Framework for application of linear control methods to nonlinear flight control

Zoe Mbikayi [*], Alexandr V. Efremov[†], Yury V. Tiumentsev[‡]
Moscow Aviation Institute, Moscow, 125080

This paper presents a framework where a linear control method can be used to achieve adaptive and nonlinear control of an aircraft by performing an online approximation of the dynamical model of the aircraft and computing the control law in-flight on the approximated model. Two methods are chosen to be tested in this paper. LQR control and dynamic inversion. The former is chosen due to its importance and influential status, and the latter due to its simplicity.

Nomenclature

A	=	State Matrix
B	=	Input matrix
c	=	Control stick deflection
d	=	Stationarity distance (seconds)
e	=	Error
E	=	Augmented matrix containing x and u
F	=	Low pass filter
$G(s)$	=	Aircraft linear model
g	=	Solution matrix containing A and B
h	=	Matrix of l state vectors, Altitude
i	=	Input signal
K	=	Controller gain matrix
KG	=	Kalman gain
l	=	Size of replay buffer
m	=	Number of inputs in vector u
n	=	Number of states in vector x
n_e	=	Pilot noise (remnant)

[*]Research Engineer, Department of Flight Dynamics and Control, mbikaisz@mai.ru
[†]Professor, Head of laboratory, Department of Flight Dynamics and Control, pvl@mai.ru
[‡]Professor, Department of Flight Dynamics and Control, tium@mai.ru

N	=	Scaling factor for reference tracking in LQR
Q	=	State cost matrix
q	=	Pitch rate
$\dot{q}$	=	Pitch acceleration
$\dot{q}_d$	=	Desired Pitch acceleration
R	=	Input cost matrix
r	=	Overdetermined system residual
T_c	=	Filter time constant
T_s	=	Time step
u	=	Input vector, horizontal velocity
V	=	Matrix of stable eigenvectors, Airspeed
w	=	Vertical velocity
W_C	=	Controlled object dynamics
W_P	=	Pilot dynamics
x	=	State vector
$\dot{x}$	=	State vector derivative
X	=	Solution to the Riccati equation
Z	=	Hamiltonian of Riccati equation
α	=	Angle of attack
γ	=	Flight path angle
η	=	Elevator deflection angle
θ	=	Pitch angle
ω	=	Bandwidth

I. Introduction

Controlling nonlinear systems is not always straightforward [1]. Several nonlinear control methods have been developed over the years. One of the most popular is feedback linearization [2–7]. Feedback linearization is often used in the nonlinear dynamic inversion scheme in order to cancel the system nonlinearities, which then allows to use linear control techniques [8–10]. This method however requires extensive knowledge and understanding of the dynamical system under study. This requirement is almost impossible to satisfy because there are always uncertainties present in real systems. A popular approach to solving this problem is by combining the nonlinear dynamic inversion with robust control methods such as the μ synthesis or H_∞ control [11, 12]. Using this approach has been quite effective, but still not all uncertainties are taken into account, or some known nonlinear dynamics are considered as uncertainties. The resulting control system can therefore be marginally or overly conservative in performance and stability robustness [13]. To solve these issues, the incremental nonlinear dynamic inversion appeared, first as simplified nonlinear dynamic inversion [14, 15]. This method has been widely applied to various aerospace systems [16–21]. Incremental nonlinear dynamic inversion works by feeding back synchronous sensor measurement or estimation of control deflection and angular acceleration. This reduces the requirement of accurate knowledge of the dynamics of the vehicle to the knowledge of the control effectiveness matrix. A linear implementation of the nonlinear dynamic inversion controller is evaluated later in this work within the framework of the method presented.

Another popular method is the classical gain scheduling, which has been extensively studied for all kinds of systems and in all kinds of configurations [22–26]. The main idea of method is based on linear plant varying models. By having several of linear models obtained at various operating points throughout the operation envelope of the system, it is possible to use linear controllers computed for each operating point. Then a multivariate surface can be built such that the controllers smoothly transition to each other depending on some predefined scheduling variables.

Besides the feedback linearization and gain scheduling based methods, there have been other methods developed that are Lyapunov based. One of them is the sliding mode control [27–29]. As its name suggest, the sliding mode control technique tries to make the system slide along a crossection of its normal behavior by applying certain classes of discontinuous control signals. But in essence, applying discontinuous signals means the structure of the controller changes. Which brings this method close to gain scheduling.

The classes of methods discussed above can be considered as deterministic algorithms. Current advances in data science and machine learning gave rise to a generation of new methods that are nondeterministic in nature, in control theory in general and flight control in particular. These methods take advantage of simulation environments to generate data that can be used to build models or synthesize control systems.

This paper presents a framework based on a nondeterministic approach, where a nonlinear system is rendered linear, in order to apply a linear control technique. This is achieved by performing an online local linear approximation of the nonlinear system, and computing a control law directly online on the estimated linear system. This method inherently solves the robustness issue by being adaptive, and furthermore it only requires an initial approximate model of the system under study, and does not require extensive knowledge of the said system.

II. Online approximation

A. Linear approximation

The framework is based on linear mathematical modeling using algebraic principles. As shown in [30, 31], nonlinear systems can be represented approximately by several piecewise linear systems. This means that a nonlinear system given as shown in Eq. 1 can be represented at any operating point as shown in Eq. 2.

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

$$\dot{x} = Ax + Bu \tag{2}$$

Where A is the $[n \times n]$ state matrix with n states, and B is the $[n \times m]$ input matrix with m outputs.

With this background in mind, the approximation problem can be tackled, and reduced to finding matrices A and B given a set of l data points of the tuple $[u, x, \dot{x}]$.

Algebraically, the problem can be written as the system of equations shown in Eq.3, which can in turn, following basic principles of linear algebra, be re-written as Eq. 4

$$\begin{cases} \dot{x}_1 = Ax_1 + Bu_1 \\ \dot{x}_2 = Ax_2 + Bu_2 \\ \qquad \vdots \\ \dot{x}_l = Ax_l + Bu_l \end{cases} \tag{3}$$

$$Eg = h, \tag{4}$$

where E is an $[l \times (n + m)]$ matrix of rank $(n + m)$, with $l \geq (n + m)$ created by concatenation of vectors x and u such that $E = [x, u]$ and g is a matrix containing the solutions A and B such that $g = [A, B]$.

It can be seen that Eq. 3 is a system of equations where the number of equations can be more than the number of variables. The solution to a system of equations represented as in Eq. 4 would be obtained by multiplying both sides of the equation by the inverse of E. However, since E is not square, computing the inverse becomes impossible, and therefore there is no solution to the system.

However, an approximate solution can be found by isolating g [32].

Let us multiply both sides of the equation by E^T and get Eq. 5.

$$E^T Eg = E^T h \tag{5}$$

It is clear that $E^T E$ is a square matrix, and therefore g can be isolated by multiplying both sides of Eq. 5 by the inverse of $E^T E$, namely $(E^T E)^{-1}$. The solution is therefore obtained as given by Eq. 6

$$g = (E^T E)^{-1} E^T h, \tag{6}$$

where $(E^T E)^{-1} E^T$ is known as the Moore-Penrose inverse [33] of E, whose existence is a necessary condition for the solution of Eq. 4.

Given a set of l data points of the tuple $[u, x, \dot{x}]$, Eq. 6 results in a matrix g containing the approximated system represented by matrices A and B, which are necessary for the computation of the control laws when using most linear control methods.

The solution shown above is the least square solution, which can be explained as follows:
Since the system presented in Eq. 3 has no exact solution, there will always be some residual for each equation in the system. This residual is given in Eq. 7

$$r = h - Eg, \tag{7}$$

where $r = [r_1 ... r_l]^T$.
The sum of the residuals make out the variance of error given as the total error in Eq.8.

$$e(g) = ||r||^2 = r^T r = (h - Eg)^T (h - Eg) = h^T h - g^T E^T h - h^T Eg + g^T E^T Eg \tag{8}$$

The solution g that minimizes the error $e(g)$ can be obtained as given in Eq. 9.

$$\begin{aligned} \frac{\partial}{\partial g} ||r||^2 = 0 = \frac{\partial}{\partial g} \left(h^T h - g^T E^T h - h^T Eg + g^T E^T Eg \right) = 2(-E^T h + E^T Eg) \\ \equiv E^T Eg = E^T h \end{aligned} \tag{9}$$

It can be seen that Eq. 9 is the same as Eq. 5. Which means, the solution obtained in 6, is the least square approximate solution.

B. Replay buffer

The method presented in the previous section uses a saved dataset in order to approximate the best linear system fitting the recorded data. However, that operation is still performed offline. In order to use the method online, a replay buffer is introduced.

The replay buffer is a cache of finite size l where the tuple $[u, x, \dot{x}]$ is stored at each time step. When the memory limit of the replay buffer is reached, the oldest tuples are discarded to give space to new ones, such that the replay buffer always contains l last data points. This allows the use of a finite amount of data in an online setting, and allows to apply the approximation method presented in the previous section. The size l of the buffer is selected depending on the computational capabilities of the computer performing the control task and the nonlinear character of the system under study.

As it can be seen from Eq. 6, the solution to the online approximation problem includes the computation of the Moore-Penrose inverse, which can be time consuming computationally. Therefore, l must be chosen such that the computational time does not decrease the performance of the system. In addition, there is a wide range of choice of fast algorithms that can be used to find the computationally complex inverse. One of these are algorithms is given in [34].

C. Online approximation algorithm

Using the two concepts defined in the previous two sections, an online approximation algorithm is designed and presented as a pseudo-code shown as algorithm 1.

Algorithm 1: Online local linear approximation of nonlinear system

```
Result: $g = [A, B]$
initialize replay buffer (memory size = $l$);
initialize $A$ and $B$;
while system online do
    store tuple $[u, x, \dot{x}]$ in replay buffer;
    if $t < l$ then
        $A$ = initial $A$;
        $B$ = initial $B$;
    else
        sample $[u, x, \dot{x}]$ from replay buffer;
        $E = [x, u]$ ;
        if $E$ is full rank then
            $g = (E^T E)^{-1} E^T h$ ;
            $g_{previous} = g$;
        else
            if $g_{previous}$ exists then
                $[A, B] = g_{previous}$
            else
                $[A, B]$ = initial $[A, B]$
            end
        end
    end
end
```

The initial A and B are computed offline with a dataset of any size, as long as the data stays close to the operating point chosen for the initial approximation. These initial matrices A and B are used as output of the algorithm while the replay buffer has not been filled.

Another point worth noting is the condition of existence of the Moore-Penrose inverse of E which will always be satisfied if E is full rank. If this condition is not fulfilled, then a mean squared error solution to the identification problem does not exist, and therefore the previous identified model is given as output. The idea of using the previous identified model when the condition fails is motivated by the fact that if E is not full rank, then there has not been enough changes in the states x and input u, which, by logical reasoning, means the dynamics of the system has not changed. Therefore the previous model identified should still hold correct.

D. Stationarity distance

Depending on the control method selected, computing a new control law at each time step might not be a computationally effective method.

Assuming the system is slow varying, a concept of stationarity distance d is introduced and defined as the distance between two linear system dynamics with acceptable transition rate. This means that the system approximation algorithm presented in algorithm 1 will no longer be approximating the system at each time step, but rather at intervals d. And the interval d is chosen such that there is not a big change between the two system dynamics approximated and it is small enough that the change from one linear system to another is smooth.

III. Linear Quadratic Regulator

The linear quadratic regulator method has been chosen to as a linear method to be tried along the approximation algorithm. This choice was mainly because it is one of the most influential and important solutions to the optimal control problem today. The working mechanism of the LQR method is presented in [35].

The LQR solution results in a controller K given as shown in Eq. 10.

$$K = R^{-1}(B^T X) \tag{10}$$

where R is the $[m \times m]$ control cost weight matrix and X is the $[n \times n]$ matrix of solution to the continuous Riccati differential equation shown in Eq. 11.

$$\dot{X} = A^T X + XA - (XB)R^{-1}(B^T X) + Q \tag{11}$$

where Q is the $[n \times n]$ state cost weight matrix.

Solving this differential equation is not computationally expensive but in an online environment, the solution is found via iterations. It was found, via experiments that for a system with 3 states and 1 input, it would take around 400 iterations, or 4 seconds in a simulation with an sampling rate and integration step of 0.01 for the solution to converge.

Therefore, a different solution can be found by assuming an infinite horizon system. This implies that $\dot{X} = 0$ and the LQR problem becomes of solving the algebraic equation shown in Eq. 12.

$$A^T X + XA - (XB)R^{-1}(B^T X) + Q = 0 \tag{12}$$

To solve Eq. 12, the Hamiltonian Z of size $[2n \times 2n]$ is defined as shown in Eq. 13.

$$Z = \begin{bmatrix} A & -BR^{-1}B^T \\ -Q & -A^T \end{bmatrix} \tag{13}$$

The Hamiltonian Z will have $2n$ eigenvalues, where half of them have a negative real part, and half of them a positive real part.

Then the $[2n \times n]$ matrix $V = [V_1; V_2]$ of eigenvectors corresponding to the negative eigenvalues is defined, with V_1 and V_2 both having $[n \times n]$ size.

The solution to the Riccati equation is then calculated as shown in Eq. 14.

$$X = V_2 \cdot V_1^{-1} \tag{14}$$

A. Regulation using LQR

Taking into account the concepts presented above, the control algorithm based on LQR is presented in algorithm 2.

From algorithm 2 it can be seen that at every interval d, the the algorithm gives new system matrices A and B. These matrices are then used to start the iteration process of computing K.

Another thing that can be seen from the algorithm is that the controller K used at any given moment, is the K computed for the previous A and B. This highlights the importance of choosing the right stationarity distance d. The general control loop structure can be seen in Fig. 1.

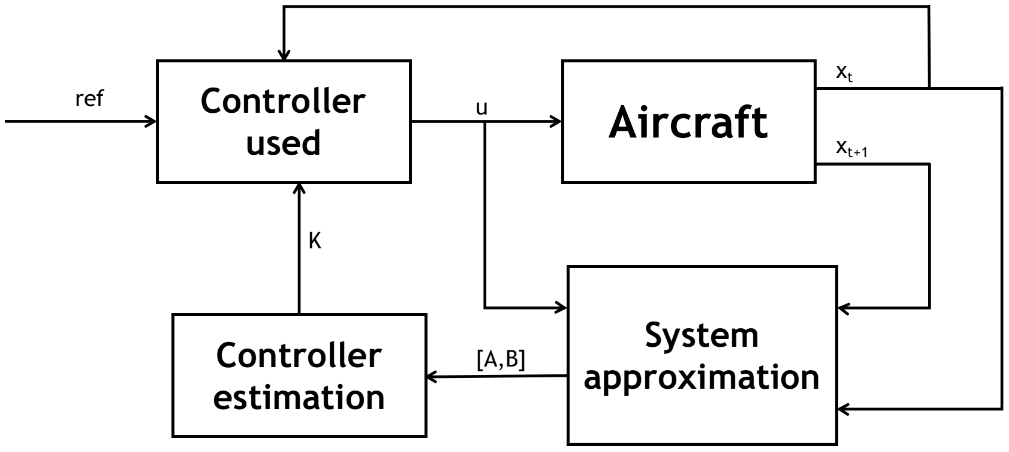


Fig. 1 General control structure with online approximation

Algorithm 2: LQR with online system approximation

```
Result: $g = [A, B]$
initialize replay buffer (memory size = $l$);
initialize $A$ and $B$;
initialize $Q$ and $R$ ;
initialize $X$ ;
initialize $d$;
while system online do
    store tuple $\{u, x, \dot{x}\}$ in replay buffer;
    if $t < l$ then
        $A$ = initial $A$;
        $B$ = initial $B$;
    else
        if mod(t,d) = 0 then
            sample $[u, x, \dot{x}]$ from replay buffer;
            $E = [x, u]$ ;
            if $E$ is full rank then
                sample $\{u, x, \dot{x}\}$ from replay buffer;
                $E = [x, u]$ ;
                $g = (E^T E)^{-1} E^T h$ ;
                $g_{previous} = f$;
            else
                if $g_{previous}$ exists then
                    $[A, B] = g_{previous}$ ;
                else
                    $[A, B]$ = initial $[A, B]$ ;
                end
            end
        else
            $[A, B] = g_{previous}$ ;
        end
    end
    $Z = \begin{bmatrix} A & -BR^{-1}B^T \\ -Q & -A^T \end{bmatrix}$ ;
    find stable eigenvectors $[V_1; V_2]$ ;
    $X = V_2 \cdot V_1^{-1}$ ;
    if mod(t+1,d) = 0 then
        $K = R^{-1}(B^T X)$ ;
        $K_{previous} = K$ ;
    else
        $K = K_{previous}$ ;
    end
end
```

IV. Linear Dynamic Inversion

Another control strategy worth exploring is the linear implementation of the dynamic inversion method. Consider the longitudinal dynamics of an airplane in state space representation as shown in Eq. 15.

$$\begin{bmatrix}\dot{u}\\ \dot{w}\\ \dot{q}\\ \dot{\theta}\end{bmatrix} = \begin{bmatrix}X_u & X_w & X_q & X_\theta\\ Z_u & Z_w & Z_q & Z_\theta\\ m_u & m_w & m_q & m_\theta\\ 0 & 0 & 1 & 0\end{bmatrix}\begin{bmatrix}u\\ w\\ q\\ \theta\end{bmatrix} + \begin{bmatrix}X_\eta\\ Z_\eta\\ m_\eta\\ 0\end{bmatrix}[\eta] \tag{15}$$

The pitch rotational equation of motion can then be re-written as in Eq. 16.

$$\dot{q} = m_u \cdot u + m_w \cdot w + m_q \cdot q + m_\theta \cdot \theta + m_\eta \cdot \eta \tag{16}$$

Eq. 16 can be inversed to find the elevator deflection η needed in order to achieve a desired pitch acceleration $\dot{q}_d$. The block diagram realization of Eq. 17 is shown in Fig. 2.

$$\eta = \frac{1}{m_\eta}(m_u \cdot u + m_w \cdot w + m_q \cdot q + m_\theta - \dot{q}_d) \tag{17}$$

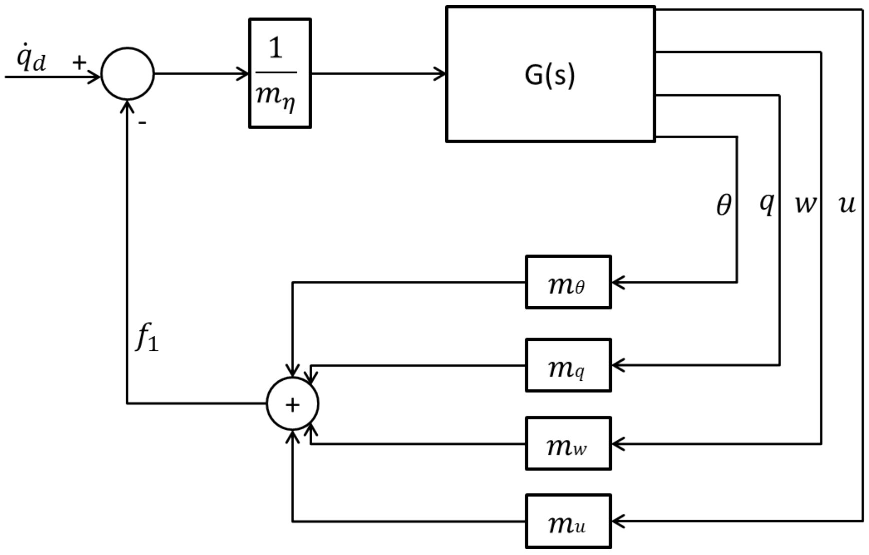


Fig. 2 Implementation of dynamic inversion

The implementation shown in Fig. 2 requires 5 coefficients from matrices A and B representing the longitudinal dynamics.

Knowing the $\dot{q}$ from Eq. 16, the expression f_1 representing the feedback signal in Fig. 2 can be re-written as shown in Eq. 18. This effectively reduces the number of needed coefficients to 1 (m_η), by using the pitch acceleration in feedback. And the realization can be seen in Fig. 3.

$$f_1 = m_u \cdot u + m_w \cdot w + m_q \cdot q + m_\theta \cdot \theta$$

$$f_1 = \dot{q} - m_\eta \cdot \eta \tag{18}$$

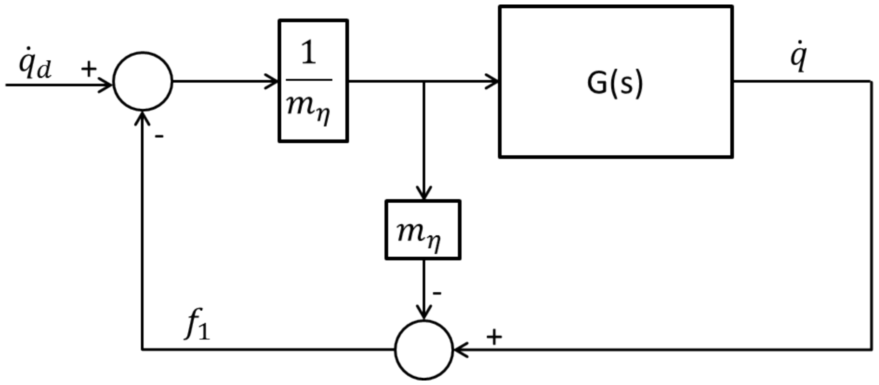


Fig. 3 Realizable dynamic inversion

Using the dynamic inversion method as shown in Figs. 2 and 3 effectively cancels the rotational dynamics of the aircraft with the transfer function $\frac{\dot{q}}{\dot{q}_d}$ being equivalent to 1. However, to achieve pitch rate control, this is not enough. Therefore an outer loop taking the pitch rate q in feedback is added as shown in Fig. 4, where ω is a coefficient defining the closed loop performance as the bandwidth (see Eq. 19).

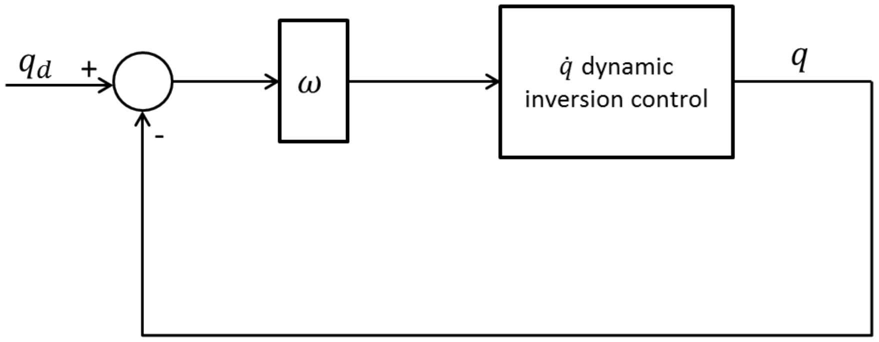


Fig. 4 Pitch rate control

$$\frac{q}{q_d} = \frac{\omega}{s + \omega} \tag{19}$$

V. Effectiveness of online approximation

To study the effectiveness of the approximation technique, two models of aircrafts are used. The first one is a linear coupled longitudinal and lateral model of a helicopter which is unstable. The second model is a longitudinal nonlinear model of a supersonic airplane introduced in [36] and [37]. The stability characteristics of this model are assumed to be unknown.

A. Linear helicopter full model

The highly coupled linear model of the helicopter presented in [38] was used. This linear model was used for the sole purpose of evaluating the identification algorithm, given the full knowledge of the linear model. The evaluation looks the identification accuracy and amount of data needed to complete the identification task. The model has 4 inputs and 8 states that are measurable. The state derivatives are also measurable. Using this model is advantageous because it allows to evaluate the different parameters that go into the approximation algorithm in order to estimate its limitations.

Since the system is linear, the approximation should only yield one approximation that should not vary, and should maintain the unstable character of the original system. Another highlight is that in order to satisfy the Moore-Penrose inverse existence condition, the input signals in each control channel should be linearly independent.

For this evaluation, square signals are given as inputs. The amplitudes and frequencies of the signals are randomly changed after each period, and the approximation is done offline. The input signal is shown in Fig. 5. The signal was the same for the 4 inputs, but with different delays introduced to avoid linear correlations.

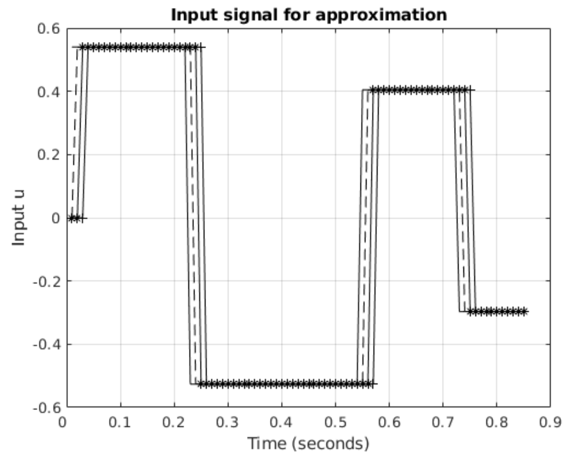


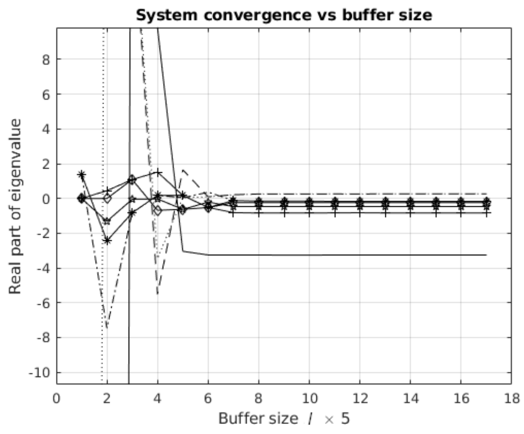
Fig. 5 Input signal for system approximation

1. Buffer size l

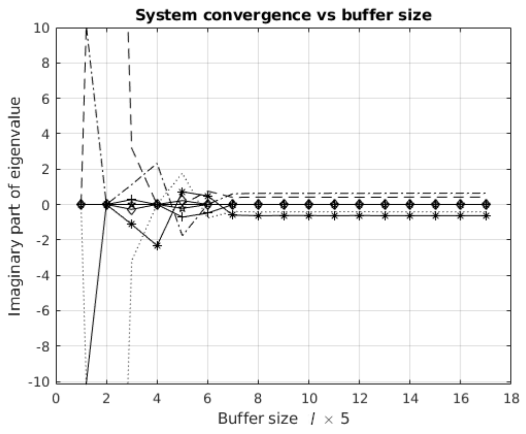
The size of the buffer size l was varied from 5 to 85, with an interval of 5. The simulations were run with a sampling time of 0.01 seconds, and so the time to collect 85 data points was merely 0.84 seconds. For each buffer size, the eigenvalues of the resulting system were recorded and then plotted as a function of the buffer size. The results of this process can be seen in Fig. 6.

As it can be seen, the eigenvalues converge quite quickly and convergence is reached by the time $l = 40$.

The values of the eigenvalues of original system and those of the approximated system shown in table 1. It can also



(a) Real part of the eigenvalues



(b) Imaginary part of the eigenvalues

Fig. 6 Identification convergence as a function of buffer size l

be seen that the approximation has kept the unstable nature of the system.

Table 1 Eigenvalues of real model vs approximated model

Original model	Approximated model
$-3.2514 + 0.0000i$	$-3.2514 + 0.0000i$
$0.2662 + 0.4137i$	$0.2662 + 0.4137i$
$0.2662 - 0.4137i$	$0.2662 - 0.4137i$
$-0.1686 + 0.6332i$	$-0.1686 + 0.6332i$
$-0.1686 - 0.6332i$	$-0.1686 - 0.6332i$
$-0.8335 + 0.0000i$	$-0.8335 + 0.0000i$
$-0.2425 + 0.0000i$	$-0.2425 + 0.0000i$
$-0.4614 + 0.0000i$	$-0.4614 + 0.0000i$

2. Sampling time

In the previous subsection, the influence of the buffer size m on the convergence rate of the system was investigated. And the sampling time was constant and equals to 0.01. In this section, the influence of the sampling time is investigated. To do this, the sampling time is varied, and an attempt is made to define the minimum buffer size m for which the values of eigenvalues will match the real system.

As it can be seen in Fig. 7, up until the sampling time is 0.1, the buffer size required is varying with a mean equivalent to 1 second. And when the sampling time is from 0.1 seconds to 0.9 second, the buffer size required is linearly dependent to the minimum simulation time for data sampling.

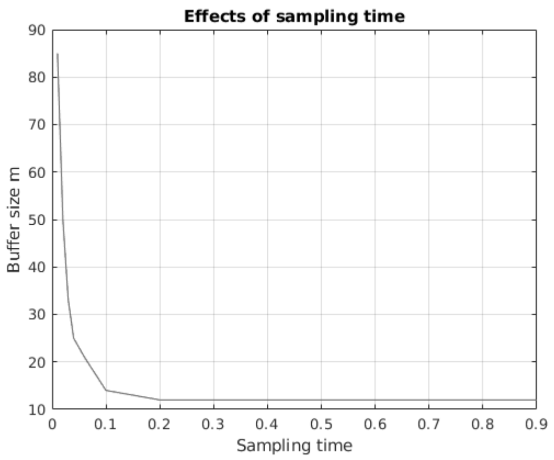


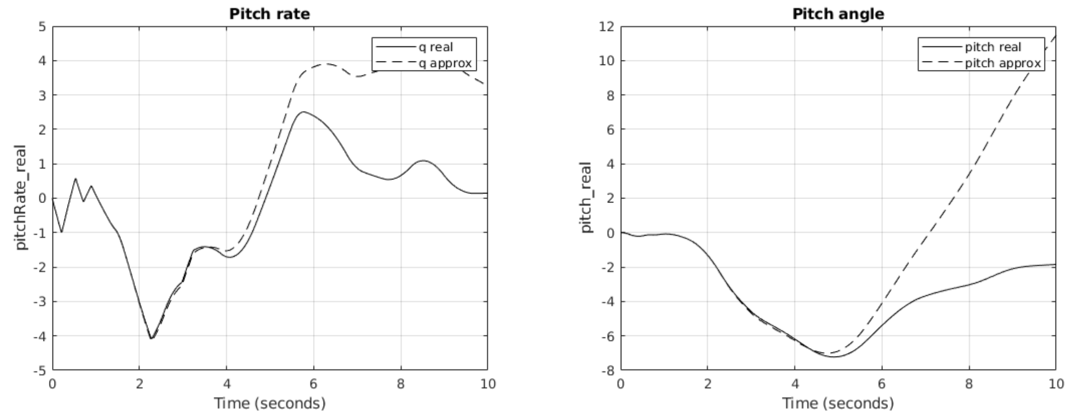
Fig. 7 Effects of sampling time on identification

B. Nonlinear longitudinal airplane model

After assessing the effectiveness of the approximation method on a linear MIMO helicopter model, it was then tested on nonlinear model of the longitudinal motion of a supersonic airplane introduced in [36] and [37].

The model has one input to the longitudinal angular control, and 5 outputs representing three longitudinal states, velocity vector, pitch rate, and pitch and derivatives of these states, acceleration, and jerk. The data needed for the approximation algorithm can be collected directly from these outputs.

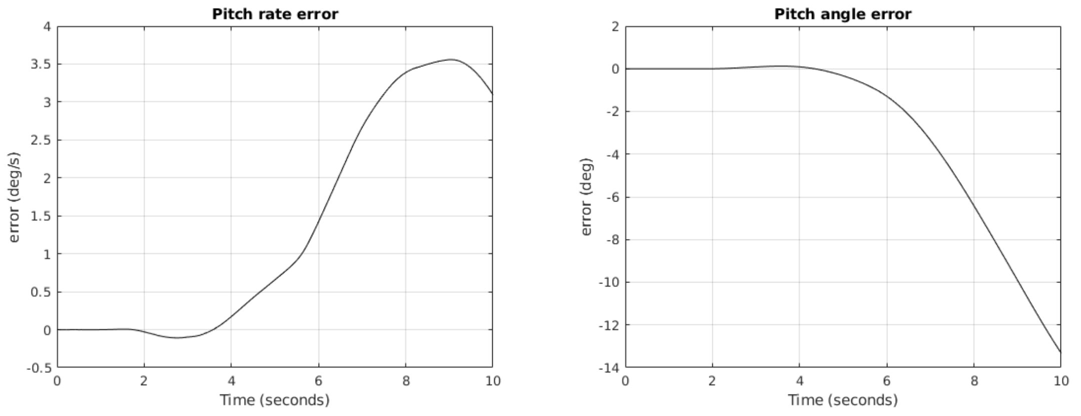
And as done before, 1 second worth of flight data is collected at a sampling rate of 0.01 sec in order to compute an initial approximation of the model. Then a simulation was run both on the real model and using the approximated model, without using the online approximation capabilities. The results of this process are shown in Figs. 8 and 9.



(a) Pitch rate **(b) Pitch angle**

Fig. 8 Using only initial approximation

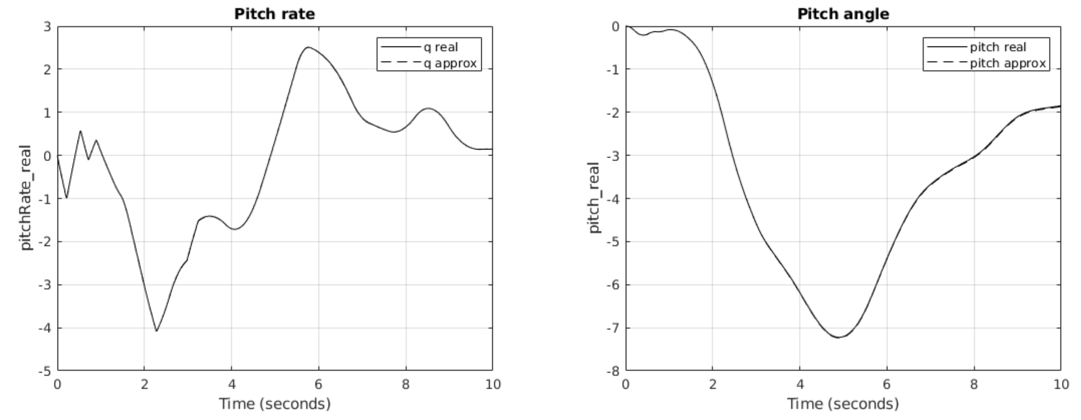
As it can be seen, after about 4 seconds, the approximated model is no longer valid and starts diverging from the true system. This displays the locality of the approximated linear model relative to the nonlinear model.



(a) Pitch rate error **(b) Pitch angle error**

Fig. 9 Error when using only initial approximation

Then the online approximation algorithm was activated with the same input signal and same initial approximation. The comparison between the real model and the approximated model is shown in Figs. 10 and 11.



(a) Pitch rate **(b) Pitch angle**

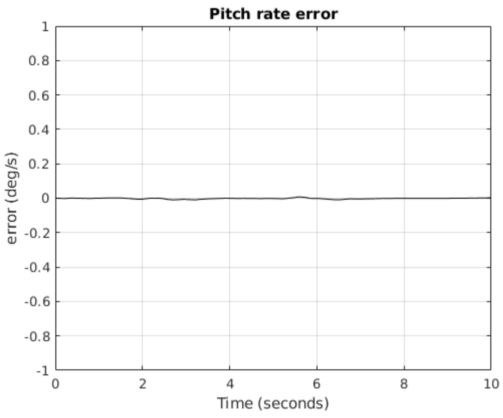
Fig. 10 Online approximation

From this it can be seen that the divergence behavior noticed in Fig. 8 has disappeared as the system is constantly adjusting itself in order to provide the best approximation.

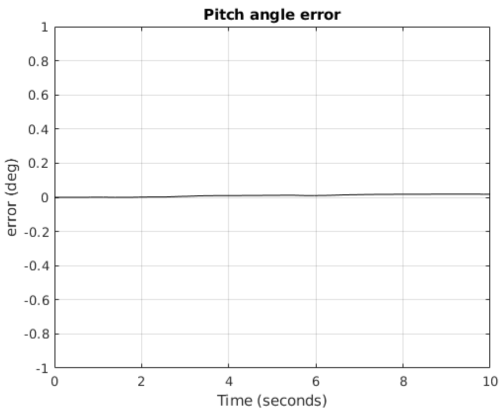
C. Measurement noise

In a real flight setting, the sensors usually pick up noise in their measurement.

To test the robustness of the identification algorithm in this setting, Gaussian noise $N(\mu, \sigma)$ was added to the measured state and input signals such that:



(a) Pitch rate error



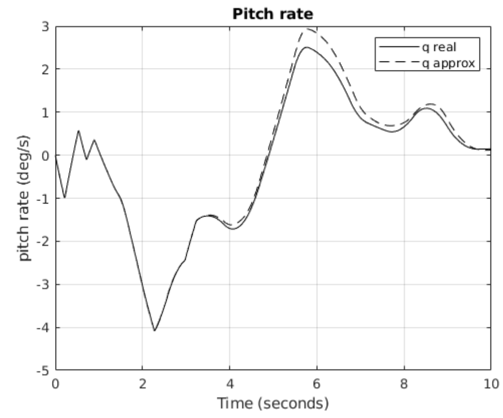
(b) Pitch angle error

Fig. 11 Error with online approximation algorithm

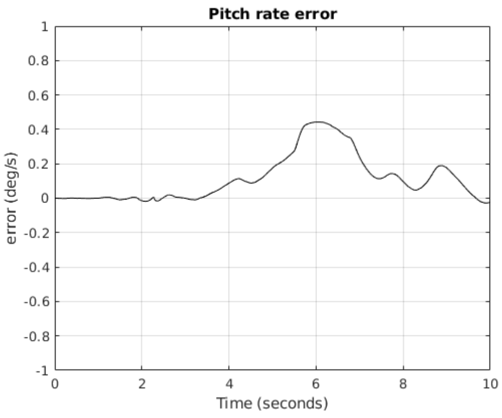
$$\begin{cases} xt_{used} = xt + N(\mu = 0, \sigma = 2) \\ ut_{used} = ut + N(\mu = 0, \sigma = 0.2) \end{cases}$$

The standard deviation value of $\sigma = 2$ represent about 35% of the maximum value of xt without measurement noise. This means that in the extremes of the distribution, the noise values are more than the real values of xt. This is taken as worst case scenario.

And it is assumed that the pilot commands are easily measurable without the same magnitude of noise as the airplane's response. Hence the standard deviation of $\sigma = 0.2$.



(a) Pitch rate



(b) Pitch rate error

Fig. 12 When measurement noise is added

It is seen that even with measurement noise, the algorithm still manages to minimize the error and provides a model that is close to the real model, which is amply enough for control design purposes.

VI. LQR on the nonlinear aircraft model

The LQR method presented in section III was applied on the aircraft model introduced in subsection V.B.

The LQR is a full state feedback method. And since the model used is a longitudinal model, the state vector is comprised of the velocity vector, the pitch, and pitch rate ($[V, q, \theta]$). These measurements are readily available.

The parameters shown in table 2 were used with algorithm 2 in a general structure as shown in Fig. 1.

Table 2 Parameters used

Parameter	Value
l	301
d	0.01 seconds
Q	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1000 & 0 \\ 0 & 0 & 1 \end{pmatrix}$
R	0.1

Additionally, a scaling factor $N = 0.17$ was added to the command signal for reference tracking, as per usual practice when using LQR.

No importance was given to the choice of the weights or scaling factor as this is only intended to study the feasibility of the algorithm and effectiveness of the framework presented.

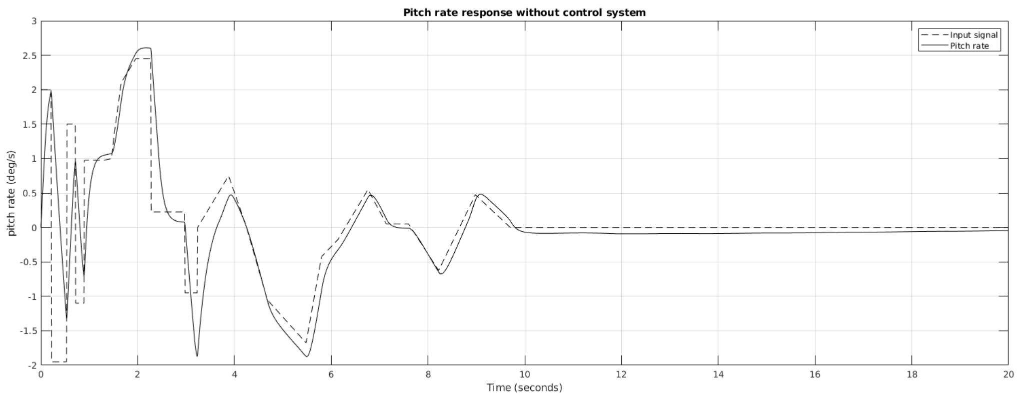


Fig. 13 Online system identification and LQR estimation

The results can be seen in Figure 13. The proposed algorithm was able to make a system identification online and capture the unstable nature of the model. The eigenvalues of the initial model identified offline can be seen in Table 3. Next, the algorithm computed LQR gains needed to stabilize and provide reference tracking capabilities to the identified system, with a stationarity distance of 0.01 seconds. Those gains are then used on the real vehicle and good performance is achieved. For reference, Fig. 14 shows the response of the system when there is a control system made simply of an angle of attack feedback gain.

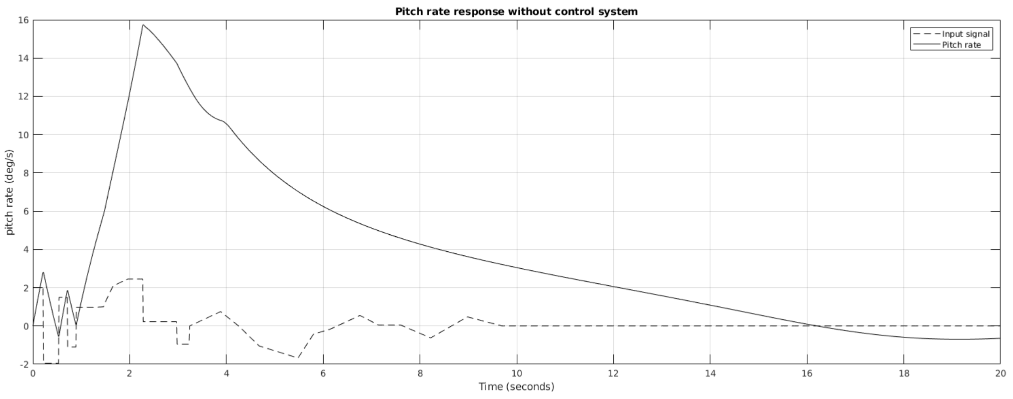


Fig. 14 Angle of attack feedback

Table 3 Eigenvalues of initial model

$-0.9883 + 0.0000i$
$0.1093 + 0.3696i$
$0.1093 - 0.3696i$

Fig. 15 shows the evolution of two of the controller gains computed online. As it can be seen, the changes in gains are sometimes abrupt and fast. Although this would not cause much issue to an automatic system (see Fig. 13), it would show up as unsatisfactory flying qualities when a human pilot flies the aircraft. The fast changes of control gains are due to fast changes in the identified model. There is therefore a need to introduce a filter to smooth out the gains of matrices A and B identified online. This problem is highlighted in the next section.

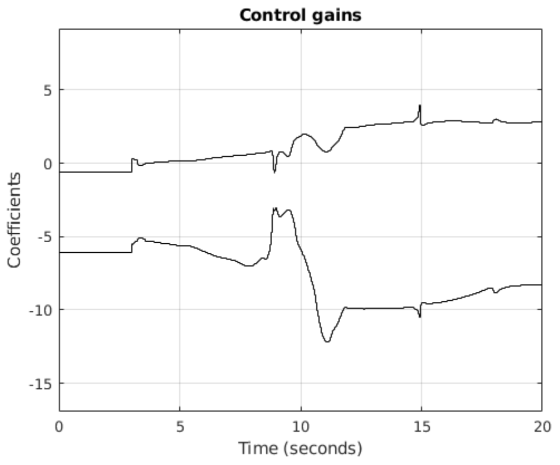


Fig. 15 LQR Controller gains evolution

VII. Dynamic inversion control

The linear dynamic inversion scheme shown in Fig. 2 was implemented on the supersonic passenger airplane model introduced in section V.B. The parameters of the control system were taken from matrices A and B identified in flight using the algorithm given in sub-section II.C. And this control scheme is used for the remainder of this paper.

The experimental evaluation of the algorithms was done in one of the ground based simulation of the Pilot-Vehicle Laboratory of Moscow Aviation Institute (see Fig. 16). Matlab and Simulink environments are used for the to run the airplane mathematical model. And in order to improve the performance and execution speed of the identification algorithm, it was compiled in C++ and used in Simulink via an S-function.

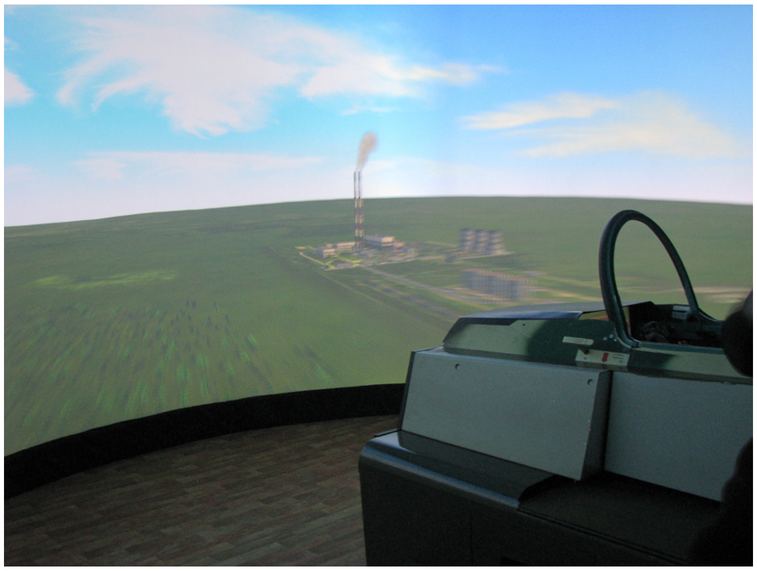


Fig. 16 Ground based simulator for experiments

The model was trimmed in an approach configuration with parameters given in table 4.

Table 4 Approach trim

h	V	α	γ
400 m	305.7 Km/h	14.36 deg	0 deg

The particularly high angle of attack is due to the fact that the slender delta wing design used in supersonic aircraft has a low lift curve slope. To achieve the required lift at low speed, the aircraft needs therefore to be flown at large angles of attack.

A pitch compensatory task was performed for a duration of 144 seconds and the identification algorithm run in real time in order to evaluate its feasibility. The change evolution of matrices A and B are shown in Fig. 17.

Empirical results seen in Figure 17 reveal the high sensitivity of the identification algorithm, which can be seen from the high rate of change of the parameters of matrices A and B. This noise can also be appear due to small inaccuracies

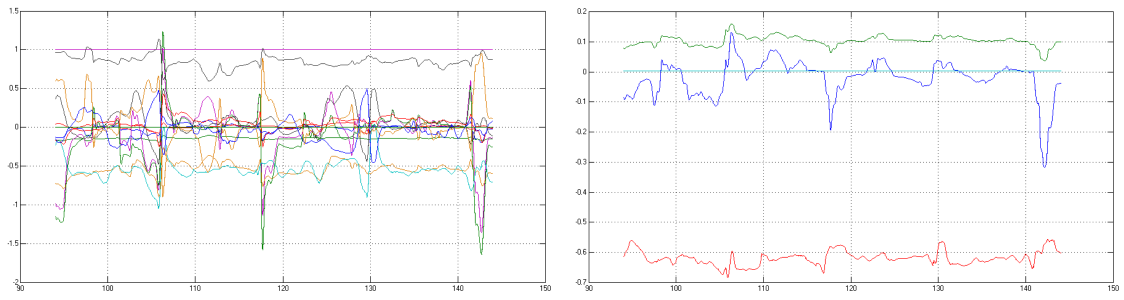

(a) Matrix A **(b) Matrix** B

Fig. 17 Matrices A **and** B **identified online**

in the estimation of the states derivatives $\dot{x}$ if these are not measurable. With the controller design depending on these parameters, the controller gains will also show a high variance. A possible solution to this is to introduce a filter to smooth out the identified parameters before synthesizing a control law.

A. Filtering

As shown in the previous section, the identification algorithm results in parameters that have some noise. To clean the noise, two filters were tested and compared.

1. Kalman filter

A one dimensional Kalman filter was implemented as shown in Fig. 18 with the following nomenclature and parameters:

- est - Estimate: output of the filter;
- $meas$ - Measurement: discrete input signal;
- e_{est} - Error in the estimate;
- e_{meas} - Error in the measurement;
- KG - Kalman gain.

To begin the iterations, a guess of the error in the measurement and an initial guess of the error in the estimates are given. And the first measurement is taken as the initial estimate.

Since dynamic inversion control is used, the filter is implemented only for the parameters that are used in the control scheme.

An experiment designed in the same way as the previous one was run, and figure 19 shows the results of the filter on the parameters. It is empirically evident that the filter converges to the true value of the parameters, and is not sensible to low frequency noise. The initial error in the estimate and error in the measurement parameters can be considered as the tunable parameters of the filter. These parameters were varied in order to study their effects on the sensitivity, but it

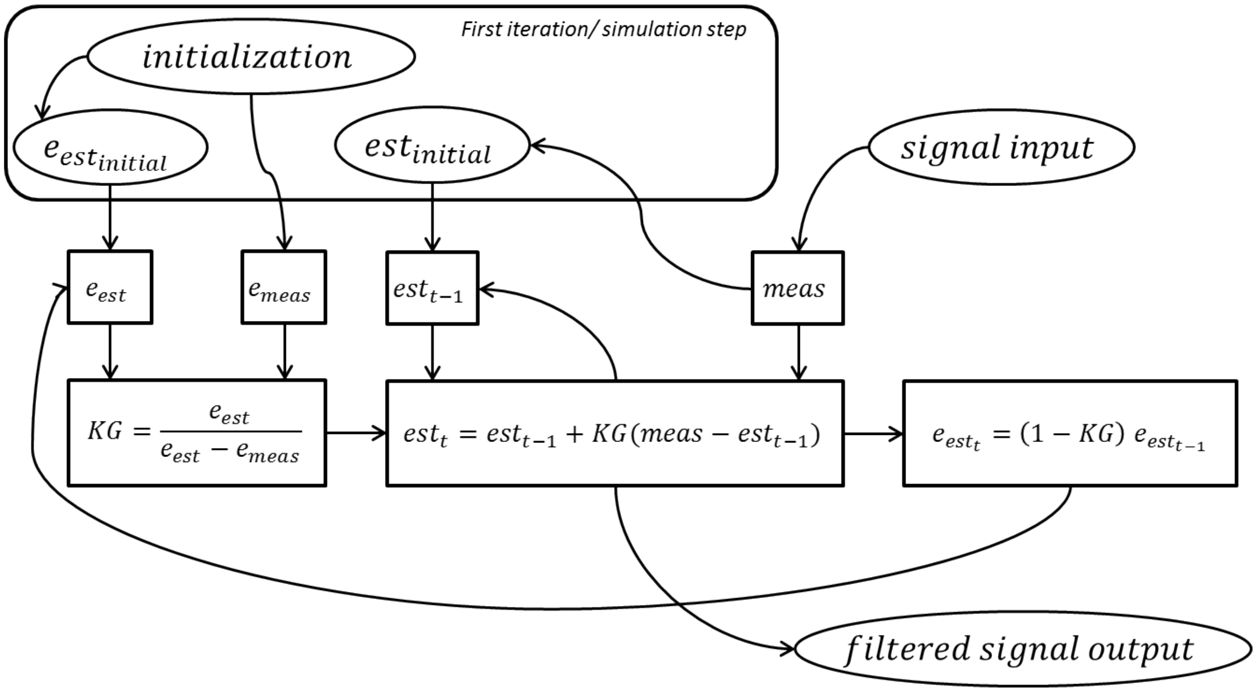

First iteration/ simulation step
initialization
$e_{est_{initial}}$
$est_{initial}$
signal input
e_{est}
e_{meas}
est_{t-1}
meas
$KG = \frac{e_{est}}{e_{est} - e_{meas}}$
$est_t = est_{t-1} + KG(meas - est_{t-1})$
$e_{est_t} = (1 - KG)\, e_{est_{t-1}}$
filtered signal output

Fig. 18 Kalman filter

was found that as long as the value of each parameter was small enough (< 20), they had little to no effect on sensitivity, and hence no effect on on the convergence rate. And using large values for these parameters resulted in a depreciation of the filter performance.

2. First order low pass filter

Besides the Kalman filter introduced in the previous subsection, a simple first order low pass filter of the shape shown in Eq. 20, with the discrete implementation shown in Eq. 21 was introduced.

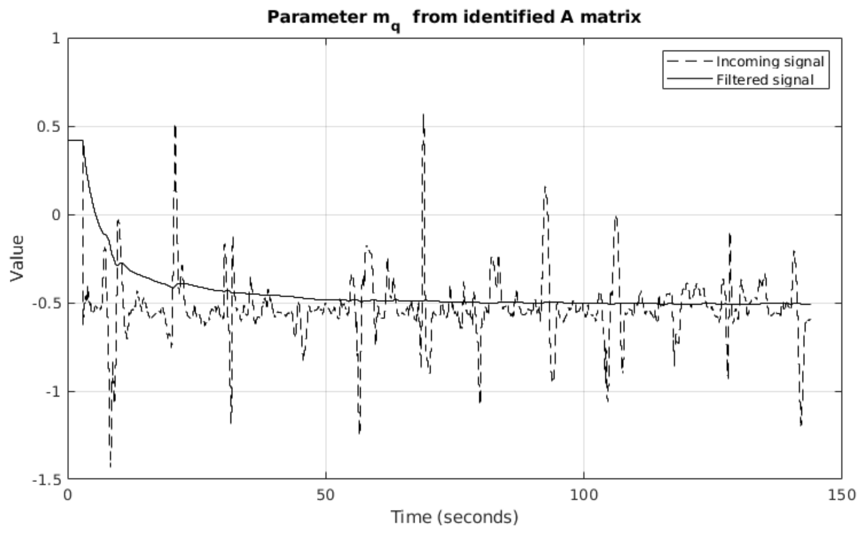
$$F = \frac{1}{T_c \cdot s + 1} \tag{20}$$

$$F_{discrete} = \frac{T_c \cdot T_s}{z - \left(T_c \frac{T_c - T_s}{T_c}\right)} \tag{21}$$

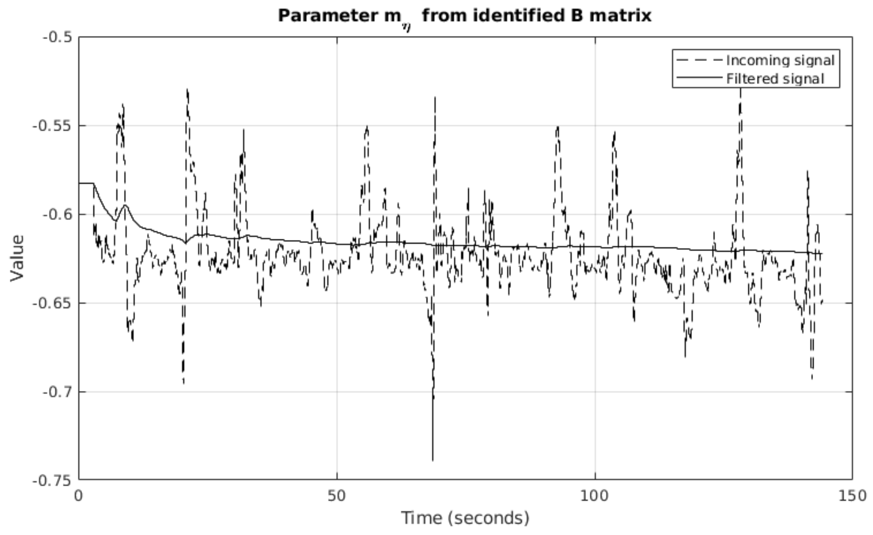
Where T_c is the time constant and T_s is the time step.

The time constant is chosen to be 5 seconds. This number was chosen experimentally. It is also worth noting that the filter is initialized with the first measurement (First measurement used as first output).

The filter was used for the same parameters tested in the previous subsection, and the results can be seen in Fig 20. As it can be seen, the well defined time constant allows to tune the sensitivity of the filter, and with the chosen time constant, the filter is sensitive to low frequency changes while filtering all the high frequency noise, theoretically



(a) Parameter from matrix ***A***



(b) Parameter from Matrix ***B***

Fig. 19 Kalman filter on identified parameters

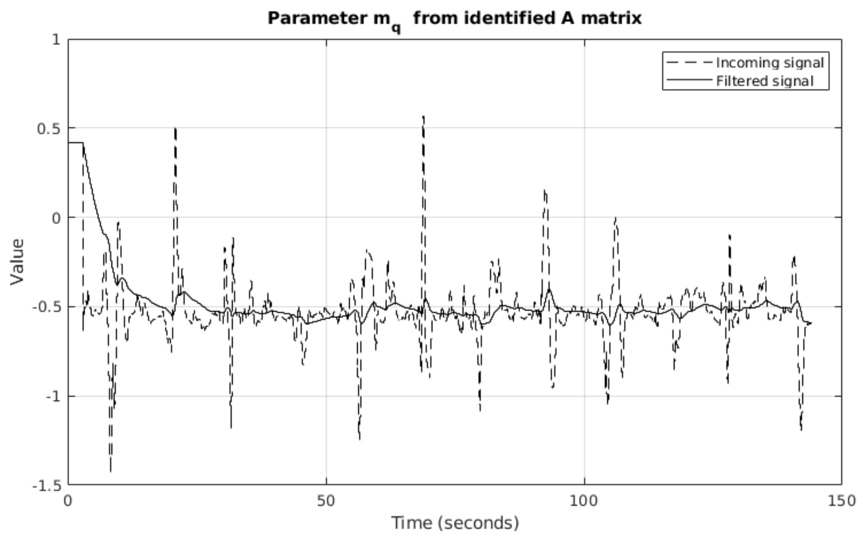
making its convergence rate faster than the previous filter.

3. Experimental comparison

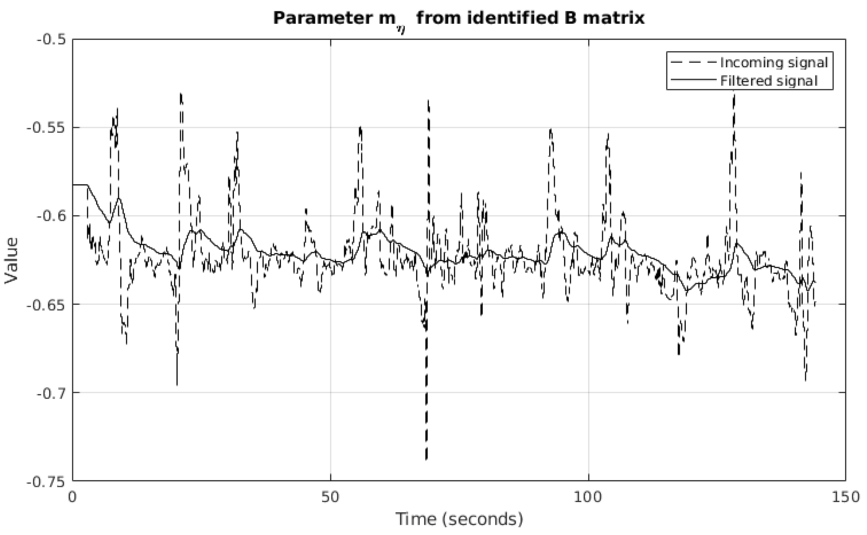
In order to learn about the convergence rate of the two filters proposed, the nonlinear airplane model was linearized in the approach stage and a pitch tracking task was performed.

The coefficients of matrix A used were increased by a 10 % and those of matrix B decreased by 10 %. This was done to simulate uncertainties of a fault that occurs in flight. The coefficients of the identified matrices are shown in Figs. 21 and 22.

Since the model used is a linear model, the identification algorithm performs without noise, and when a fault/



(a) Parameter from matrix ***A***



(b) Parameter from Matrix ***B***

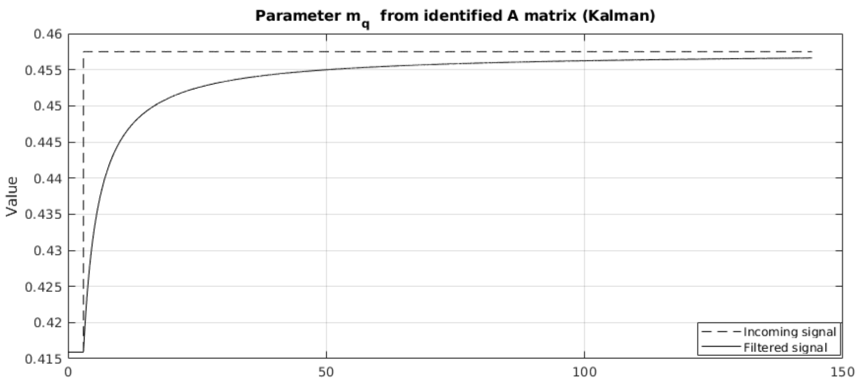
Fig. 20 First order low pass filter on identified parameters

uncertainty is detected, new matrices describing the new dynamics are computed.

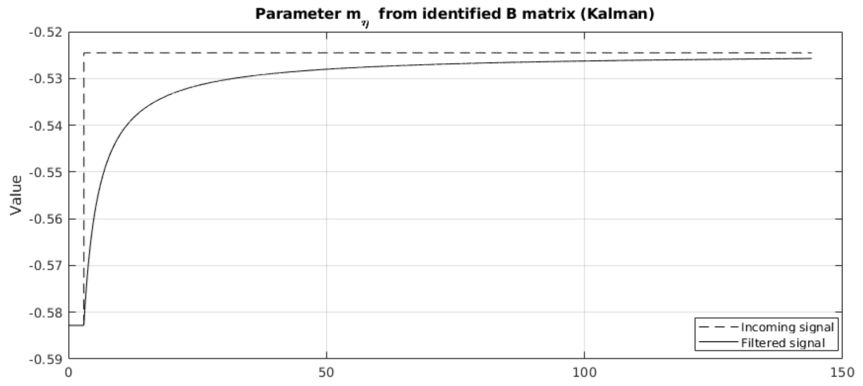
Both the low pass filter and Kalman filter allow a smooth transition from the previous dynamics to the newly detected dynamics. However, empirical evidence shows that the low pass filter has a much faster convergence rate, and as shown in the previous subsection, it shows good performance on noisy data.

VIII. Experimental evaluation

A pitch tracking task was performed on the dynamics of the airplane with the dynamic inversion based controller. For the tracking task, the experiment is designed as a single loop compensatory task (see Fig. 23).



(a) Parameter from matrix A



(b) Parameter from Matrix B

Fig. 21 Using Kalman filter

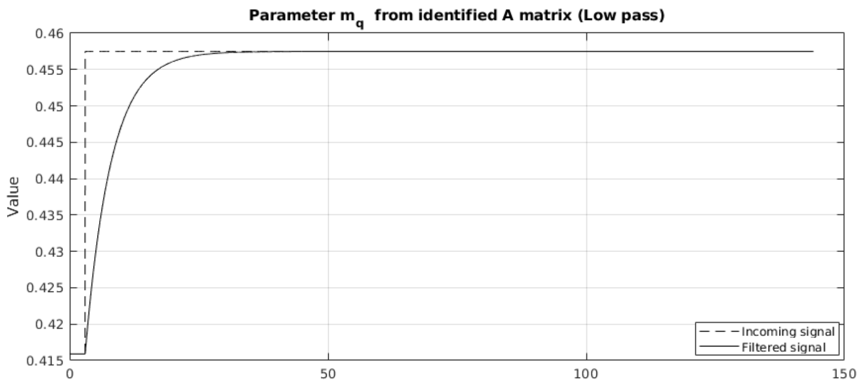
The pitch tracking task was carried out with a polyharmonic input signal represented by Eq. 22, which appeared as a random signal to the operator.

$$i(t) = \sum_{k=1}^{15} A_k \cos \omega_k t \tag{22}$$

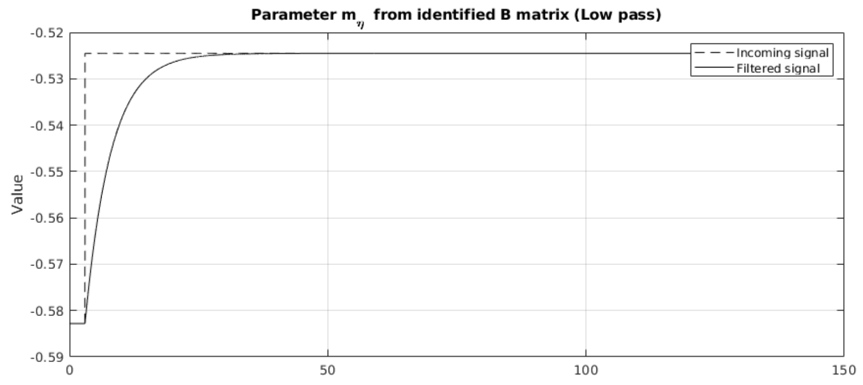
The amplitudes A_k and orthogonal frequencies $\omega_k = K\frac{2\pi}{T}$ where T is the duration of the experiment, were selected from the requirements of correspondence between the power distributions of the polyharmonic signal and a random signal characterized by the spectral density $\frac{K^2}{(\omega^2+0.5^2)^2}$. The Fourier coefficient technique described in [39] was used for the calculation of the main pilot-vehicle system characteristics.

Several sets of experiments were performed, with the following characteristics (At least three experiments were performed for each case, and the results were averaged):

1) **Case 1**: The dynamics of the airplane are entirely known (no faults, no uncertainties), and the dynamic inversion controller is designed with the nominal values of the dynamics. This case is used as a reference experiment.
2) **Case 2**: The nominal dynamics are used to compute the controller, but the actual dynamics is changed due to



(a) Parameter from matrix *A*



(b) Parameter from Matrix *B*

Fig. 22 Using first order low pass filter

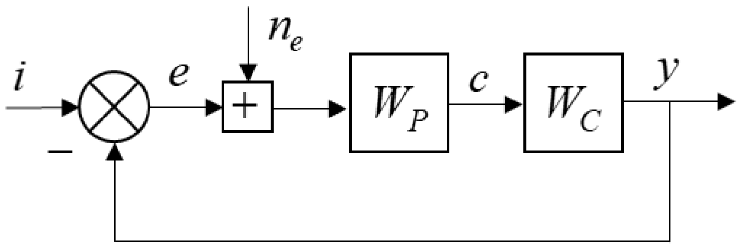


Fig. 23 Pilot-Aircraft system

uncertainties/ fault. The change is simulated by a 5 % decrease in the values of control effectiveness matrix B. The results of this experiment are shown compared to the reference experiment.

3) **Case 3**: All other experiments performed with a controller designed using the nominal dynamics, and where uncertainties/ fault in matrix B were more than 5 % of the nominal values and/ or uncertainties/ fault in the matrix A resulted in instabilities and inability of the operator to control the aircraft, and therefore the experiments could not be recorded

4) **Case 4**: The dynamic inversion controller is designed to use coefficients from matrices A and B identified in

flight. In this case, uncertainties/ faults were simulated by increasing the values of matrix A by a factor of 1.6 and decreasing values of matrix B by a factor of 0.4. This results in an uncertainty of 60 % in both matrices. The results of this experiment are shown compared to the reference experiment.

The results of the experiment cases described above are shown in Figs. 24 and 25, where σ_e^2 is the variance of error, W_C is the controlled object dynamics, and Φ is the pilot-aircraft closed loop dynamics.

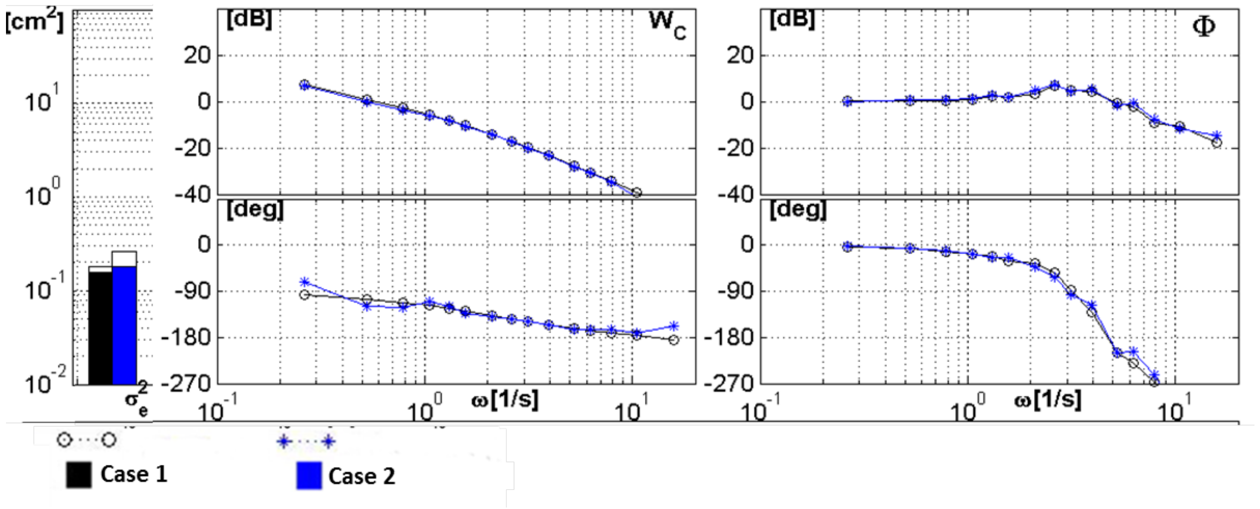


Fig. 24 Experiment Results: Case 1 vs Case 2

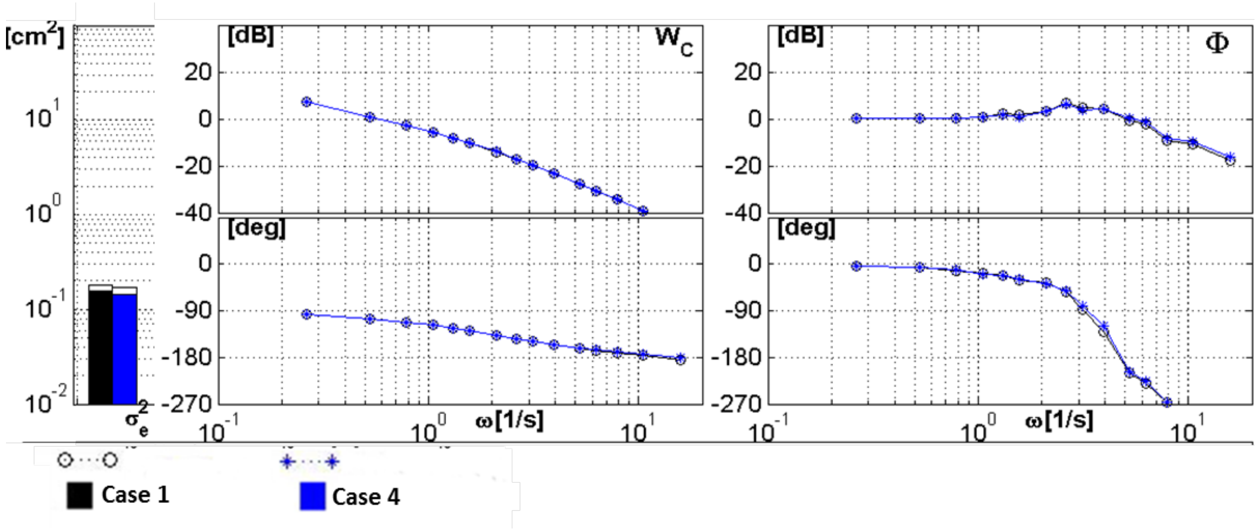


Fig. 25 Experiment Results: Case 1 vs Case 4

As it can be seen, when an uncertainty of 5% is added in the B matrix, the performance of the dynamic inversion controller starts decreasing. This change can be seen in the frequency response characteristics of the controlled object. And the performance can be evaluated from the variance of error. When uncertainties were more than 5 %, the flying qualities worsened considerably and the aircraft could be evaluated as uncontrollable, and the tracking task could not be completed.

On the other hand, when the online identification algorithm is activated, and the controller coefficients are directly computed in flight, the frequency response characteristics of the controlled object, and the pilot-aircraft closed loop system are practically the same (see Fig. 25) even though Case 4 has an uncertainty/ fault simulated as a 60 % change of the nominal values of the system matrices. This implies that the online identification algorithm detects the change in the dynamics, provides new coefficients for the controller, which in turn maintains the controlled element dynamics as desired, and the pilot evaluation of the aircraft/ flying qualities remain the same.

IX. Discussion and conclusion

This paper presents a framework in which an identification algorithm allows to approximate the system matrix A and control effectiveness matrix B in flight. This permits an indirect adaptive control approach in which the identified matrices are used to synthesize a control law, also in flight.

The identification algorithm, which is linear in nature, works on the basis of mean squared error minimization. Its effectiveness was shown using a linear mathematical model of a helicopter. To identify nonlinear system, an assumption was made, which is that nonlinear systems can be approximated by piecewise-linear systems. This assumption eliminated the possibility of using any recursive algorithm, as it would use all new data available to improve the approximation. Although this sounds advantageous, in reality it would not help, given the linear nature of the algebraic data structure used for the identification. This means a recursive method could well try to approximate a nonlinear system on a single linear data structure, boycotting the idea of piecewise linear systems.

To solve this issue, a replay buffer was introduced. The replay buffer has a limited memory and always stores the latest flight flight data. So, at each time step, the oldest data point in the buffer is removed, and the new data point added. Then the linear identification algorithm is used to approximate a system that would match the data existing in the replay buffer at each time step, or after the stationarity distance time has passed. This makes the identification recursive in nature, given that it is performed every time there is a new data point, but also allows the piecewise systems given the limited memory of the replay buffer. Therefore a system identified at time t could be totally different from the system identified at time $t + l$, where l is the memory size of the replay buffer, and yet, the systems identified in between times t and t and $t + l$ could be recursively correlated.

The stationarity distance is a concept introduced to quantify the amount of that during which we suppose the dynamics of the aircraft has not changed. For this entire paper, the stationarity distance was kept at 0.01 seconds, which is the same as the sampling time. Therefore the identification algorithm was run at each time step.

While evaluating the online identification algorithm, it was revealed that it can be sensitive to small changes and small inaccuracies in the calculation of the derivatives of the states, if the latter are not measurable. Therefore, the resulting matrices A and B could be noisy. Additionally, there could be cases when a fault occurs and the dynamics abruptly change. This will lead to an abrupt change in the control system, and might be unpleasant for the human pilot. To solve these two issues, two filters are introduced. A single variable implementation of the Kalman filter and a simple discrete low pass filter. Both filters were designed in such a way that the first measurement is used as the initial estimate/ state and output of the filters.

Experimental investigations showed the effectiveness of the filters and proved that the simple low pass filter was preferable given the direct access to the time constant of the filter, which controls the convergence rate, and sensitivity to low frequency noise.

Although the paper showed the effectiveness of the identification algorithm in presence of measurement noise, the fact remain that the algorithm depends on these measurements, and the subsequently synthesized controller depends on them as well. Therefore it is expected that the measurement task is solved with enough accuracy.

Another drawback is that, like any other nondeterministic approach, it is not possible to guarantee the performance of the framework presented. The only guarantee is that the identification algorithm will provide a dynamical system that minimizes the mean squared error given the data in the replay buffer (see section II.A). If the data present in the replay buffer was recorded during a long interval, where nonlinearities were present, the identification will give the best approximation of the nonlinear representative data, on a linear data structure. Therefore, even though the estimation will be optimal, it will not be correct. This speaks to the size of the replay buffer that should be chosen such that it is not too large, not too small. Future works could look into the possibility of establishing an algorithm for the optimal choice of the replay buffer size.

After the dynamics have been identified, a control law can be synthesized. To do this, two methods were tested. The first method, LQR, was chosen due to its importance in the control theory field. Computing an LQR based controller, however, is not easy given the fact that the continuous Riccati differential equation needs to be solved, which can be time consuming and not practical for real time purposes. Therefore an analytic solution to this equation, using the Hamiltonian theory of dynamical systems, was used. Even though this allowed a faster computation of the LQR solution, running both the identification algorithm and LQR algorithm in the Matlab/ Simulink software was time consuming and therefore could not run in real time. This was easy to solve, by simply compiling the algorithms in C++ and calling the compiled code in Simulink via S-functions.

The effectiveness of the LQR control algorithm was shown via computer simulations, and compared to a case when the controller is designed by taking the angle of attack in feedback.

The second control method, dynamic inversion, was chosen due to its simplicity and efficacy and was used for most of the experiments conducted. These experiments were conducted on one of the ground based flight simulators of the Pilot-Vehicle Laboratory at Moscow Aviation Institute.

The dynamic inversion controller was first evaluated for robustness experimentally, and proved infective at handling uncertainties of more than 5 % in the control effectiveness matrix, as the pilot-vehicle system became unstable, and the airplane uncontrollable.

Then it was shown that using the framework presented in this paper, the dynamics of the airplane could be identified online, and the control system coefficients could be effectively changed in order to provide the same performance characteristics to the uncertain/ faulty system as the nominal plant. And the fact that this identification is done on small time intervals allows the same control structure to be used on nonlinear dynamical systems, and the changes in the system due to nonlinearities would be caught by the piecewise identification algorithm. Therefore at each time step, the control algorithm controls the local linear approximation of the nonlinear system.

Acknowledgment

The paper is prepared in the framework of the Program for Development of a World-Class Research Center "Supersonic" in 2020-2025 funded by the Russian Ministry of Science and Higher Education (Agreement dated 16 Nov 2020 № 075-15-2020-924)

References

[1] Slotine, J.-J. E., Li, W., et al., *Applied nonlinear control*, Vol. 199, Prentice hall Englewood Cliffs, NJ, 1991.

[2] Charlet, B., Lévine, J., and Marino, R., "On dynamic feedback linearization," *Systems & Control Letters*, Vol. 13, No. 2, 1989, pp. 143–151. https://doi.org/https://doi.org/10.1016/0167-6911(89)90031-5, URL https://www.sciencedirect.com/science/article/pii/0167691189900315.

[3] Yeşildirek, A., and Lewis, F., "Feedback linearization using neural networks," *Automatica*, Vol. 31, No. 11, 1995, pp. 1659–1664. https://doi.org/https://doi.org/10.1016/0005-1098(95)00078-B, URL https://www.sciencedirect.com/science/article/pii/000510989500078B.

[4] Guillard, H., and Bourles, H., "Robust feedback linearization," *Proc. 14 th International Symposium on Mathematical Theory of Networks and Systems*, 2000.

[5] Ochi, Y., and Kanai, K., "Design of restructurable flight control systems using feedback linearization," *Journal of Guidance, Control, and Dynamics*, Vol. 14, No. 5, 1991, pp. 903–911.

[6] Bijnens, B., Chu, Q., Voorsluijs, G., and Mulder, J., "Adaptive feedback linearization flight control for a helicopter UAV," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2005, p. 6284.

[7] Kim, B. S., Calise, A., and Kam, M., "Nonlinear flight control using neural networks and feedback linearization," *Proceedings. The First IEEE Regional Conference on Aerospace Control Systems,*, IEEE, 1993, pp. 176–181.

[8] Miller, C., "Nonlinear dynamic inversion baseline control law: architecture and performance predictions," *AIAA Guidance, Navigation, and Control Conference*, 2011, p. 6467.

[9] Da Costa, R., Chu, Q., and Mulder, J., "Reentry flight controller design using nonlinear dynamic inversion," *Journal of Spacecraft and Rockets*, Vol. 40, No. 1, 2003, pp. 64–71.

[10] Wu, G., Meng, X., and Wang, F., "Improved nonlinear dynamic inversion control for a flexible air-breathing hypersonic vehicle," *Aerospace Science and Technology*, Vol. 78, 2018, pp. 734–743.

[11] Reiner, J., Balas, G. J., and Garrard, W. L., "Flight control design using robust dynamic inversion and time-scale separation," *Automatica*, Vol. 32, No. 11, 1996, pp. 1493–1504.

[12] Lee, H., Reiman, S., Dillon, C., and Youssef, H., "Robust nonlinear dynamic inversion control for a hypersonic cruise vehicle," *AIAA guidance, navigation and control conference and exhibit*, 2007, p. 6685.

[13] Hodel, A., Whorton, M., and Zhu, J., "Stability metrics for simulation and flight-software assessment and monitoring of adaptive control assist compensators," *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2008, p. 7005.

[14] Smith, P., "A simplified approach to nonlinear dynamic inversion based flight control," *23rd Atmospheric Flight Mechanics Conference*, 1998, p. 4461.

[15] Bacon, B. J., Ostroff, A. J., and Joshi, S. M., "Reconfigurable NDI controller using inertial sensor failure detection & isolation," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 37, No. 4, 2001, pp. 1373–1383.

[16] Lu, P., van Kampen, E.-J., de Visser, C., and Chu, Q., "Aircraft fault-tolerant trajectory control using Incremental Nonlinear Dynamic Inversion," *Control Engineering Practice*, Vol. 57, 2016, pp. 126–141.

[17] Simplício, P., Pavel, M., Van Kampen, E., and Chu, Q., "An acceleration measurements-based approach for helicopter nonlinear flight control using incremental nonlinear dynamic inversion," *Control Engineering Practice*, Vol. 21, No. 8, 2013, pp. 1065–1077.

[18] Nuebler, K., Lutz, T., Kraemer, E., Colliss, S., and Babinsky, H., "Shock control bump robustness enhancement," *50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, 2012, p. 46.

[19] Smeur, E. J., de Croon, G. C., and Chu, Q., "Gust disturbance alleviation with incremental nonlinear dynamic inversion," *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2016, pp. 5626–5631.

[20] Matamoros, I., and de Visser, C. C., "Incremental nonlinear control allocation for a tailless aircraft with innovative control effectors," *2018 AIAA Guidance, Navigation, and Control Conference*, 2018, p. 1116.

[21] Van Ekeren, W., Looye, G., Kuchar, R., Chu, Q., and Van Kampen, E. J., "Design, implementation and flight-test of incremental backstepping flight control laws," *AIAA Guidance, Navigation, and Control Conference, 2018*, American Institute of Aeronautics and Astronautics Inc.(AIAA), 2018, pp. AIAA–2018.

[22] Rugh, W. J., and Shamma, J. S., “Research on gain scheduling,” *Automatica*, Vol. 36, No. 10, 2000, pp. 1401–1425.

[23] Leith, D. J., and Leithead, W. E., “Survey of gain-scheduling analysis and design,” *International journal of control*, Vol. 73, No. 11, 2000, pp. 1001–1025.

[24] Shamma, J. S., and Athans, M., “Gain scheduling: Potential hazards and possible remedies,” *IEEE Control Systems Magazine*, Vol. 12, No. 3, 1992, pp. 101–107.

[25] Packard, A., “Gain scheduling via linear fractional transformations,” *Systems & control letters*, Vol. 22, No. 2, 1994, pp. 79–92.

[26] Nichols, R. A., Reichert, R. T., and Rugh, W. J., “Gain scheduling for H-infinity controllers: A flight control example,” *IEEE Transactions on Control systems technology*, Vol. 1, No. 2, 1993, pp. 69–79.

[27] Berstecher, R. G., Palm, R., and Unbehauen, H. D., “An adaptive fuzzy sliding-mode controller,” *IEEE Transactions on Industrial Electronics*, Vol. 48, No. 1, 2001, pp. 18–31.

[28] Wang, W., Yi, J., Zhao, D., and Liu, D., “Design of a stable sliding-mode controller for a class of second-order underactuated systems,” *IEE Proceedings-Control Theory and Applications*, Vol. 151, No. 6, 2004, pp. 683–690.

[29] Chen, Y.-P., and Lo, S.-C., “Sliding-mode controller design for spacecraft attitude tracking maneuvers,” *IEEE transactions on aerospace and electronic systems*, Vol. 29, No. 4, 1993, pp. 1328–1333.

[30] Takagi, T., and Sugeno, M., “Fuzzy identification of systems and its applications to modeling and control,” *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-15, No. 1, 1985, pp. 116–132. https://doi.org/10.1109/TSMC.1985.6313399.

[31] Herrnberger, M., and Lohmann, B., “Nonlinear Control Design for an Active Suspension using Velocity-Based Linearisations,” *IFAC Proceedings Volumes*, Vol. 43, No. 7, 2010, pp. 330–335. https://doi.org/https://doi.org/10.3182/20100712-3-DE-2013.00110, 6th IFAC Symposium on Advances in Automotive Control.

[32] Williams, G., “Overdetermined Systems of Linear Equations,” *The American Mathematical Monthly*, Vol. 97, No. 6, 1990, pp. 511–513. https://doi.org/10.1080/00029890.1990.11995638.

[33] Ben-Israel, A., “The moore of the moore-penrose inverse,” *The Electronic Journal of Linear Algebra*, Vol. 9, 2002.

[34] Courrieu, P., “Fast computation of Moore-Penrose inverse matrices,” *arXiv preprint arXiv:0804.4809*, 2008.

[35] Francis, B., “The optimal linear-quadratic time-invariant regulator with cheap control,” *IEEE Transactions on Automatic Control*, Vol. 24, No. 4, 1979, pp. 616–621. https://doi.org/10.1109/TAC.1979.1102097.

[36] Erokhin, P. V., Tikhonova, A. M., Urusova, O. A., and Shishkov, I. N., “Preliminary assessment of the possibility of creating a supersonic passenger airplane (IN RUSSIAN),” *XXX Scientific and technical conference on aerodynamics*, 2019.

[37] Ivanteeva, L. G., Zabrodin, R. V., Novikov, A. P., Novikov, M. P., Potapov, A., and Yudin, V. G., “A family of promising supersonic passenger airplane design with a low sound impact level (IN RUSSIAN),” *XXX Scientific and technical conference on aerodynamics*, 2019.

[38] Efremov, A. V., Efremov, E. V., Mbikayi, Z., Esaulov, S. Y., Ivchin, V. A., and Myasnikov, M. I., "Synthesis of a helicopter control system using inverse dynamics and its upgrade with the use of a sidestick controller," *46th European Rotorcraft Forum, ERF 2020*, 2020, pp. 9–17.

[39] Efremov, A. V., Rodchenko, V. V., and Boris, S., "Investigation of Pilot Induced Oscillation Tendency and Prediction Criteria Development." Tech. rep., MOSCOW INST OF AVIATION TECHNOLOGY (USSR), 1996.