JOURNAL OF AIRCRAFT
Vol. 47, No. 1, January–February 2010

Neural-Network-Based Flush Air Data Sensing System Demonstrated on a Mini Air Vehicle

Ihab Samy,* Ian Postlethwaite,† and Da-Wei Gu‡
University of Leicester, Leicester, England LE1 7RH, United Kingdom
and
John Green§
BlueBear Systems Research, Ltd., Bedford, England MK41 6JE, United Kingdom

DOI: 10.2514/1.44157

Flush air data sensing systems have been widely applied to large (manned) aircraft, where pressure orifices are typically located at the nosetip. This paper investigates the feasibility of a flush air data sensing system designed to estimate the air data states of a small unmanned air vehicle flown at speeds as low as Mach 0.07. Furthermore, due to the presence of a nose propeller, the pressure orifices are located at the wing leading edge. The motivation behind this project is the fact that traditional air data booms are physically impractical for small unmanned air vehicles. Overall, an 80 and 97% reduction in instrumentation weight and cost, respectively, were achieved. Both parametric and multilayer perceptron neural network models have been previously applied in the literature to model the aerodynamic relationship between aircraft surface pressure and the air data states. In this paper, an extended minimum resource allocating network radial basis function neural network is used as the flush air data sensing system model, due to its good generalization capabilities and compact structure. Computational fluid dynamic simulations are implemented to identify the ideal pressure port locations, and wind-tunnel tests are carried out to train and test the extended minimum resource allocating network radial basis function neural network.

Nomenclature

C_p = pressure coefficient
c = chord, m
$E1$ = neural network estimation error threshold
$E2$ = neural network rms estimation error threshold
$E3$ = neural network rms estimation error threshold
$N_{\max}$ = maximum number of hidden neurons
P_0 = total pressure, Pa
P_∞ = freestream static pressure, Pa
p_i = surface pressure, Pa
q_∞ = freestream dynamic pressure, Pa
T_∞ = freestream temperature, °C
t/c = thickness to chord ratio
V = local airspeed, m/s
V_∞ = freestream airspeed, m/s
x/c = normalized wing chord
α = angle of attack, deg
$\dot{\alpha}$ = angle of attack rate, deg /s
α_{eff} = local angle of attack, deg
β = sideslip, deg
β_{eff} = local angle of sideslip, deg
γ_{df} = decay factor for $E3$ threshold
δ = neural network learning rate
ε = calibration parameter
$\varepsilon_{\max}$ = initial $E3$ threshold
$\varepsilon_{\min}$ = final $E3$ threshold
$\boldsymbol{\theta}$ = vector of neural network free parameters
θ_i = flow incidence angle, deg
λ_i = angle the normal to the surface at port i makes with the longitudinal axis of the nosecap, deg
λ_n = neural network connecting weights
$\boldsymbol{\mu}$ = radial basis function center
σ = radial basis function width
ϕ_i = clockwise angle looking aft around the axis of symmetry starting at the bottom of the nosecap, deg
$\hat{}$ = indicates neural network estimates

Received 5 March 2009; revision received 21 July 2009; accepted for publication 31 July 2009.

*Ph.D. Student, Control and Instrumentation Department.
†Head of Control and Instrumentation Department.
‡Professor at the Control and Instrumentation Department.
§Senior Engineer.

I. Introduction

TRADITIONALLY, critical air data are measured using air data booms protruding from the aircraft local flowfield; freestream static pressure P_∞ and total pressure P_0 are measured using a pitot-static tube while angle of attack α and angle of sideslip β are measured using small vanes mounted on the air data boom. Using these four basic air data quantities (P_∞, P_0, α, β), as well as temperature T_∞, most other air data of interest can be directly calculated, such as airspeed, altitude, and rate of climb. In this paper, we are only interested in the critical air data: P_∞, P_0, α, β. Different designs and applications may exist, however, the basic air data boom remains one of the most popular methods for such air data measurements [1].

Despite their popularity, air data booms are known to have measurement disadvantages in addition to possible malfunctions: accuracy may be adversely affected by boom bending and vibration, probe size, and geometry, and by the flow interference due to the probe itself. Furthermore, in military-related applications, external instrumentation is undesirable in stealth vehicles. As a result, in recent years more research has been carried out to find alternative solutions to air data booms. One example is an optical air data system, which measures the atmosphere outside of an air vehicle and provides information regarding the environment ahead of the flight vehicle [2]. These systems are very accurate and, more important, are not affected by weather external to aircraft such as icing or plugging. However, with the primary goal of most air vehicle manufacturers being the reduction of costs, researchers found the concept of air data measurements using a matrix of pressure orifices (ports) to be a cheaper alternative to optical systems and air data booms.

The measurement of flush surface pressures to estimate air data parameters has been known for some time and is referred to as a flush air data sensing (FADS) system. The first FADS system was

developed and tested on the NASA X-15 hypersonic aircraft [3,4]. It consisted of a hemispherical nose (mounted with four pressure ports) which was steered into the wind vector to measure the air data. Results were promising, however, the concept of the steered nose was considered too complex. Consequently, over the years the FADS system experienced many modifications and successful applications. Most aeronautical applications of the FADS system originate from the initial tests carried out by NASA in the early 1980s. Examples include [5–7]. Recently, the FADS system was implemented on the NASA Dryden F-18 Systems Research Aircraft [8]. This system uses 11 pressure ports in the radome of the aircraft and was tested at speeds up to Mach 1.6, α up to 80 deg, and $\beta = \pm 20$ deg. Other applications of the FADS system published in AIAA journals include [9–22].

From the literature, we will find that few examples have been extended to mini (unmanned) air vehicles (MAVs). This motivated the investigation of such an application. MAVs are found within the spectrum of unmanned air vehicles (UAVs) and are characterized by their small size and low weight. Several military missions have taken advantage of this feature to use them in applications which can only be attained at great risks. The FADS system is an invaluable alternative to air data booms *especially* for MAV applications. This is because current air data booms can be too heavy and expensive for use on a MAV. Additionally, due to the dangerous and secretive environments that they can be exposed to, external instrumentation is best avoided.

Most applications tend to mount the FADS system near the aircraft nosetip mainly for two reasons. First, the aerodynamic model relating the surface pressure and air data states is derived around a blunt body, and so is most valid at the nosecap which can be approximated as a sphere. Second, the nosetip has been used traditionally as the air data measurement location for air data booms. Unfortunately, many MAVs (as it is in our case) are driven by a nose propeller which can obstruct the FADS system. Therefore as an alternative we consider placing the FADS system at the wing leading edge. In fact, in [14], it was reported that the aerodynamic model developed for a FADS system mounted at the spherical nosecap is equally applicable to the wing leading edge.

In this paper, a computational fluid dynamics (CFD) analysis of the MAV is carried out to identify suitable locations for the pressure ports, and a wind-tunnel test is carried out to investigate the feasibility of estimating three air data states (P_∞, V_∞, α) from a FADS system mounted on a MAV wing. The possibility of estimating the sideslip β from the current FADS system configuration will also be discussed in Sec. VI. Our work is distinct from previous studies in that 1) a FADS system is designed and implemented on the wing of a MAV which flies at speeds as low as Mach 0.07 and 2) an extended minimum resource allocating network (EMRAN) radial basis function (RBF) neural network (NN) is configured to model the aerodynamic relationships in the FADS system. This paper is organized as follows. In Sec. II, the FADS system and MAV are introduced. Section III defines the NN and training algorithm used in the FADS system. Section IV shows the CFD results and concludes the locations of the pressure ports. Section V is the results section, followed by the conclusions section.

II. Flush Air Data Sensing System Applied to a Mini Air Vehicle

A. Mini Air Vehicle

The MAV used to test the FADS system is shown in Fig. 1. The MAV uses an MH64 wing section[¶] and flies at a maximum speed of 20 m/s. From Fig. 1, we can see that the MAV is driven by a nose propeller and therefore mounting a FADS system at the wing leading edge is more suitable than the nose. Furthermore, the wing leading edge cannot be approximated as a sphere and therefore a NN modeling approach in the FADS system is more appropriate than the aerodynamic model (which is derived based on spherical shape assumptions) to be defined in Sec. II.B. Some of the MAV properties

[¶]MH64 airfoil data available at http://www.mh-aerotools.de/airfoils/mh64koo.htm.

Fig. 1 MAV (courtesy of BlueBear Systems Research, Ltd.).

are shown in Table 1. However, note that only the wing section (Fig. 2) is used in the wind-tunnel and CFD tests.

B. Flush Air Data Sensing System Model

The relationship between the aircraft surface pressure and the air data (P_∞, P_0, α, β) is known as the FADS system model or the air data model. The standard air data model used in the FADS system is defined in many parts of the literature [1,8,13–15,22]. It can be derived based on three airflow assumptions: irrotational (potential) flow, incompressible flow, and airflow over a blunt body (e.g., sphere). Irrotational flow assumes that air simply translates over the body with no rotation, that is, with zero angular velocity, whereas incompressible flow assumes that the air density is constant everywhere in the atmosphere. Blunt bodies have the property that most of the aerodynamic drag is caused by perpendicular forces instead of tangential forces. The blunt body assumed in the air data model is a sphere, as the pressure ports are generally located at the nosecap of the aircraft which has an almost spherical shape. Together these assumptions greatly simplify the air data model [13].

The air data model can be defined as [13]

$$p_i(\theta_i) = q_\infty[\cos^2\theta_i + \varepsilon\sin^2\theta_i] + P_\infty \tag{1}$$

where p_i is the surface pressure measured at port i, θ_i is the flow incidence angle between the surface normal at the ith port and the airspeed vector, q_∞ is the freestream dynamic pressure, and ε is a calibration parameter which is empirically determined to account for the assumptions mentioned earlier (e.g., incompressible flow). The angle θ_i can be defined in terms of the local angle of attack and sideslip (α_{eff}, β_{eff}) as follows [13]:

$$\begin{aligned}\cos(\theta_i) = {} & \cos(\alpha_{\text{eff}})\cos(\beta_{\text{eff}})\cos(\lambda_i) + \sin(\beta_{\text{eff}})\sin(\phi_i)\sin(\lambda_i) \\ & + \sin(\alpha_{\text{eff}})\cos(\beta_{\text{eff}})\cos(\phi_i)\sin(\lambda_i)\end{aligned} \tag{2}$$

where λ_i is the angle the normal to the surface at port i (i.e., the line perpendicular to the tangent of port i) makes with the longitudinal axis of the nosecap, and the clock angle ϕ_i is the clockwise angle looking aft around the axis of symmetry starting at the bottom of the nosecap. Combining Eqs. (1) and (2) we can define the overall air data model relating a vector of surface pressure measurements $\mathbf{p_i}$, to the air data states as

Table 1 Some of the MAV characteristics

Characteristic	Value
Speed range	8–20 m/s
Mass	450 g
Wing span	488 mm
Wing root chord	250 mm
Wing tip chord	200 mm
Wing thickness, t/c	8.61%

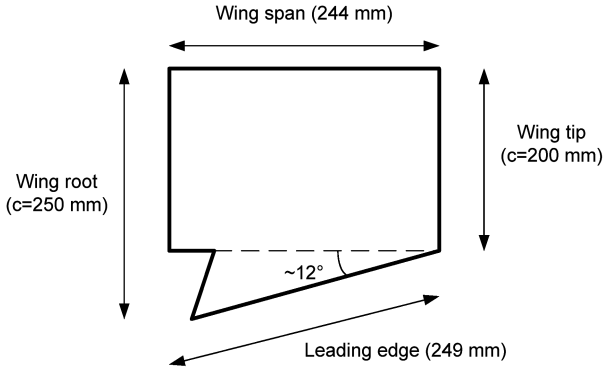


Fig. 2 Top view of the wing section (*c* is the wing chord).

$$\mathbf{p}_i(\phi_i, \lambda_i) = \boldsymbol{f}_i(P_\infty, P_0, \alpha_{\text{eff}}, \beta_{\text{eff}}, \varepsilon, \phi_i, \lambda_i) \tag{3}$$

where $i = 1, 2, \ldots, N$ and N is the total number of pressure ports in the FADS system.

Equation (3) is quite complex and can be difficult to solve. Furthermore, α_{eff} and β_{eff} are local flow angles influenced by the aircraft-induced wash [1]. Therefore, they must be calibrated so that the true freestream aerodynamic orientation (α, β) is calculated. To further complicate Eq. (3), the calibration parameter ε is itself dependant on the air data (e.g., ε would change with airspeed). There are two popular methods, cited in the literature, to solve Eq. (3): nonlinear regression methods (see, e.g., [8,22]) and the triples algorithm (see, e.g., [13]). The nonlinear regression approach substitutes the measured $\mathbf{p_i}$ into Eq. (3) and air data states are then estimated using a nonlinear least-squares regression algorithm. This method has known to suffer from stability problems (i.e., solutions can often diverge) but, more important, can be computationally demanding due to its iterative approach. In fact, the same authors who proposed this method for solving Eq. (3) acknowledged that it was too risky to be used in real time and was subsequently abandoned [13]. As an alternative, however, they proposed the triples algorithm [13]. The triples algorithm is a clever way to solve Eq. (3). It strategically selects three pressure ports to decouple α_{eff}, β_{eff} from P_∞, P_0, ε. Simpler approaches to modeling Eq. (3) are via lookup tables, where calibration data (from either wind-tunnels tests or real flight) are used to build the lookup table. This is the simplest of methods but can also suffer high memory usage and slow execution times.

As an alternative, we investigate the feasibility of using NNs to model the air data system model. NNs provide a means for modeling linear or nonlinear systems without the need of detailed system knowledge. They primarily rely on sufficient training data from which they can develop their structure. This makes them an attractive modeling solution to applications where the theory is poor but the relevant data are plentiful. An additional advantage of NNs is that they can be implemented in just a few lines of code, which is suitable in MAV applications where computational power may be limited. The two most popular NN architectures are the multilayer perceptron (MLP) and the RBF NN [23]. MLP NNs trained with error backpropagation algorithms have been successfully implemented in a FADS system by several authors [1,11,12,18,20]. Instead, we consider an EMRAN RBF NN due to its good generalization capabilities and compact structure.

III. Neural Network Model

A. Extended Minimum Resource Allocating Network Radial Basis Function Neural Network

Over the past years, there has been a particular interest in RBF NNs due to their good generalization performance [24,25]. A typical two-layered RBF NN consisting of the inputs, a hidden layer, and an output layer is shown in Fig. 3. The input units simply transfer the system input vector to the hidden units which form a localized response to the input pattern. The hidden layer includes the nonlinear radial basis functions. Theoretical analysis and practical results have

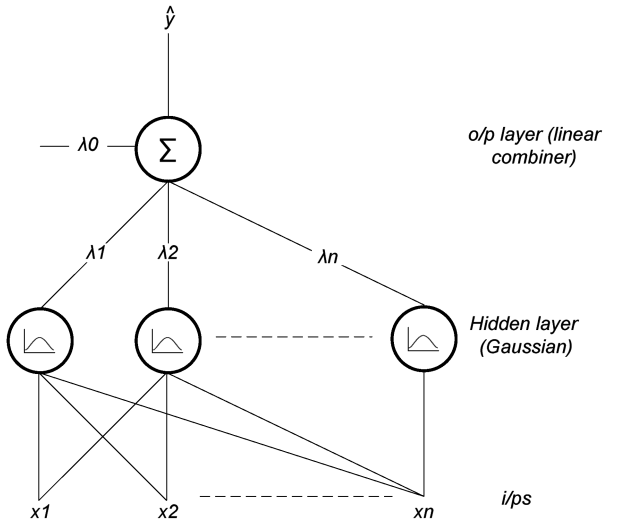


Fig. 3 Fully connected RBF NN; x, λ, and $\hat{y}$ are inputs, weights, and output, respectively.

shown that the choice of nonlinearity is generally not crucial to the RBF NN performance [26]. Gaussian functions are typically used. The output layer then performs a simple linear combination of the hidden layer outputs.

A single output RBF NN with N hidden neurons can be expressed as follows:

$$\hat{y} = \lambda_0 + \sum_{n=1}^{N} \lambda_n \exp\left[\frac{-\|\mathbf{x} - \boldsymbol{\mu}_n\|^2}{\sigma_n^2}\right] \tag{4}$$

where $\hat{y}$ is the NN estimate of target y, $\mathbf{x}$ is the input vector, λ_n are the individual weights found between the hidden and output layer, $\boldsymbol{\mu}$ and σ are the centers and widths of the Gaussian functions (hidden neurons), respectively, and $\|\cdot\|$ is the Euclidean norm. In the conventional implementation of the RBF NN, N in Eq. (4) is fixed and assumed to be known a priori. The training algorithm is then used to update only the weights of the RBF NN, that is, the centers and widths of the Gaussian functions are fixed. This approach is appealing, as well-established linear optimization methods such as the least mean square (LMS) algorithm could be used to update the linearly connected weights.

However, fixing the centers and widths of the hidden neurons has shown to result in large NN structures (as in principle we would require one hidden neuron for each input pattern). This dimensionality problem can significantly increase the NN processing time [26]. Moreover, heuristically selecting a suitable number of hidden neurons as well as their respective centers and widths can be a challenging task. To overcome the drawbacks of conventional RBF NNs, a more advanced network was developed by Li et al. [25] known as the EMRAN RBF NN, which is in fact an extension to the resource-allocating network RBF NN originally proposed by Platt in 1991 [27].

The EMRAN RBF NN proposed in [25] consists of a group of input signals, a hidden layer, and an output layer (Fig. 3). It starts with zero hidden units (neurons) and only adds hidden units if *all* of the following three criteria are met (for a single output NN):

$$e_k = y_k - \hat{y}_k > E1 \tag{5}$$

$$e_{\text{rms}k} = \sqrt{\sum_{j=k-(M-1)}^{k} \frac{e_j^2}{M}} > E2 \tag{6}$$

$$d_k = \|\mathbf{x_k} - \boldsymbol{\mu}_{\mathbf{rk}}\| > E3 \tag{7}$$

where $\boldsymbol{\mu}_{\mathbf{rk}}$ is the center of the hidden neuron closest to the input vector $\mathbf{x_k}$. $E1$, $E2$, and $E3$ are fixed thresholds and $E3 = \max\{\varepsilon_{\max},$

$\gamma_{\mathrm{df}}^{k}, \varepsilon_{\min}\}$ where γ_{df} is a positive decay constant ($0 < \gamma_{\mathrm{df}} < 1$). $E1$ and $E2$ ensure that the estimation error and the root mean square (rms) estimation error for the past M samples are below a predefined threshold, that is, they check if the NN estimates are sufficiently accurate. $E3$ checks if the minimum distance between the current input vector and the centers of the hidden neurons is significantly small (i.e., so that there is at least one hidden neuron which is sensitive to the current input vector pattern).

If less than three of the criteria in Eqs. (5–7) are met, then the NN training algorithm updates (i.e., tunes) the network parameters (centers, widths, and weights) of *only* the most active (so-called winner) neuron. This reduces the number of parameters to be updated, which speeds up the training process. An additional reason why the EMRAN RBF NN is superior to other NN models is that it can automatically remove (prune) hidden units that contribute the least to the NN estimates. This maintains a minimum structure size [25]. However, this is only implemented once the maximum number of hidden neurons allowed ($N_{\max}$, defined by the user) is reached. The process of pruning is as follows. If all three criteria in Eqs. (5–7) are met and $N = N_{\max}$, then the EMRAN algorithm adds a new hidden neuron in place of the hidden neuron with the lowest activation (i.e., lowest output).

B. Neural Network Training Algorithm

The EMRAN algorithm is a set of conditions which decide how/when the RBF NN structure should be further adapted to better suit the training data. However, we have yet to define a training algorithm to update (only when directed by the EMRAN algorithm) the NN free parameters. Good characteristics of the chosen training algorithm include faster execution times, lower estimation errors, and a compact structure. Most NN training algorithms are based on gradient descent optimization methods where the NN free parameters are updated in a way that minimizes the output estimation error. For example, the standard LMS algorithm is typically used in the conventional RBF NN to update the linearly connected weights. This technique would benefit from linear optimization but, as we have mentioned earlier, the NN can sometimes suffer from the curse of dimensionality. Other popular examples are the error back-propagation algorithm (BPA) and its extension, the extended BPA which are both used to train the famous MLP NN. However, in this case, as the MLP NN has nonlinear neurons in its output layer and hidden layer, the process of training is a nonlinear optimization problem and, as such, there are risks of being trapped in local minima. There have also been examples where an extended Kalman filter is used to update the NN free parameters [28].

The gradient descent training algorithm typically used to train the EMRAN RBF NN is as follows [29]:

$$\boldsymbol{\theta}_{\mathbf{k+1}} = \boldsymbol{\theta}_{\mathbf{k}} + \Delta\boldsymbol{\theta}_{\mathbf{k}} \tag{8}$$

$$\Delta\boldsymbol{\theta}_{\mathbf{k}} = -\delta \left.\frac{\partial E}{\partial \boldsymbol{\theta}}\right|_{k} \tag{9}$$

where $\boldsymbol{\theta}$ is the vector of free parameters (made up of centers, widths, and weights), δ is the learning rate, and E is the instantaneous squared error (cost function):

$$E = \frac{1}{2}e^2 \tag{10}$$

where e is the NN estimation error as in Eq. (5). During each iteration, the NN free parameters are adjusted in a direction opposite to the gradient in Eq. (9), that is, toward the minimum squared error. The hidden layer of the EMRAN RBF NN consists of nonlinear neurons (Gaussian functions) and therefore the gradient descent training algorithm in Eq. (8) is a nonlinear optimization problem, that is, we run the risk of being trapped in local minima. As such, the NNs used in the FADS system must be rigorously tested and the parameters ($E1$, $E2$, $\varepsilon_{\max}$, $\varepsilon_{\min}$, γ_{df}) in Eqs. (5–10) must be carefully tuned until we achieve acceptable performance (e.g., low estimation errors and fast execution times). This is also the case when choosing the NN learning rate δ, as a high learning rate guarantees good estimations but it also degrades the global approximation capability of the NN.

C. Method of Neural Network Training and Validation

Before implementing the NN in the FADS system, the network structure must be trained and validated. The NN used in the FADS system is trained based on the estimation error e in Eq. (10), that is, the free parameters are updated in such a way that reduces the estimation error. Therefore, the desired (i.e., target) NN outputs must be defined a priori so that the estimation error e can be calculated at each time step. This method of training is more formally known as supervised training. Supervised training can be either implemented online or offline. In offline training, the NN is trained and, once satisfactory performance is achieved (e.g., by judging the accuracy and generalization capability of the NN), the NN structure is frozen and used onboard the aircraft with no further training. On the other hand, online training continuously updates the NN structure in real time and is generally preferable in time-varying systems. However, online training can also be computationally slow. Moreover, it requires additional instrumentation onboard the aircraft to measure the target outputs, which can be inadequate in applications where instrumentation costs are to be reduced, as it is in our case.

The method used to train the NNs in the FADS system is offline training. In our case, the purpose of the FADS system is to provide an alternative way of measuring the air data in a MAV, that is, to avoid using an air data boom. To be able to train the NN online (i.e., onboard the aircraft), we would require an extra air data sensor (such as an air data boom) to provide the target outputs. This therefore contradicts the purpose of the FADS system. As such, the NN is trained offline and, once predefined stopping criteria are satisfied, the NN structure is frozen and used onboard the MAV with no further training.

A common problem with NNs is that of overfitting (overtraining) the training data. In this case, the NN estimations are accurate during the training process but are poor when exposed to "new" data, that is, the NN generalization capabilities are poor. To overcome this problem, it is common practice to query/validate the NN using a testing data set as it is trained. NN training can then be stopped once the test set error starts to increase (as this indicates that the NN has been overtrained). In our case, the offline training stopping criteria chosen are based on checking NN convergence as well as avoiding the overfitting phenomenon. Two data sets are used to do so:

1) The train data set is used to train the NN with learning switched *on*.

2) The test data set is used to query/validate the NN with learning switched *off*.

NN offline training is then stopped based on two criteria:

1) Criterion 1: If the rms estimation error of the test data set increases for more than 100 consecutive epochs, offline training is stopped.

2) Criterion 2: If the rate of change of rms per epoch Δrms is less than 0.1% for more than 100 consecutive epochs for both the test and train data sets, offline training is stopped.

One epoch represents one pass through the whole data set, criterion 1 is the overfitting criterion and criterion 2 checks if the NN has converged. Note that the offline training stage builds the final NN structure to be used onboard the aircraft. The NN training and validation results will be shown in Sec. V.D.

IV. Computational Fluid Dynamics Results and Conclusions

A. Computational Fluid Dynamics Results

CFD simulations were implemented on the MAV wing using the MH64-wing coordinates defined in.¶ A CFD analysis of the pressure distribution over a wing can help us identify which parts of the wing would be most suitable for mounting the pressure ports in the FADS system. Note that the aim of a FADS system is to convert surface pressure to air data states, and so it is important that the pressure ports

are mounted in areas which are highly sensitive to the air data (P_∞, V_∞, α).

Two-dimensional CFD simulations of an aerofoil can have some limitations. For example, we are unable to investigate the aerodynamic properties vs sideslip. In more advanced applications, a 2-D aerofoil would be unsuitable if the wing tip vortices are of interest to the engineers. Therefore, it is expected that numerical uncertainties would exist with respect to the meshed model and the pressure distributions over the aerofoil. However, in our case, we are only interested in *approximating* the pressure distributions to locate the steep pressure gradients at the wing leading edge. Therefore, as we are not interested in simulating the exact air flow over the wing and calculating the exact pressure magnitudes, a 2-D CFD simulation is sufficient and, more important, strict convergence criteria ($1e-4$ accuracy is used) are not necessary.

The air flow simulated over an aerofoil, and indeed the whole aircraft, can be categorized according to its viscosity and compressibility [30]. Air flowing past an aerofoil is essentially viscous. Viscosity effects are caused by the friction between the air and the aerofoil, the thermal conduction between areas of high and low temperature, and the mass diffusion when fluid concentration gradients are present [30]. Air is also compressible, which implies that it would consistently change its density ρ as it expands and contracts throughout the flow. However, for most low-speed flights (speeds less than $\approx$100 m/s), it can be assumed that the air flow is inviscid and incompressible [30]. This is mathematically convenient and can simplify the CFD simulation. In theory, however, such a flow does not exist as the frictional forces of the viscous air flow are a major contribution to the aerodynamic drag. However, as we are 1) assuming low-speed flight and 2) concerned with identifying the location of the steep pressure gradients on the wing and not as interested in replicating the MAV flight in CFD simulation, an inviscid, incompressible air flow is sufficient for our task. Furthermore, as we are not using an accurate mathematical representation to express the relationship between the air data states and pressure data, it is more important that the pressure data measured varies with air data to train the NN than it is to accurately tune and predict the pressure data in CFD simulation.

The software package Gambit is used to build the MH64 aerofoil, which is later analyzed in the CFD software Fluent. Figure 4 shows the aerofoil built in Gambit. The quadrilateral "cells" are generated during the meshing stage. Increasing the number of cells can increase the accuracy of the CFD simulations but it can also increase the processing time. In our case, the domain around the aerofoil was divided into 12,240 cells (Fig. 4).

The aerofoil designed in Gambit can then be uploaded into Fluent. In our case, the air flow was assumed to be inviscid and incompressible. Standard sea-level atmospheric conditions were also considered and pressures were referenced to the atmospheric pressure (101.325 kPa). The convergence criterion was set to $1e-5$ for all CFD simulations, and tests were carried out at $V_\infty = 12$, 15, 18, and 20 m/s and $\alpha = 0$, 2, 4, 6, and 8 deg.

In our case, we are interested in locating the pressure gradients on the aerofoil surface. For this purpose, we require an x-y plot of aerofoil surface pressure vs chord. The C_p plots are displayed in Figs. 5–8 where C_p can be expressed as

$$C_p = \frac{p - p_\infty}{q_\infty} = 1 - \left(\frac{V}{V_\infty}\right)^2 \tag{11}$$

where p is the local pressure and V is the local airspeed. The pressure ports of the FADS system must be mounted on the wing in areas where 1) there are steep pressure gradients, and 2) there are strong variations between the upper and lower surface pressures.

This offers a wealth of nonredundant information from which the air data values may be estimated. From Fig. 5, we can see that criteria 1 and 2 are satisfied for $x/c < 0.1$. If we zoom into Fig. 5 (Fig. 6), we can conclude that most of the "action" occurs at $x/c < 0.01$ and C_p is almost constant for $x/c > 0.01$. Therefore, initially we can conclude that the pressure ports must be located at $x/c < 0.01$. We must next consider the variation of pressure with V_∞. However, in this case, C_p would not change for a change in V_∞. This is because the ratio V/V_∞ in Eq. (11) would remain the same and therefore C_p would be constant. Instead, we consider the static pressure on the surface of the aerofoil (Figs. 7 and 8). Once again, from Fig. 8, we conclude that the steep pressure gradients occur at $x/c < 0.01$.

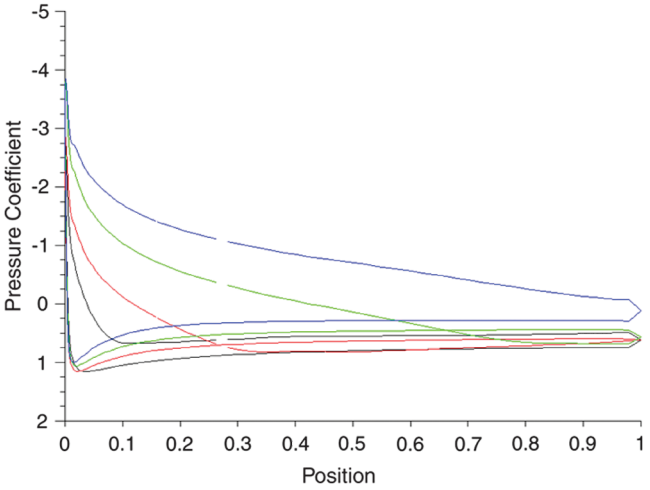


Fig. 5 C_p plot for $V_\infty = 20$ m/s, $\alpha = 2, 4, 6, 8$ deg.

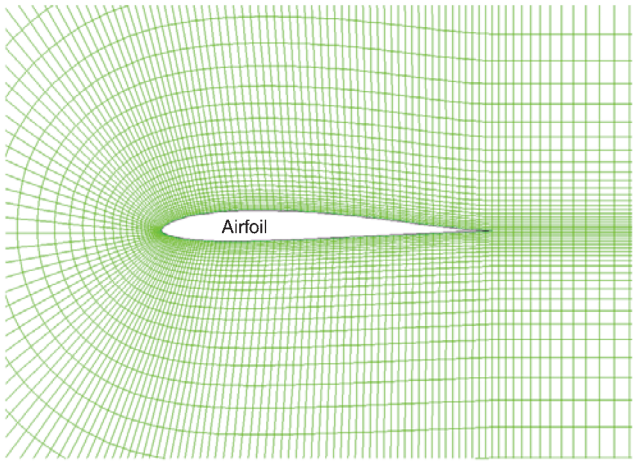


Fig. 4 Gambit 2-D model, 12,240 quadrilateral cells.

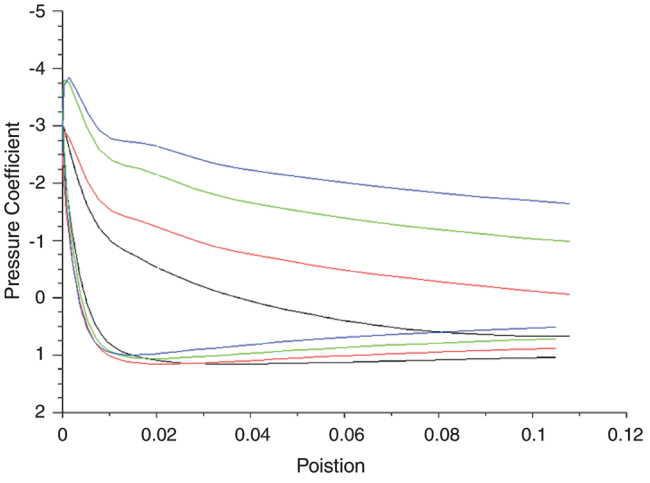


Fig. 6 Zooming into Fig. 5.

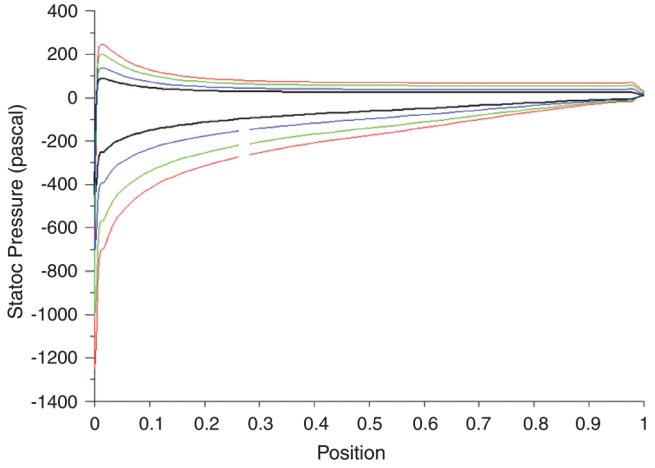


Fig. 7 C_p plot for $V_\infty = 12, 15, 18, 20$ m/s and $\alpha = 2$ deg.

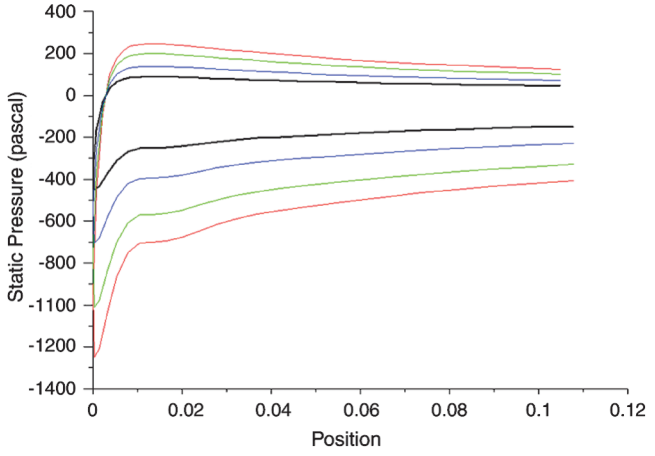


Fig. 8 Zooming into Fig. 7.

B. Location of the Matrix of Pressure Orifices

The number of pressure ports (orifices) must be chosen as a compromise between the need to accurately estimate the air data states and the need to reduce instrumentation costs. Choosing a large number of pressure ports can improve the fault tolerance properties of the FADS system and its robustness to noise caused by, for example, air turbulence or instrumentation noise. However, not only do a large number of pressure ports increase the instrumentation cost, but in MAVs we must take into account the space and weight limitations. In our case, there are three air data states to be estimated (P_∞, V_∞, α) and therefore a minimum of three pressure ports are required. Two extra ports are added to improve the redundancy options and the noise sensitivity of the FADS system. We will from now on refer to these five pressure ports collectively as the matrix of pressure orifices (MPO) which include P1, P2, P3, P4, and P5 (Fig. 9).

It is important that the MPO are distributed in such a way that 1) ports close to each other give different pressure measurements (this avoids having redundant information) and 2) the pressures measured are sensitive to variations in the air data states P_∞, V_∞, α. In Sec. IV.A, we concluded that the latter criterion is satisfied at $x/c < 0.01$. Taking this into account, two pressure ports were placed on the upper surface of the wing, two on the lower surface, and one close to the tip of the wing leading edge (see side view of Fig. 9). The latter port is included as a stagnation pressure source (i.e., for approximate measurements of P_0).

As far as the author is aware, there is no universally accepted standard for the minimum hole spacing (i.e., the distance between adjacent pressure ports) and indeed the size of the pressure ports (i.e., the diameter of the pressure orifices). For example, [6,13,21,31] use a hole diameter of 0.76, 0.79, 6.40, and 5.00 mm and a minimum hole spacing of 1.08, 15.25, 61, and 406 mm, respectively. In general, it was found from the literature that a minimum hole spacing of 1 mm and a hole diameter of 0.5 mm were suitable.

Distributing the pressure ports along the wing span (see front view of Fig. 9) can help increase the hole spacing. This is important for two reasons. First, this avoids having redundant information. Second, it can avoid any flow interference between the adjacent ports. Pressure orifices drilled into the surface of the wing will result in uneven edges around the port and so the air flow over the orifices will not be

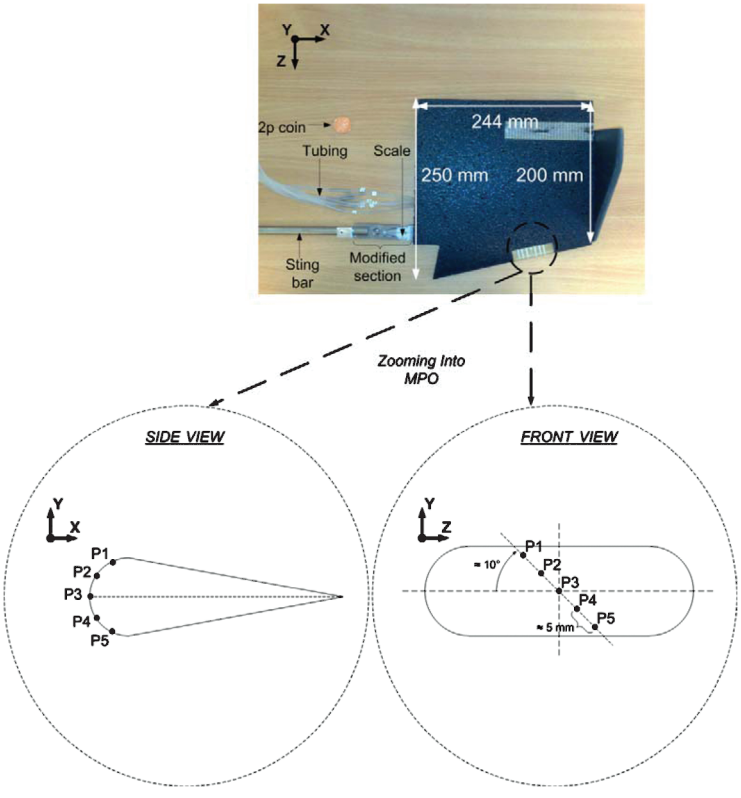


Fig. 9 Top view of wing and MPO (with pressure ports P1, P2, P3, P4, P5); MPO not shown to scale.

smooth. This disruption of the air flow can consequently affect the pressure measurements of adjacent ports. This flow interference effect between adjacent ports is one of the main reasons why pressure ports in FADS systems must be suitably spaced.

Consider the side view of Fig. 9. To increase the hole spacing, we could of course spread the ports over the whole aerofoil (i.e., shifting the pressure ports toward the trailing edge). However, by doing so, we would no longer fulfil the criterion that ports must be placed at $x/c < 0.01$. Therefore, instead we distribute them along the wing span (see front view of Fig. 9). In conclusion, there are three important criteria when choosing the locations of the five pressure ports:

1) They must all be located at $x/c < 0.01$.

2) A minimum hole spacing of 1 mm and hole diameter of 0.5 mm must be used.

3) The MPO must be placed far enough from the wing root and tip.

The latter criterion is important if we are to avoid the effects of flow separation at the wing tip and the turbulent air behind the nose propeller, that is, the nose propeller wash. For this reason, the middle port P3 (Fig. 9) was placed at a span of 200 mm from the wing root (i.e., 44 mm from the wing tip). Table 2 shows the locations of the five pressure ports in terms of x/c, and Fig. 9 shows the MPO mounted on the MAV wing where 0.50 mm hole diameter and 5.00 mm hole spacing were used.

V. Wind-Tunnel Results and Conclusions

A. Wind-Tunnel Setup

Wind-tunnel experiments were carried out at Leicester University (Fig. 10). The wind tunnel used is an open-ended subsonic wind tunnel capable of reaching speeds of 40 m/s. It has a sufficiently large working section of 0.46 m × 0.46 m. The MAV wing section was sting mounted in the wind tunnel. The external balance supporting the sting bar included a scaled-turning knob (calibrated in degrees) which allowed the variation and measurement of wing α. To allow for sideslip variations, the standard sting bar was slightly modified (Fig. 9). The part allowing the sideslip comprises two rectangular pieces of metal with one end rounded to a semicircle. The straight edge of one is connected to the sting bar and the straight edge of the other is connected to a bar which is fixed to the wing (near the wing center of gravity). The two plates can rotate about a hole through which passes a bolt to lock them together at the chosen angle. The rounded ends carry a scale to indicate the set wing β. A pitot-static tube was mounted in the wind tunnel ahead of the wing to measure P_0 and P_∞. Using these two measurements, wind speed V_∞ can also be calculated. The pitot-static tube is connected to the data-acquisition (DAQ) card and P_0, P_∞ are recorded in LabView (wind speed V_∞ is then calculated in LabView). A potentiometer was connected to the turning knob of the external balance and calibrated to measure applied α, and β was simply recorded from the scale on the sting bar. Digital acquisition of β settings was not necessary as dynamic tests were only done for changes in α but fixed β. To compensate for any noise corruption, wind-tunnel data were initially filtered. A second-order Butterworth low-pass filter was implemented in LabView. To comply with the Nyquist sampling theorem, the cutoff frequency chosen must be smaller than half the sampling frequency. With a sampling frequency of 50 Hz, the cutoff frequency was heuristically chosen to be 1.5 Hz.

Each pressure orifice was connected to a calibrated differential (with respect to room pressure) pressure sensor via pressure tubing. It is important to keep the tubing distance to a minimum to avoid any time lags. The sensor board was therefore placed as close as possible to the wind tunnel, giving an approximate tubing length of 0.38 m. The pressure sensors have a pressure range of ±5 in. H_2O and were read in by a PC-based 16-bit DAQ card at a sampling frequency of 50 Hz. The DAQ software used to log the pressure data for each test was LabView.

Table 2 Pressure port locations (also refer to Fig. 9)

Port	x/c position
P1	0.009
P2	0.003
P3	0.001
P4	0.003
P5	0.009

B. Tests Outline

Two types of wind-tunnel tests were carried out: static and dynamic. Static tests involve fixing the wind-tunnel settings (i.e., V_∞, α, β settings) and recording the data only once the measurements have reached a steady-state value. The data can then be stored (in LabView) and analyzed offline in Matlab/Simulink. The static tests carried out are outlined in Fig. 11. Overall, there are 252 separate tests with different V_∞, α, β settings. So, for example, in Fig. 11 one of the tests would consider a setting of $\beta = 0$ deg, $V_\infty = 15$ m/s, and $\alpha = -9$ deg.

The second group of tests are dynamic tests, where $\dot{\alpha} \neq 0$. Static tests are useful as a first step toward analyzing the NN modeling capabilities and to investigate whether or not the wing surface pressure can be related to the air data states (P_∞, V_∞, α) as predicted by the CFD simulations (Sec. VI). However, in real flight, static test results are of little use, as the aerodynamic state of the aircraft rarely stays constant. For this reason, it is important to perform dynamic tests. In our case, this was considered by continuously varying the angle of attack for each test (i.e., $\dot{\alpha} \neq 0$) and fixing the airspeed and sideslip at $V_\infty = 15$ m/s and $\beta = 0$ deg, respectively. The wing angle of attack is randomly varied at different $\dot{\alpha}$ and waveforms: square wave, sine wave, and ramp type. Note that certain time evolutions of $\dot{\alpha}$ may not be feasible in real flight, but were necessary to analyze the modeling capabilities of the NN.

C. Wind-Tunnel Data

Wind-tunnel data were recorded before any NN implementation. As discussed in Sec. IV.B it is important that 1) the pressure ports (P1, P2, P3, P4, P5) give different pressure measurements and 2) these pressure measurements vary significantly with the air data states (P_∞, V_∞, α). Figure 12 shows some of the wind-tunnel data recorded during the tests. From Fig. 12, we notice that 1) ports give different pressure measurements, which are important to avoid having redundant information, 2) steep pressure gradients are observed at each port for a change in angle of attack setting, and 3) for fixed angle of attack, the pressure measurements at each port vary with the different airspeed settings.

D. Neural Network Training and Validation Stage

The NN is trained and validated, as discussed in Sec. III.C, before the static and dynamic tests. The wind-tunnel data (some of which has been shown in Fig. 12) is analyzed offline in a Matlab/Simulink environment. The following data are used for the NN training and validation stage:

1) TrD: This is the training set which includes the alpha settings of −9, −5, −1, 3, 7, and 11 deg.

2) VeD: This is the validation set which includes the alpha settings of −7, 1, and 9 deg.

Note that in each case all the β settings (−9 to 9 deg, in increments of 3 deg) and V_∞ settings (12, 15, and 20 m/s) are considered. Therefore, 126 static tests are used to train the NN with learning switched on, and the remaining 63 are used to query/validate the trained NN with learning switched off.

NN training is stopped if $\Delta\text{rms} < 0.1\%$ for more than 100 consecutive epochs (for both TrD and VeD) and/or the rms estimation error for VeD increases for more than 100 consecutive epochs. These criteria check for NN structural convergence and avoid the overfitting phenomenon, respectively (Sec. III.C).

The NN learning rate δ chosen is also crucial, as a high learning rate guarantees good estimations but it also degrades the global approximation capability of the network. There is no formal guideline to defining the optimum learning rate and the other NN tuning parameters [1]. The designer must apply a heuristic-based approach when choosing the NN tuning parameters. Satisfactory

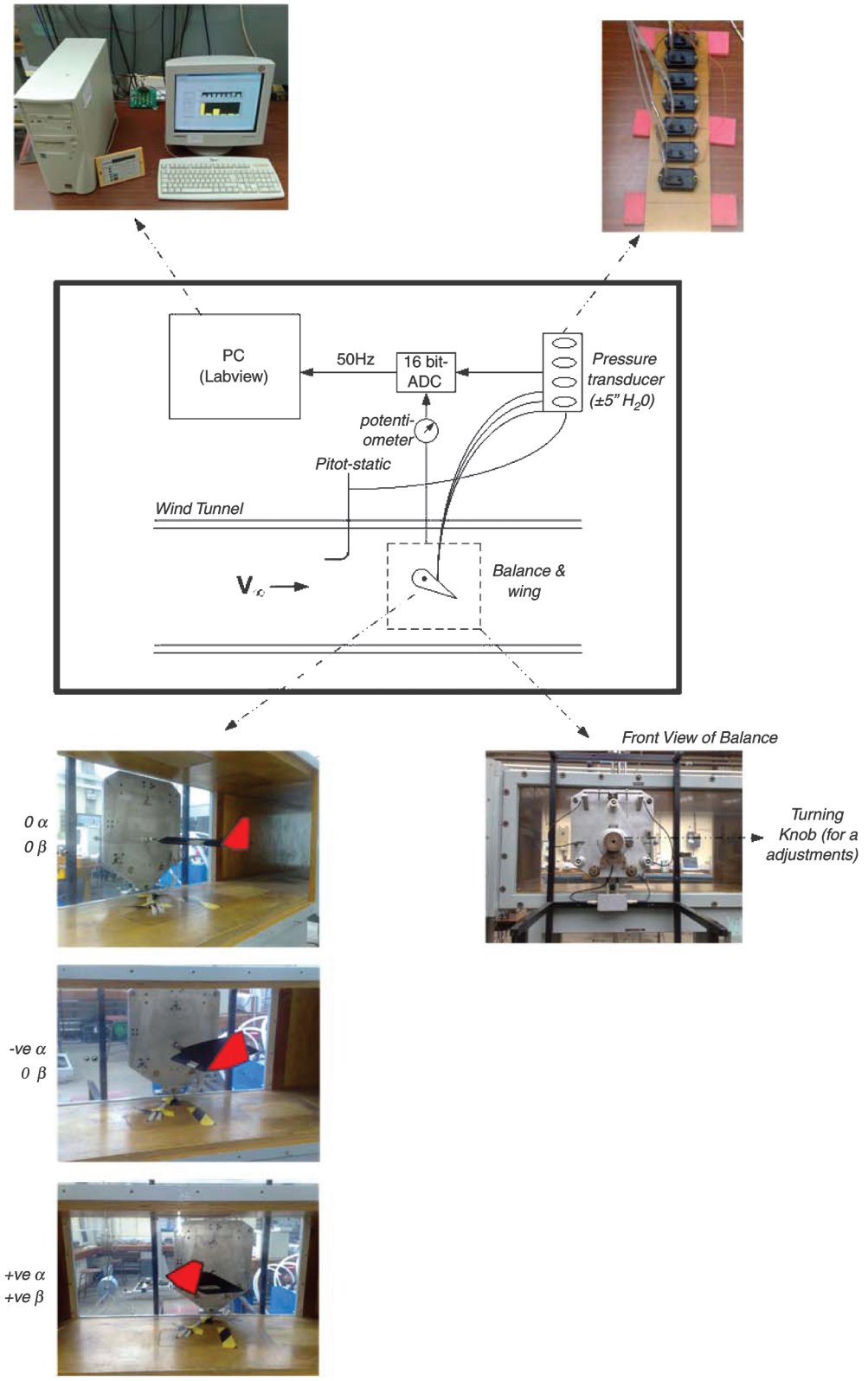


Fig. 10 Wind-tunnel settings and instrumentation.

performance can then be judged based on the estimation characteristics and execution speed of the NN. The EMRAN RBF NN tuning parameters used in our case are defined in Table 3.

Figure 13 shows the NN training stage. Note that the estimation error units are not included as it constitutes mixed NN outputs (i.e., the average of the estimation error for $\hat{P}_\infty$, $\hat{V}_\infty$, $\hat{\alpha}$). Training is stopped after 582 epochs and the NN structure frozen. The resulting NN is a fully connected 5-3-3 NN, shown in Fig. 14.

E. Static Test Results

The NN static test stage involves querying the 5-3-3 NN (Fig. 14) with the testing data set TeD: −7, −3, 1, 5, 9, and 13 deg. Note that all the β settings (−9 to 9 deg, in increments of 3 deg) and V_∞ settings (12, 15, and 20 m/s) are also considered. Therefore, overall, there are 126 static tests. The results are broken down in the Appendix. Overall, the NN rms estimation errors were 0.44 lb/ft^2, 0.62 m/s, and 0.51 deg for $\hat{P}_\infty$, $\hat{V}_\infty$, and $\hat{\alpha}$, respectively.

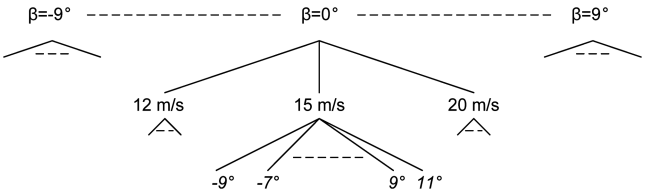


Fig. 11 Wind-tunnel static tests. Seven different β settings (in 3 deg increments), three different V_∞ settings, and 12 different α settings (in 2 deg increments). Overall, there are 252 separate static tests.

F. Robustness to Noise and Faults

NN-based FADS systems rely on accurate pressure measurements to produce accurate air data estimates. However, due to the rough terrains that aircraft (especially MAVs) can often be exposed to, pressure orifices can be susceptible to blockage. Noisy input pressure data, sensor errors, and electrical wiring failures are other example scenarios which can degrade the performance of the FADS system. For this reason, it is important that the fault and noise tolerance capabilities of the FADS system are rigorously tested before implementation. In this section, we investigate the robustness of the FADS system to one popular type of fault scenario: total sensor failure.

Fault detection has been the subject of many engineering applications over the years. The most straightforward methods include physical redundancy where faults are detected based on a voting scheme, and limit value checking methods where the sensor measurement is continuously checked against a predefined threshold. Following fault detection, faulty measurements must then be accommodated in the system to avoid any performance degradation. In our study, we will investigate the effects on the NN estimation performance once a fault is detected and methods that can be adopted to reduce the effect of faults.

To investigate this, we artificially introduce faults in the NN testing data set, TeD. The type of fault we consider is total sensor failure. This is a catastrophic failure where the sensor stops working and outputs a constant zero reading. As we are so far only considering static tests, the time evolution of the fault is not considered.

The trained 5-3-3 NN structure (Fig. 14) is queried with TeD but this time one, two, or three of the NN inputs are set to zero, that is, we consider single and multiple sensor failures. Three fault accommodation scenarios are then studied:

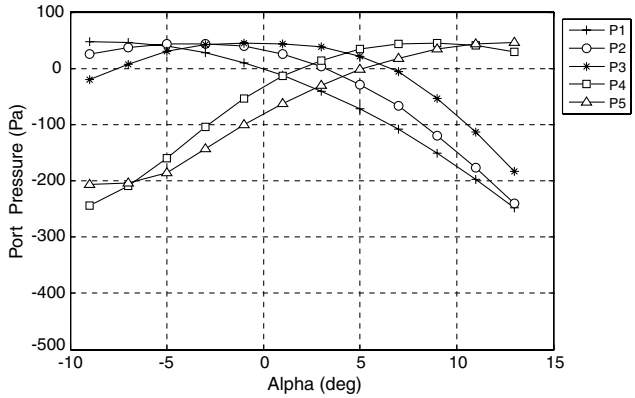
1) No correction: Faulty inputs are not accommodated.

2) Next port: In this case, we simply accommodate the fault by replacing its measurement with the neighboring nonfaulty port. So, for example, in Fig. 9, if P1 measurements are faulty, then pressure from P2 replaces them. The NN will therefore have pressures ports P2, P2, P3, P4, and P5 as inputs. If P1, P3, and P5 are faulty, then the NN inputs would be P2, P2, P2, P4, and P4.

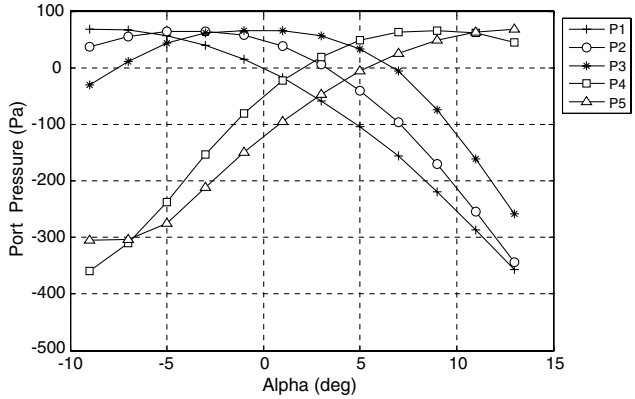
3) AA-NN: The faulty port reading is replaced with the corresponding output from an autoassociative NN (AA-NN). An AA-NN is one which simply reproduces its inputs at its outputs. In our case, a fully connected 5:10:5 EMRAN RBF NN was developed to do so.

Figure 15 shows the results for the fault accommodation tests. In general, we notice that, if the fault is not accommodated, the NN rms estimation errors increase significantly especially in the presence of multiple faults (with $\hat{\alpha}$ rms error reaching 14 deg for three faults). However, when fault accommodation is implemented by replacing the faulty port measurement with its neighboring port, estimation errors are reduced significantly. In our case, using the "next-port" option resulted in an average 50.39% decrease in the NN rms estimation errors. On the other hand, using the "AA-NN" option resulted in a larger reduction of 69.6%. The use of the autoassociative NN greatly improved the robustness of the FADS system to faults.

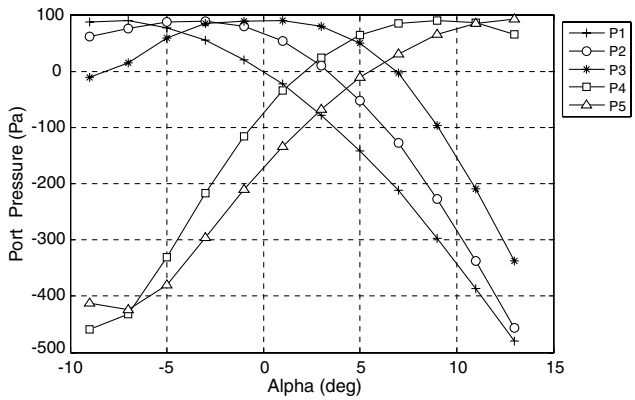
A drawback of using the AA-NN is the further memory usage required onboard the air vehicle. The next-port option does not suffer from this which is why, in many applications of the FADS system, the designer tends to include redundant pressure orifices 1) to improve the redundancy options in the event of a fault and 2) to take the average of redundant pressure measurements and therefore reduce the noise effects. In our design, we can see that the pairs P1, P2 and P4, P5 could be considered as redundant port locations as they have similar pressure distributions (Fig. 12).



a)



b)



c)

Fig. 12 Port pressure distribution for three different speeds and fixed $\beta = 0$ deg: a) $V_\infty = 12$ m/s, b) $V_\infty = 15$ m/s, and c) $V_\infty = 20$ m/s.

Table 3 EMRAN RBF NN tuning parameters

Tuning parameter	Value
Learning rate	0.2
$E1$	0.2
$E2$	0.1
$\varepsilon_{\max}$	0.6
$\varepsilon_{\min}$	0.3
γ_{df}	0.997

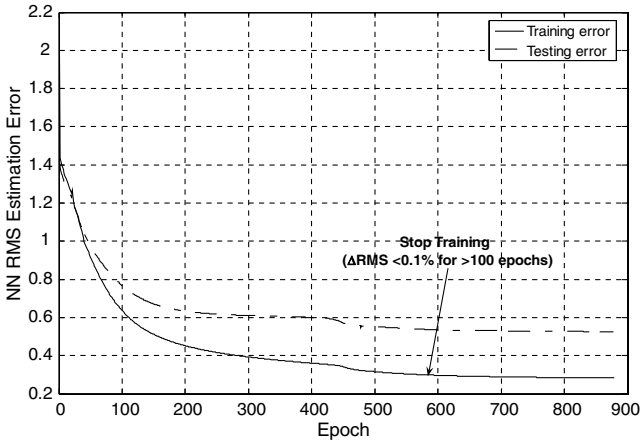


Fig. 13 NN training/testing rms estimation errors.

In conclusion, including redundant pressure ports in the FADS system improves the robustness of the overall system to faults. Furthermore, it can mitigate the effects of system noise. However, in applications where space onboard the vehicle is limited or instrumentation costs need to be low, an AA-NN can be used to reproduce pressure measurements which are faulty or corrupted by noise.

G. Dynamic Test Results

The 5-3-3 NN structure concluded in Fig. 16 is used in the dynamic tests with learning switched off. At a fixed $\beta = 0$ deg and $V_\infty = 15$ m/s, the wing angle of attack is varied continuously (i.e., $\dot{\alpha} \neq 0$) and the NN estimate $\hat{\alpha}$ recorded. An overall alpha rms estimation error of 0.58 deg was achieved.

Figures 16 and 17 show the NN estimation characteristics for two of the dynamic tests carried out. Let us refer to them as dynamic test 1 (DT1) and dynamic test 2 (DT2), respectively. Let us now investigate the estimation characteristics of DT1 in detail (Fig. 16). It can be seen that, at certain time frames, the NN estimations are significantly poor. For example, at around 120 s, where $\alpha > 15$ deg, the NN underestimates α by almost 10 deg. At first, it may seem that this is simply due to random estimation error patterns, that is, the NN performance

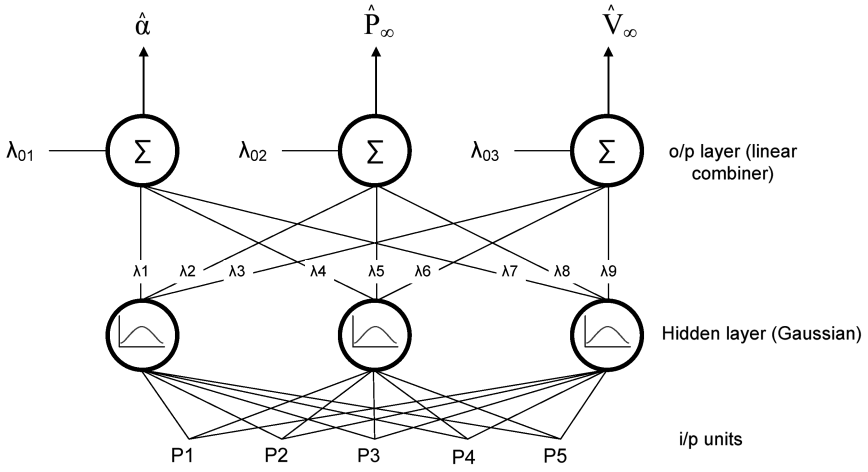


Fig. 14 RBF NN with 5-3-3 structure; λ represents the weights, P represents the pressure ports in Fig. 9.

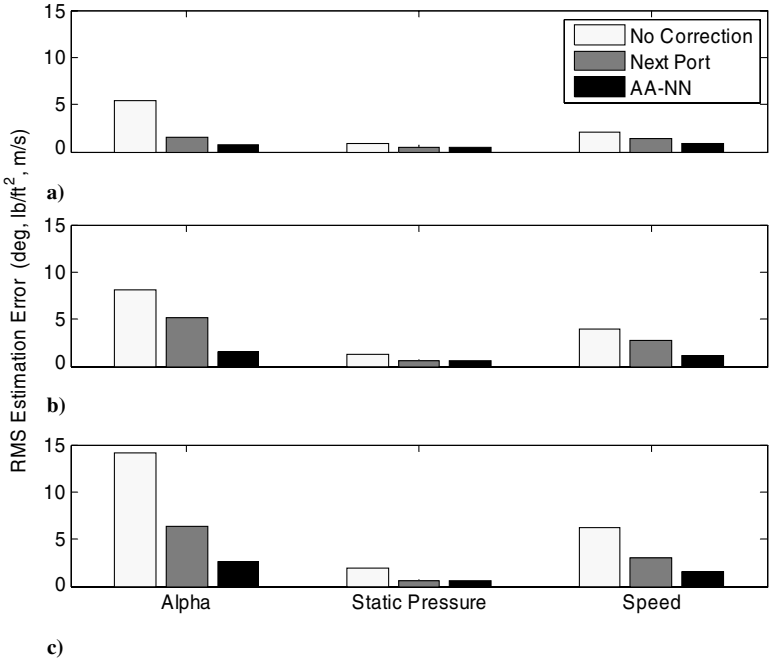


Fig. 15 NN rms estimation errors in the presence of total sensor failure: a) P1 faulty, b) P1, P5 faulty, and c) P1, P3, P5 faulty.

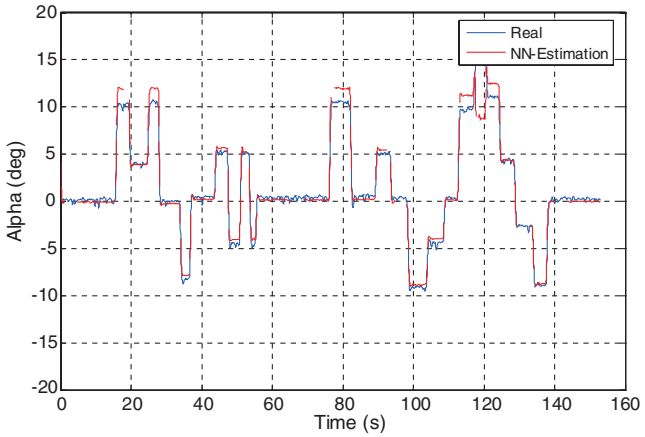


Fig. 16 RBF NN estimation ($\hat{\alpha}$) for DT1.

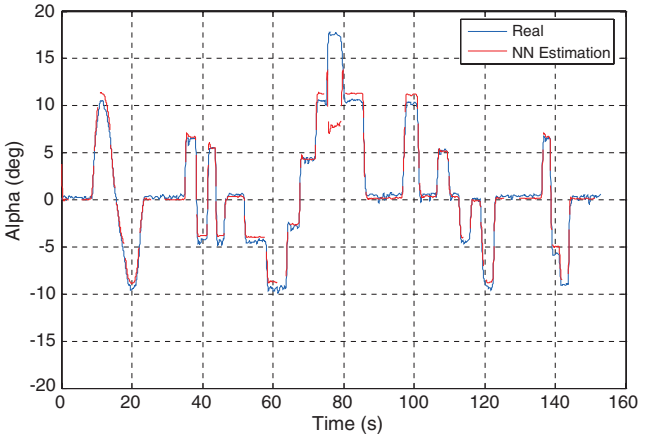


Fig. 17 RBF NN estimation ($\hat{\alpha}$) for DT2.

is generally accurate except for some random time frames. However, if we observe the estimations from DT2 (Fig. 17), we notice that, at approximately 75 s, where again $\alpha > 15$ deg, the NN underestimates α by almost 10 deg. This shows that the NN performance is in fact poor for specific α settings, such as $\alpha > 15$ deg.

One explanation for this observation is related to the NN domain of validity [1]. The RBF NN is essentially a multidimensional interpolator. The accuracy of the surface fit to new data (i.e., its ability to generalize) is highly dependent on the location of these data in the input space, that is, the location of that point relative to the domain of validity of the NN where the domain of validity defines the boundaries of the data set used to train the NN [32]. A general rule is that, when the NN is queried with data which lie outside these boundaries, the NN estimations are poor or, in other words, the extrapolation properties of the NN are poorer than its interpolation properties. This can be better explained by reiterating the RBF NN structure. The fundamental principle of the RBF NN is that the output of each RBF found in the hidden layer (i.e., the output of each hidden neuron) is proportional to the distance between the input vector and the center of that hidden neuron. So, for example, in our case we chose a Gaussian-RBF. Therefore, data which lie far from the center of the Gaussian function will output a close-to-zero value. These centers are tuned based on the training data set, TrD. Therefore, testing data which are far away from the domain of TrD will also lie far from the centers of the Gaussian functions, and so the NN output will always be close to zero. In other words, the NN estimations will be poor for testing data lying outside the domain of TrD, that is, outside the domain of validity.

For this reason, it is important that the NN is trained with as much flight data as possible and, more important, with data that cover the entire flight data range. This reduces the likelihood of the testing data lying outside the NN domain of validity. In our case, the method of dividing the training (TrD) and testing (TeD) data sets was chosen for ease of presentation. However, a more robust approach is needed to define TrD, so that the domain of validity encompasses all possible input data patterns. This is particularly important in our case, as the NN structure is frozen after the offline training stage and will not be further trained during actual flight. In conclusion, a more robust approach to training the NN must include 1) collecting all the possible flight data (this would depend on previous flight experience), 2) defining the domain of validity of the collected flight data, and 3) including the outermost values of the domain of validity to train the NN.

A good survey paper for the different methods to defining the domain of validity can be found in [33]. The simplest approach is by defining the minimum and maximum of each NN input parameter (in our case, that would be each pressure port). Testing data that lie outside these limits is said to be outside the domain of validity. Despite its simplicity, this method unfortunately overestimates the domain of validity. More complex methods involve defining the convex hull (CH) of the training data set. The vertices of this convex hull can then be used to determine the exteriority of a new input pattern. Patterns that lie outside the CH have an exteriority > 0, and ones that are in the CH have an exteriority $= 0$. A convex hull is a multidimensional surface which tightly encompasses a set of multidimensional data [1]. There are several techniques to defining the convex hull and its vertices. They are generally based on iterative mathematical algorithms which continuously check the exteriority of a new input pattern. If this exteriority is greater than zero, then the input pattern is included in the convex hull and so on. See, for example, Chapter 4.4 in [1] for a code which calculates the vertices of the convex hull. Another method of calculating the domain of validity is by defining a sphere that encompasses all the input data patterns. Data patterns which lie outside the sphere are considered to be outside the domain of validity [32]. As in the min / max method, this approach is simple to implement as we would only need to know the radius of the sphere to define whether the input pattern lies outside the domain of validity. However, similar to the min / max method, this technique can grossly overestimate the domain of validity.

Regardless of the technique used to defining the domain of validity, it is important that, once it has been defined, the vertices (i.e., the outermost values) of the domain of validity are included in the NN training set, as this greatly reduces the need for NN extrapolation. As wind-tunnel tests generally investigate only a group of flight test conditions, it is difficult to define the domain of validity based on only the wind-tunnel data.

To demonstrate the importance of carefully selecting the NN training set TrD, we will reconsider the NN estimations for DT1 in Fig. 16. For simplicity purposes, let us define the domain of validity using the min / max approach described earlier. We will assign pressure from TrD and DT1 the subscripts "_TrD" and "_DT1," respectively. The following code (shown for P1 only) can be used to indicate (graphically), whether a pressure input is outside the domain of validity defined by P1_TrD (i.e., defined by the min / max range of the pressure from P1 in the training set TrD):

%P1_TrD is the pressure from port 1 in the training set TrD
%P1_DT1 is the pressure from port 1 in the dynamic test DT1
If $\mathbf{P1_{DT}1} <$ **min** (P1_TrD) OR $\mathbf{P1_{DT}1} >$ **max** (P1_TrD)
    This indicates that P1_DT1 is outside the domain of validity.
    Plot the number 25.
end

The code states that if pressure from P1_DT1 exceeds the bounds defined by P1_TrD, then plot a value of 25. The plot value is not meaningful, it is chosen simply for ease of representation. Similarly, the preceding code can be implemented for the remaining pressure ports P2, P3, P4, and P5 but with plot values of 20, 15, 10, and 5, respectively. Figure 18 shows the results from this task. We have also included the residual from DT1 which was calculated simply as the difference between the NN estimates $\hat{\alpha}$ and the measured (real) value α. Note also that the plots, in Fig. 18, for P4 and P5 cannot be seen, as they do not exceed any limits.

From Fig. 18 we notice how the residual significantly increases (i.e., NN estimations are poor) only when the pressure input patterns

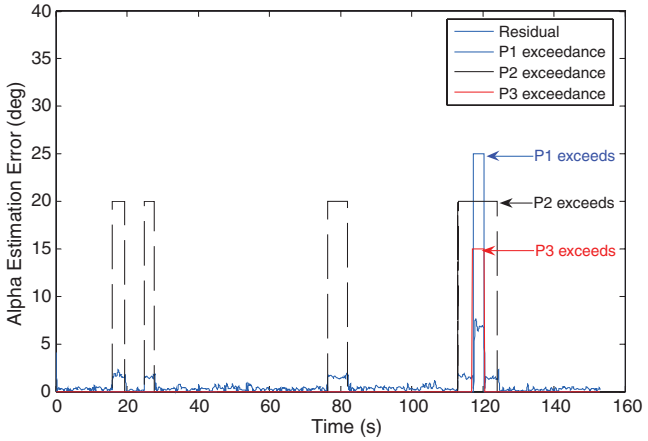
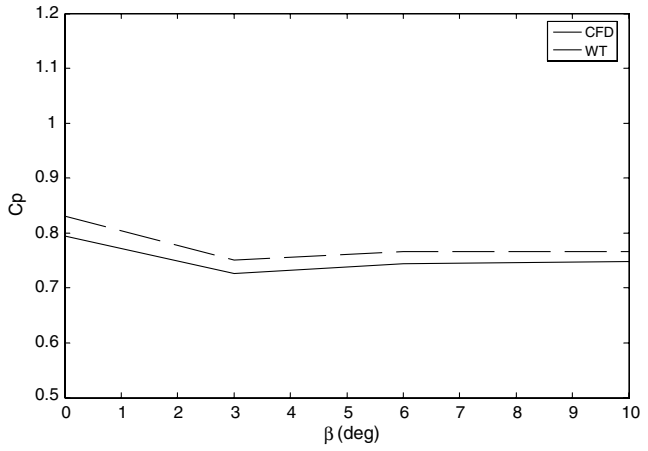


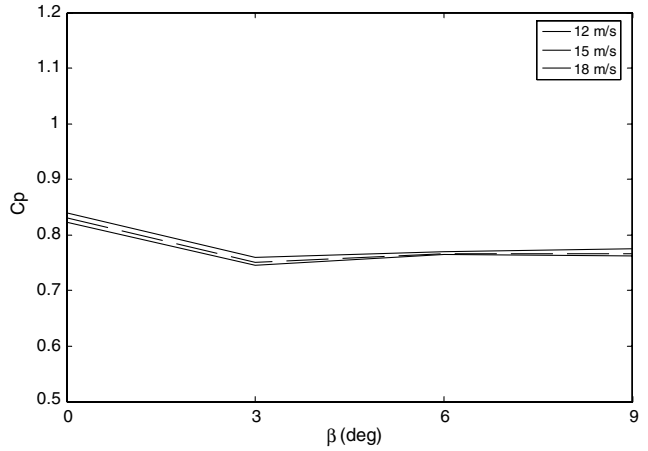
Fig. 18 Domain of validity test for P1, P2, P3. If P1, P2, P3 lie outside the domain of validity, a value of 25, 20, 15 is plotted, respectively, otherwise a value of zero is plotted. NN residual also shown (solid blue line).

lie outside the NN domain of validity. So, for example, the residual is highest at 120 s when pressure from ports P1, P2, and P3 exceed the min / max limits.

The simple domain of validity test presented here has confirmed the importance of appropriately selecting the NN training set. Different approaches exist for doing so, and it is up to the designer to choose a suitable method. It is perhaps also important to note that training data are best obtained from real flight tests, as the FADS system will eventually be used during real flight. The NN domain of validity must then be updated from flight test runs, that is, it must be updated with every new pressure pattern which is exterior to the existing domain of validity.



a)



b)

Fig. 19 Wind-tunnel data a) and CFD for pressure port P3 (V_∞= 15 m/s, $\alpha = 0$ deg), b) when changing the airspeed $V_\infty = 12$, 15, 18 m/s.

H. Computational Fluid Dynamics vs Wind Tunnel

To validate the CFD results, we can also plot C_p vs β from the wind-tunnel data. Such a plot is shown in Fig. 19a where the CFD predictions and the wind-tunnel data are both plotted for pressure measured from P3. There are two important observations to make from Fig. 19a:

1) CFD and wind-tunnel data are almost similar. There will be some differences due to the assumptions considered in the CFD simulations (e.g., inviscid flow).

2) Both the CFD and wind-tunnel data show that the variation of pressure with sideslip is insignificant.

Figure 19b also shows the wind-tunnel data for a change in β for three airspeed settings. From both Figs. 19a and 19b, we can conclude that the pressure ports located as in Fig. 9 are insensitive to changes in sideslip. Therefore, it can be difficult to define a NN model that relates the surface pressure to the sideslip, as this relationship is almost constant, especially for $\beta > 3$ deg (see Fig. 19). It is for this reason that the estimation of sideslip was disregarded during the project and only the three air data states estimates ($\hat{P}_\infty$, $\hat{V}_\infty$, $\hat{\alpha}$) were considered in the NN (Fig. 14).

VI. Conclusions

This paper has investigated the application of a FADS system on the wing leading edge of a MAV which flies at speeds as low as 8 m/s. Traditionally, the aerodynamic model used to relate the aircraft surface pressure to the air data is derived based on several assumptions (such as spherical nose shapes). Furthermore, the model is highly nonlinear and can be difficult to solve. Instead, an EMRAN RBF NN model was proposed. The location of the pressure orifices on the MAV wing was concluded from a simple CFD study of the approximate pressure gradients at the wing leading edge. It was noted that, despite the fact that numerical uncertainties would be present, the CFD study was sufficient for our purposes here as shown by the good performance of the FADS system. A 5-3-3 NN was designed and shown to give estimation accuracies of 0.44 lb/ft^2, 0.62 m/s, and 0.51 deg for $\hat{P}_\infty$, $\hat{V}_\infty$, and $\hat{\alpha}$, respectively. It was found that the NN must be robustly trained and ideally the outermost values of flight data range should be included in the NN training set. This is best implemented using real flight data, as wind-tunnel tests are limited to specific flight conditions. The FADS system must be tested in real flight to gain more confidence in their estimations accuracies, execution speeds, and stability to fluctuating pressure measurements (caused by, e.g., atmospheric debris partially blocking the pressure ports). The FADS system can be flight tested in parallel to an air data boom and the latter can be used to validate the performance of the FADS system. If the FADS system is successful in real flight, then we can investigate the possibility of combining the FADS system and air data boom for a more accurate, fault tolerant, and robust (to sensor noise) air data system. Furthermore, the air data boom can be used to train (online) the NN-FADS system. As pointed out in Sec. V.H, estimating the sideslip (with our current design) can be difficult. However, it was found that pressure close to the wing root and wing tip is in fact sensitive to changes in the sideslip. Instead of redesigning the MPO structure, we can simply add extra pressure ports so that sideslip can be estimated. This is feasible for several reasons:

1) There is sufficient space and weight left on the MAV to mount extra pressure sensors.

2) The NN processing time is currently 98% lower than the flight data sampling time and therefore increasing the number of pressure ports should not cause any significant time delays.

3) Pressure sensors are cheap and therefore increasing the number of pressure ports will not be costly.

UAVs are currently ineligible for a standard airworthiness certificate, and are only assigned a special airworthiness certificate in the experimental category for research and development purposes [34]. However, it is highly feasible that these restrictions will eventually be removed and UAVs will be integrated into the National Airspace System. One of the policies that, for example, the FAA adopts for regulators to issue an airworthiness certificate is based on the air vehicle's potential to do damage. This categorizes the air vehicles in terms of weight, size, speed, etc. Ultimately, weight has relevance for airworthiness risks, and the FADS system suggested here takes this into consideration. Our FADS system weighed approximately 35 g, whereas the mini air data boom typically used by BlueBear Systems Research, Ltd. for their UAVs weighs 170 g. In this case, a reduction in weight of 135 g may not seem significant but, relatively speaking, an 80% reduction in weight can be crucial in large unmanned air vehicles for both flight and airworthiness purposes. In addition, the FADS system's overall cost is almost $120 (£75) in comparison to the air data boom, which costs almost $4000 (£2500). This large cost reduction is mainly of benefit to the military industry, where UAVs are more likely to be destroyed during mission.

Appendix A: NN Estimation Errors

Acknowledgments

The first author is grateful for the financial support from the Overseas Research Students award. The authors would like to sincerely thank the BlueBear Systems Research, Ltd. for providing the mini air vehicle and installing the flush air data sensing system. Special thanks are also given to Paul William (Leicester University) for help with the wind-tunnel tests.

References

[1] Rohloff, T., "Development and Evaluation of Neural Network Flush Air Data Sensing Systems," Ph.D. Thesis, Dept. of Mechanical Engineering, Univ. of California, Los Angeles, Los Angeles, 1998.

[2] Andersen, D., and Haley, D., "NASA Tests New Laser Air Data System on SR-71 Blackbird," NASA Release 93-163, http://www.nasa-usa.de/home/hqnews/1993/93-163.txt [retrieved 17 Sept. 1993].

[3] Cary, J. P., and Keener, E. R., "Flight Evaluation of the X-15 Ball-Nose Flow: Direction Sensor as an Airdata System," NASA TN D-2923, 1965.

[4] Wolowicz, C. H., and Gosett, T. D., "Operational and Performance Characteristics of the X-15 Spherical Hypersonic Flow Direction Sensor," NASA TN D-3076, 1965.

[5] Larson, T. J., and Siemers, P. M., III, "Subsonic Tests of an All-Flush-Pressure-Orifice Air Data System," NASA TP-1871, 1981.

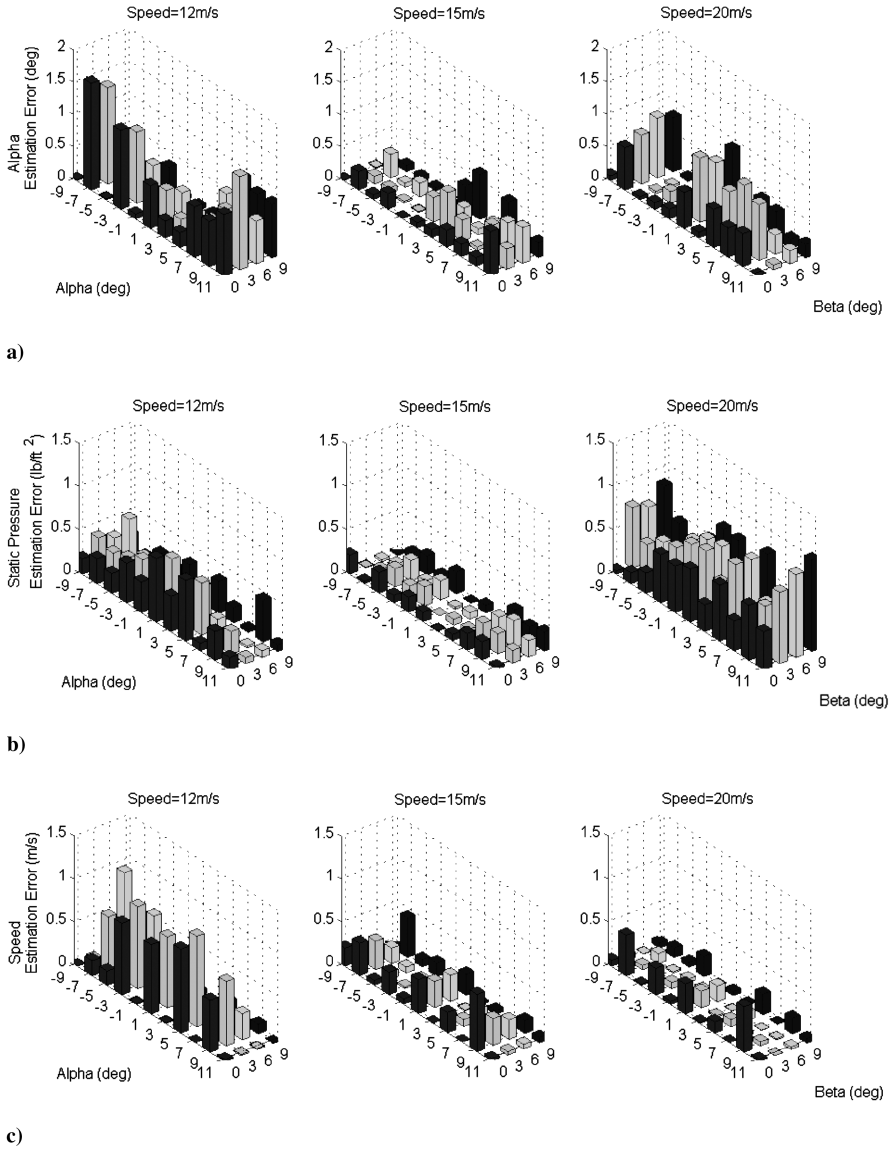


Fig. A1 NN estimation errors at different wind-tunnel settings: a) alpha estimation errors, b) static pressure estimation errors, and c) speed estimation errors. Note that only positive β range shown here.

[6] Larson, T. J., Whitmore, S. A., Ehernberger, L. J., Johnson, J. B., and Siemers, P. M., III, "Qualitative Evaluation of a Flush Air Data System at Transonic Speeds and High Angles of Attack," NASA TP-2716, 1987.

[7] Larson, T. J., Moes, T. R., and Siemers, P. M., III, "Wind Tunnel Investigation of a Flush Airdata System at Mach Numbers from 0.7 to 1.4," NASA TM-101697, 1990.

[8] Whitmore, S. A., Davis, R. J., and Fife, J. M., "In Flight Demonstration of a Real Time Flush Airdata Sensing System," *AIAA Atmospheric Flight Mechanics Conference*, AIAA Technical Papers A95-39269 10-08, AIAA-1995-3433, 1995.

[9] Wenger, C., and Devenport, W., "Seven-Hole Pressure Probe Calibration Method Utilizing Look-Up Error Tables," *AIAA Journal*, Vol. 37, No. 6, 1999, pp. 675–679. doi:10.2514/2.794

[10] Whitmore, S., and Ellsworth, J., "Simulation of Flush Air-data System for Transatmospheric Vehicles," *Journal of Spacecraft and Rockets*, Vol. 45, No. 4, 2008, pp. 716–732. doi:10.2514/1.33541

[11] Rediniotis, O., and Vijayagopal, R., "Miniature Multihole Pressure Probes and Their Neural Network Based Calibration," *AIAA Journal*, Vol. 37, No. 6, 1999, pp. 666–674. doi:10.2514/2.790

[12] Rohloff, T. J., Whitmore, S., and Catton, I., "Air Data Sensing from Surface Pressure Measurements Using a Neural Network Method," *AIAA Journal*, Vol. 36, No. 11, 1998, pp. 2094–2101. doi:10.2514/2.312

[13] Whitmore, S. A., Cobleigh, B. R., and Haering, E. A., "Design and Calibration of the X-33 Flush Airdata Sensing (FADS) System," *36th Aerospace Sciences Meeting and Exhibit*, AIAA Paper 1998-201, 1998.

[14] Whitmore, S. A., Moes, T. R., Czerniejewski, M. W., and Nichols, D. A., "Application of a Flush Airdata Sensing System to a Wing Leading Edge (LE-FADS)," AIAA Paper 1993-634, 1993.

[15] Cobleigh, B. R., Whitmore, S. A., Haering, E. A., Borrer, J., and Roback, V. E., "Flush Air Data Sensing (FADS) System Calibration Procedures and Results for Blunt Forebodies," AIAA Paper 1999-4816, 1999.

[16] Davis, M. C., Pahle, J. W., White, J. T., Marshall, L. A., Mashburn, M. K., and Franks, R., "Development of a Flush Airdata Sensing System on a Sharp Nosed-Vehicle for Flight at Mach 3 to 8," *Aerospace Sciences Meeting and Exhibit*, AIAA Paper 2000-504, 2000.

[17] Bauman, E., Pahle, J., Davis, M., and White, J., "X-34A Flush Airdata Sensing System Flight Test Results," *AIAA Atmospheric Flight Mechanics Conference and Exhibit*, AIAA Paper 2008-6570, 2008.

[18] Rohloff, T. J., Whitmore, S. A., and Catton, I., "Fault-Tolerant Neural Network Algorithm for Flush Air Data Sensing," *Journal of Aircraft*, Vol. 36, No. 3, 1999, pp. 541–549. doi:10.2514/2.2489

[19] Rohloff, T. J., and Catton, I., "Fault Tolerance and Extrapolation Stability of a Neural Network Air-Data Estimator," *Journal of Aircraft*, Vol. 36, No. 3, 1999, pp. 571–576. doi:10.2514/2.2472

[20] Crowther, W. J., and Lamont, P. J., "A Neural Network Approach to the Calibration of a Flush Air Data System," *The Aeronautical Journal*, Vol. 105, No. 1044, 2001, pp. 85–95.

[21] Brown, E. N., Friehe, C. A., and, Lenschow, D. H., "The Use of Pressure Fluctuations on the Nose of an Aircraft for Measuring Air Motion," *Journal of Climate and Applied Meteorology*, Vol. 22, No. 1, 1983, pp. 171–180. doi:10.1175/1520-0450(1983)022<0171:TUOPFO>2.0.CO;2

[22] Whitmore, S. A., Stephen, A., Moes, T. R., Timothy, R., and Larson, T. J., "Preliminary Results from a Subsonic High Angle-of-Attack Flush Airdata Sensing (HI-FADS) System: Design, Calibration, and Flight Test Evaluation," NASA TM-101713, 1990.

[23] Haykin, S., *Neural Networks: A Comprehensive Foundation*, Macmillan, New York, 1994.

[24] Powell, M. J. D., "Radial Basis Function for Multivariable Interpolation: A Review," *Algorithms for Approximation*, edited by J. C. Mason, and M. G. Cox, Clarendon, Oxford, England, U.K., 1987, pp. 143–167.

[25] Li, Y., Sundararajan, N., and Saratchandran, P., "Analysis of Minimal Radial Basis Function Network Algorithm for Real-Time Identification of Nonlinear Dynamic Systems," *IEE Proceedings on Control Theory and Applications*, Vol. 147, No. 4, 2000, pp. 476–484. doi:10.1049/ip-cta:20000549

[26] Chen, S., Cowan, F. N., and Grant, P. M., "Orthogonal Least Squares Learning Algorithm for Radial Basis Function Networks," *IEEE Transactions on Neural Networks*, Vol. 2, No. 2 1991, pp. 302–309. doi:10.1109/72.80341

[27] Platt, J. C., "A Resource Allocating Network for Function Interpolation," *Neural Computation*, Vol. 3, No. 2, 1991, pp. 213–225. doi:10.1162/neco.1991.3.2.213

[28] Kadirkamanathan, V., and Niranjan, M., "A Function Estimation Approach to Sequential Learning with Neural Networks," *Neural Computation*, Vol. 5, No. 6, 1993, pp. 954–975. doi:10.1162/neco.1993.5.6.954

[29] Fravolini, M. L., Campa, G., Napolitano, K., and Song, Y., "Minimal Resource Allocating Networks for Aircraft SFDIA," *IEEE International Conference on Advanced Intelligent Mechatronics*, Vol. 2, Inst. of Electrical and Electronics Engineers, New York, 2001, pp. 1251–1256.

[30] Anderson, J. D., *Fundamentals of Aerodynamics*, 2nd ed., McGraw–Hill, New York, 1991.

[31] Larson, T. J., Flechner, S. G., and Siemers, P. M., "Wind Tunnel Investigation of an All Flush Orifice Air Data System for a Large Subsonic Aircraft," NASA TP 1642, May 1980.

[32] Courrieu, P., "Three Algorithms for Estimating the Domain of Validity of Feedforward Neural Networks," *Neural Networks*, Vol. 7, No. 1, 1994, pp. 169–174. doi:10.1016/0893-6080(94)90065-5

[33] Helliwell, I. S., Torega, M. A., and Cottis, R. A., "Accountability of Neural Networks Trained with 'Real World' Data," *Fourth International Conference on Artificial Neural Networks*, Inst. of Electrical and Electronics Engineers, Piscataway, NJ, 1995, pp. 218–222.

[34] FAA: Unmanned Aircraft Systems (UAS) Certifications and Authorizations, U.S. Dept. of Transportation, http://www.faa.gov/aircraft/air_cert/design_approvals/uas/cert/ [retrieved 5 Nov. 2007].