. Thomas Bierling, Leonhard Höcht, Florian Holzapfel, Rudolf Maier, and Andreas Wildschek. Comparative analysis of mrac architectures in a unified framework. In *AIAA Guidance, Navigation, and Control Conference*, Guidance, Navigation, and Control and Co-located Conferences. American Institute of Aeronautics and Astronautics, 2010.
3. Rudolf Brockhaus, Wolfgang Alles, and Robert Luckner. *Flugregelung*. Springer Berlin, Berlin, 3 edition, 2010.
4. A.J Calise and R.T Rysdyk. Nonlinear adaptive flight control using neural networks. *IEEE Control Systems Magazine*, 18(6):14–25, 1998.
5. Girish Chowdhary. *Concurrent Learning for convergence in Adaptive Control without Persistency of Excitation*. PhD thesis, Georgia Insitute of Technology, Atlanta and GA, 2010.
6. Girish Chowdhary and Ravindra Jategaonkar. Aerodynamic parameter estimation from flight data applying extended and unscented kalman filter. *Aerospace Science and Technology*, 14(2):106–117, 2010.
7. Girish Chowdhary and Eric Johnson. Flight test validation of a neural network based long term learning adaptive flight controller. In *AIAA Guidance, Navigation, and Control Conference*, Guidance, Navigation, and Control and Co-located Conferences. American Institute of Aeronautics and Astronautics, 2009.
8. Girish Chowdhary and Eric N. Johnson. A singular value maximizing data recording algorithm for concurrent learning. In *American Control Conference*, San Francisco, CA, June 2011.
9. Christopher Edwards, Thomas Lombaerts, and Hafid Smaili. *Fault tolerant flight control: A benchmark challenge*. Springer, Berlin, 2010.
10. Arthur Gelb. *Applied optimal estimation*. M.I.T. Press, Cambridge and Mass., 1974.
11. Florian Holzapfel. *Nichtlineare adaptive Regelung eines unbemannten Fluggerätes*. PhD thesis, Technische Universität München, Munich, 2004.
12. P.A Ioannou and P.V Kokotovic. Instability analysis and improvement of robustness of adaptive control. *Automatica*, 20(5):583–594, 1984.
13. E. N. Johnson. *Limited Authority Adaptive Flight Control*. PhD thesis, Georgia Insitute of Technology, Atlanta and GA, 2000.
14. F.L Lewis. Nonlinear network structures for feedback control. *Asian Journal of Control*, 1(4):205–228, 1999.
15. M. M. Monahemi and M. Krstic. Control of wing rock motion using adaptive feedback linearization: Journal of guidance, control, and dynamics. *Journal of Guidance, Control, and Dynamics*, 19(4):905–912, 1996.
16. Maximilian Mühlegg, Eric Johnson, and Girish Chowdhary. Concurrent learning adaptive control of linear systems with noisy measurements. In *AIAA Guidance, Navigation, and Control Conference*, Guidance, Navigation, and Control and Co-located Conferences. American Institute of Aeronautics and Astronautics, 2012.
17. K. Narendra and A. Annaswamy. A new adaptive law for robust adaptation without persistent excitation. *IEEE Transactions on Automatic Control*, 32(2):134–145, 1987.
18. Kumpati S. Narendra and Anuradha M. Annaswamy. *Stable adaptive systems*. Dover Publications, Mineola and N.Y, 2005.
19. Nhan T. Nguyen and Stephen A. Jacklin. Neural net adaptive flight control stability. In *Verification and Validation Challenges, and Future Research," IJCNN Conference*, 2007.
20. J. Park and I. W. Sandberg. Universal approximation using radial-basis-function networks. *Neural Computation*, 3(2):246–257, 1991.
21. R.M Sanner and J.-J.E Slotine. Gaussian networks for direct adaptive control. *IEEE Transactions on Neural Networks*, 3(6):837–863, 1992.
22. Brad A. Seanor. *Flight Testing of a Remotely Piloted Vehicle for Aircraft Parameter Estimation Purposes*. PhD thesis, West Virginia University, Morgantown and West Virginia, 2002.

23. P. Shankar, R. K. Yedavalli, and J. J. Burken. Self-organizing radial basis function networks for adaptive flight control. *Journal of Guidance, Control, and Dynamics*, 34(3):783–794, 2011.
24. Shuang Cong and Ruixiang Song. An improved b-spline fuzzy-neural network controller. pages 1713–1717.
25. Brian L. Stevens and Frank L. Lewis. *Aircraft control and simulation*. J. Wiley, Hoboken and N.J, 2 edition, 2003.
26. Gang Tao. *Adaptive control design and analysis*. Wiley-Interscience, Hoboken and N.J, 2003.
27. Michel Verhaegen, Stoyan Kanev, Redouane Hallouzi, Colin Jones, Jan Maciejowski, and Hafid Smail. Fault tolerant flight control - a survey. In Manfred Morari, Manfred Thoma, Christopher Edwards, Thomas Lombaerts, and Hafid Smaili, editors, *Lecture Notes in Control and Information Sciences*, pages 47–89. Springer Berlin Heidelberg, Berlin and Heidelberg, 2010.
28. K.Y Volyanskyy, W.M Haddad, and A.J Calise. A new neuroadaptive control architecture for nonlinear uncertain dynamical systems: Beyond sigma - and e-modifications. *IEEE Transactions on Neural Networks*, 20(11):1707–1723, 2009.