Angle of attack prediction using recurrent neural networks in flight conditions with faulty sensors in the case of F-16 fighter jet

Bemnet Wondimagegnehu Merasha¹ · David N. Jansen^{2,3} · Hongbin Ma^{1,4}

Received: 9 March 2021 / Accepted: 3 December 2021
© The Author(s) 2021

Abstract

The angle of attack (AOA) is one of the critical parameters in a fixed-wing aircraft because all aerodynamic forces are functions of the AOA. Most methods for estimation of the AOA do not provide information on the method's performance in the presence of noise, faulty total velocity measurement, and faulty pitch rate measurement. This paper investigates data-driven modeling of the F-16 fighter jet and AOA prediction in flight conditions with faulty sensor measurements using recurrent neural networks (RNNs). The F-16 fighter jet is modeled in several architectures: simpleRNN (sRNN), long-short-term memory (LSTM), gated recurrent unit (GRU), and the combinations LSTM-GRU, sRNN-GRU, and sRNN-LSTM. The developed models are tested by their performance to predict the AOA of the F-16 fighter jet in flight conditions with faulty sensor measurements: faulty total velocity measurement, faulty pitch rate and total velocity measurement, and faulty AOA measurement. We show the model obtained using sRNN trained with the adaptive momentum estimation algorithm (Adam) produces more exact predictions during faulty total velocity measurement and faulty total velocity and pitch rate measurement but fails to perform well during faulty AOA measurement. The sRNN-GRU combinations with the GRU layer closer to the output layer performed better than all the other networks. When using this architecture, the correlation and mean squared error (MSE) between the true (real) value and the predicted value during faulty AOA measurement increased by 0.12 correlation value and the MSE decreased by 4.3 degrees if one uses only sRNN. In the sRNN-GRU combined architecture, moving the GRU closer to the output layer produced a model with better predicted values.

Keywords Data-driven modeling · F-16 fighter jet · Recurrent neural networks · Angle of attack

Introduction

Recent accidents of the Boeing 737 MAX show that it could accidentally make a nosedive that can cause it to crash. This nosedive was ultimately caused by a failure of the angle of attack (AOA) sensor [1]. Also, in other fixed-wing aircraft, the AOA is a critical parameter because all aerodynamic forces are functions of the AOA. It is the most vital safety parameter in preventing stall situations [2]. The stall is a function of the angle of attack (Table 1). An accelerated stall catches the pilot by surprise because while the aircraft's air-speed is above stall speed, the AOA has increased beyond the angle for which the plane can produce lift [3].

Most commercial aircraft are aerodynamically stable; they can recover from stalls automatically. In contrast, fighter jets like the F-16 are inherently unstable to make them more maneuverable. It is all the more essential to make the pilot situationally aware of the AOA in such fighter jets. There-

✉ Bemnet Wondimagegnehu Merasha
MerashaBemnetW@bit.edu.cn

David N. Jansen
dnjansen@ios.ac.cn

Hongbin Ma
mathmhb@bit.edu.cn

¹ School of Automation, Beijing Institute of Technology, Beijing, China

² Institute of Software, Chinese Academy of Sciences, Beijing, China

³ State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing, China

⁴ State Key Laboratory of Intelligent Control and Decision of Complex Systems Beijing Institute of Technology, Beijing, China

Table 1 List of acronyms

Adam	Adaptive momentum estimation algorithm
AOA	Angle of attack
CUDA	Compute unified device architecture
DOF	Degree of freedom
EKF	Extended Kalman filter
EOM	Equations of motion
ESPRIT	Estimation of signal parameters via rotational invariance techniques
GPS	Global positioning system
GRU	Gated recurrent unit
IMU	Inertial measurement unit
MIMO	Multiple input multiple output
MSE	Mean squared error
MUSIC	Multiple signal classification
NASA	National Aeronautics and Space Administration
RMSprop	Root mean square propagation
RNN	Recurrent neural networks
LSTM	Long-short-term-memory
SGD	Stochastic gradient descent
sRNN	Simple recurrent neural network
SVD	Singular value decomposition
UAV	Unmanned aerial vehicles

fore, it is crucial to have a method that measures the AOA of the aircraft even in flight conditions with faulty sensors.

Most of the literature focuses on AOA estimation and does not provide any information on the performance of their methods in the presence of faulty sensors. AOA estimation using Kinematic models together with Global positioning system (GPS), inertial measurement unit (IMU), and air-speed sensors is proposed by [4]. In [5], Extended Kalman Filter (EKF) is used to estimate the AOA using GPS, IMU, and Pitot tube measurements. In [6], Adaptive Kalman Filter is used to estimate the AOA. In [7], GPS receiver, an inertial reference system and a sensor measurement for the static and dynamic pressure are used to estimate the AOA. In [8], a neural network-based method is used to estimate the AOA. The data used for the neural network estimation of the AOA in [8] are static data. In [9], the Output-Error method with Maximum-Likelihood Estimation is used to estimate the AOA.

Most of the research on the estimation of AOA uses sensor information and provides only a limited assessment of the AOA estimation when the sensor measurements are faulty or are noisy. The AOA estimation during faulty AOA measurement, faulty total velocity measurement, and faulty total velocity and pitch rate measurement has not been addressed in detail. The literature considered using statistical methods

like the maximum likelihood method [9], but this method is sensitive to biases. In the literature, recurrent neural networks (RNN) and statistical methods like multiple signal classification (MUSIC) [10] and estimation of signal parameters via rotational invariance techniques (ESPRIT) [11] has not been used for AOA estimation in case of faulty sensors.

Furthermore, most literature uses linear models like the Kalman Filter for AOA estimation [5,6,9]. Nonlinearities have a significant impact on dynamic responsiveness in modern high-performance aircraft. Fighter aircraft, for example, may fly at high angles of attack, where the lift coefficient is difficult to describe as a linear function of angle of attack, or at high roll rates, where nonlinear, inertial cross-coupling can cause instabilities [12].

The AOA is usually measured with an angle of attack sensor, although faulty AOA sensors have caused numerous accidents in the past [1,13–16]. The AOA measurement is also affected by the total velocity and pitch rate measurements. In many fixed-winged aircraft, the total velocity is measured using an airspeed indicator that uses a pitot-static system. Errors in speed measurement may occur if the pitot tube is blocked because of dirt or any other object. Pitch rate measurement is carried out using IMU. Qantas Airbus A330-303 Flight QF72 accident investigation shows that the cause is erroneous data from IMU [17]. Using a neural network-based AOA estimation has advantages because the AOA sensor must be positioned in a clear aerodynamic environment to measure the angle of attack correctly [8].

Using redundant sensors to measure the same parameter simultaneously and a voting mechanism to correct faulty sensor measurements is one of the most popular ways to cope with faulty sensor measurements [18,19]. Since redundancy is unable to manage system noise and certain other disadvantages in terms of cost, weight, power consumption, and size, this technique will constantly encounter issues [20]. Furthermore, it is vulnerable to all redundant sensors simultaneously failing when exposed to the same severe environment. The best method to cope with faulty sensor measurements is to create trustworthy virtual sensors to replace them. The approach focuses on detecting a problem in the system's fault status, locating the faulty sensor, and then replacing the incorrect data with other reliable data (also known as accommodation) [20,21]. Accommodation is the final stage in the process of substituting defective sensor data with reliable data. The accommodation of AOA sensor readings in the face of noise and faulty sensor measurements is the subject of this research.

Data-driven modeling is gradually replacing traditional methods of modeling systems. Traditional dynamical modeling methods such as Newtons, Lagrangian, and Hamiltonian are still useful, but when the system is very complex and nonlinear, it is challenging to develop a model using traditional methods. Modeling dynamical systems using data-driven

techniques can be used for parameter estimation, state prediction, and control. Machine-learning methods like RNN are used when modeling dynamical systems. Most of the time, RNNs and their variants are used for speech recognition, time-series optimization problems, and dynamic modeling [22–24]. Previous research shows that RNNs are suitable to model complex dynamical systems [25]. AOA measurement based on data-driven techniques has the advantage that it can detect faults and changes related to structural damages [26]. It also helps in the development of accurate control methods [27].

The following are the significant contributions of this research:

1. We used the F-16 fighter jet's nonlinear model in conjunction with wind-tunnel data from NASA's Ames and Langley research center. The AOA estimate technique uses a dynamical system model as well as real-world data from experiments.
2. The majority of AOA estimation literature [5,6,9] uses an extended Kalman filter which works for some nonlinear dynamical systems that can be represented by linear operations but are not well suited when the system is complex and nonlinear. We have used RNN, which can model complex nonlinear dynamical systems [25].
3. We were able to extract various non-monotonic data matrices using our data collecting and processing technique. Our data collection mechanism also ensures that the data is in the operational region of the aircraft.
4. We compared various RNN models for AOA estimation in the presence of noise and faulty sensors. The literature [4–9] has not considered estimating AOA under flight conditions with faulty AOA, total velocity, or pitch rate measurements.

The organization of this paper is as follows: in the next section, the basic structures of the three RNNs are explained; in the third section, the algorithms used are explained briefly; in the fourth section, the equations of motion (EOM) of the F-16 fighter jet is explained, in the fifth section, the experimental setup is explained, and finally, the results obtained are analyzed.

Recurrent neural networks (RNN)

RNNs are a type of feedforward neural network that can transfer information across time steps. They are a diverse group of models capable of almost any calculation. RNNs have strong hidden layers coupling, enabling them to learn complex nonlinear patterns effectively compared to traditional multilayer perceptrons [25]. A generic recurrent neural network, with input X_t , hidden state h_t for time step t , is given by

Fig. 1 Architecture of sRNN

$$h_t = \mathcal{F}(h_{t-1}, X_t), \quad (1)$$

where $\mathcal{F}$ is a nonlinear activation function that controls recurrence and thus the recurrence function. $\mathcal{F}$ can be as simple as an element-wise logistic sigmoid function as in simple recurrent neural networks (sRNN) or as complicated as in LSTM and gated recurrent unit (GRU). The detailed mathematical derivation and training method of RNN is described in detail in [28]. In the following subsections, the basic difference between the RNN architectures used in this paper is described.

Simple recurrent neural networks (sRNN)

The sRNN transforms a sequence of input to a sequence of output. The architecture of the sRNN is shown in Fig. 1. In the figure, X_t is the current input vector at time t , h_t is a chunk of hidden layer nodes representing the state of the network, and o_t is the output vector. The sRNN accepts an input vector and returns an output vector. However, the output vector values are influenced by the current input and the whole history of previous inputs. This behavior enables it to model several dynamical systems. The output of dynamical systems depends not only on the current input but also on the past history of inputs that are applied.

The sRNN is described by the following equation:

$$\begin{aligned} h_t &= W_{\text{rec}}\sigma(h_{t-1}) + W_{\text{in}}X_t + b, \\ o_t &= g(h_t). \end{aligned} \quad (2)$$

Here, X_t is the current input vector at time t , h_t is the hidden state, o_t is the output, W_{rec} is the recurrent weight matrix, W_{in} is the input weight matrix, b is the bias, σ is element-wise function, $g(\cdot)$ is a user defined function to calculate the output and h_0 is provided by the user.

LSTM (long-short-term memory)

When using back-propagation through time algorithm to train RNNs, the major problem faced is the problem of vanishing gradients or exploding gradients. The exploding gradients problem refers to the large increase in the

Fig. 2 Architecture of LSTM [32]

norm of the gradient during training. Such events are due to the explosion of the long-term components, which can grow exponentially more than short-term ones. The vanishing gradients problem refers to the opposite behavior, when long-term components go exponentially fast to zero norms, making it impossible for the model to learn the correlation between temporally distant events [29]. The main problem is addressed by the research conducted by [29–31].

LSTMs solve this problem by ensuring constant error flows through constant error carousels. These constant error flows are regulated by nonlinear multiplicative units that learn to open and close gates in the network (see Fig. 2).

The memory cell, or simply the cell, is the computational unit of the LSTM network. Weights and gates are the components of LSTM cells. The gates are the memory cell's most important component. These are also weighted functions that control the flow of information in the cell. There are three gates: the forget gate, the input gate, and the output gate. The internal state is updated using the forget gate and input gate. The forget gate determines which information from the cell should be discarded. The input gate determines which input values should be used to update the memory state (Table 2). The output gate is a final check on the cell's output. Based on the input and the cell's memory, the output gate determines what to output. The following equation describes the input gate, forget gate, output gate, and cell state:

$$\begin{aligned} i_t &= \sigma(W_{x_i}X_t + W_{h_i}h_{t-1} + W_{c_i}c_{t-1} + b_{i_i}) \\ f_t &= \sigma(W_{x_f}X_t + W_{h_f}h_{t-1} + W_{c_f}c_{t-1} + b_{i_f}) \\ c_t &= f_t \odot c_{t-1} + i_t \odot [\tanh(W_{x_c}X_t + W_{h_c}h_{t-1} + W_{c_c}c_{t-1} + b_c)] \\ o_t &= \sigma(W_{x_o}X_t + W_{h_o}h_{t-1} + W_{c_o}c_{t-1} + b_{i_o}) \\ h_t &= o_t \odot \tanh(c_t) \end{aligned} \quad (3)$$

Table 2 Symbols for the LSTM equations

i_t	The input gate
f_t	The forget gate
o_t	The output gate
c_t	The cell state
h_t	The hidden state
X_t	The current input vector
b_{i_i}	The bias at the input gate
b_{i_f}	The bias at the forget gate
b_{i_o}	The bias at the output gate
b_c	The bias at the cell
W_{x_i}	The weight matrix from the current input to each of the gates
W_{h_i}	The recurrent weight matrix from the hidden state to each of the gates
W_{c_i}	The peephole weight vectors from the cell to each of the gates
σ	Sigmoid activation function applied in Schur product (element wise)
$\odot$	Element-wise vector multiplication

Gated recurrent units (GRU)

GRUs are a variation on LSTM introduced by [22]. GRUs contain two RNN: the encoder and the decoder part. One RNN encodes a source sequence into a fixed-length vector representation, and the other RNN transforms the representation into a variable-length sequence while being jointly trained to maximize the conditional probability of a target sequence given an input sequence [22]. The encoder is an RNN that reads each element in the input sequence X in a sequential manner. The RNN's hidden state after reading the end of the sequence is a summary $\mathbf{c}$ of the entire input sequence.

The decoder is another RNN which is trained to generate the output sequence $\mathbf{o}_t$ by predicting the next sequence value given the hidden state $\mathbf{h}_t$. Unlike the simple RNN mentioned in "Simple recurrent neural networks (sRNN)", however, both $\mathbf{o}_t$ and $\mathbf{h}_t$ are conditioned on $\mathbf{o}_{t-1}$ and the input sequence summary $\mathbf{c}$.

The hidden state $\mathbf{h}_t$ is calculated from the summary of input sequence $\mathbf{c}$, previous hidden state $\mathbf{h}_{t-1}$, and previous output $\mathbf{o}_{t-1}$. The decoder predicts the output sequence based on the previous output $\mathbf{o}_{t-1}$, the hidden state $\mathbf{h}_t$, and the summary of the encoded inputs $\mathbf{c}$:

$$\begin{aligned} \mathbf{h}_t &= f(\mathbf{h}_{t-1}, \mathbf{y}_{t-1}, \mathbf{c}) \\ \mathbf{P}(\mathbf{o}_t | \mathbf{o}_{t-1}, \mathbf{o}_{t-2}, \dots, \mathbf{c}) &= g(\mathbf{h}_t, \mathbf{o}_{t-1}, \mathbf{c}) \end{aligned} \quad (4)$$

Here, $f(\cdot)$ and $g(\cdot)$ are activation functions. $g(\cdot)$ must produce appropriate probabilities.

Table 3 Symbols for the GRU equations

r_t	The reset gate
z_t	The update gate
$\tilde{h}$	The hidden state candidate
b_r	The bias at the reset gate
b_z	The bias at the update gate
$b_{\tilde{h}}$	The bias at the hidden state candidate
W_{x_r}	The weight matrix from the current input to the reset gates
W_{x_z}	The weight matrix from the current input to the update gate
$W_{x_{\tilde{h}}}$	The weight matrix from the current input to the hidden state candidate
W_{h_r}	The weight matrix from the hidden state to the reset gates
W_{h_z}	The weight matrix from the hidden state to the update gate
$W_{r_{\tilde{h}}}$	The weight matrix from the reset gate to the hidden state candidate

The GRU has a new type of hidden unit recurrence, which is motivated by LSTM. LSTM uses cells as a memory. GRU, on the other hand, does not have a cell state like LSTM. Furthermore, it just has two gates: a reset gate and an update gate. When the reset gate is near 0, the hidden state is forced to ignore the prior hidden state and reset with the current input only in this formulation. This effectively permits the hidden state to discard any information that will become unnecessary in the future, resulting in a more compact representation (Table 3). The update gate, on the other hand, regulates how much information from the previous hidden state is transferred to the current hidden state:

$$\begin{aligned}
 r_t &= \sigma(W_{x_r} X_t + W_{h_r} h_{t-1} + b_r) \\
 \tilde{h}_t &= \tanh(W_{x_{\tilde{h}}} X_t + W_{r_{\tilde{h}}} r_t \odot h_{t-1} + b_{\tilde{h}}) \\
 z_t &= \sigma(W_{x_z} X_t + W_{h_z} h_{t-1} + b_z) \\
 h_t &= (1 - z_t) h_{t-1} + (z_t \tilde{h}_t).
 \end{aligned} \quad (5)$$

Training algorithms used for the experiment

Most algorithms that are used for training deep neural networks are derived from the gradient descent algorithm, which can be easily applied to any deep neural network with some tricks [33]. Training a neural network in the case of supervised learning is accomplished using an input–output dataset pair. The training algorithm adapts the weights of the neural network to make the predicted outputs ($\hat{y}_i(k)$) “close” to the desired outputs ($y_i(k)$). k denotes the number of iteration. This “closeness” of the predicted outputs to the desired outputs is measured by a loss function (cost function) $l(\hat{y}, y)$.

Table 4 Symbols for the F-16 fighter jet EOM

u	The velocity along X body axis
v	The velocity along Y body axis
w	The velocity along Z body axis
V_t	The total velocity
α	The angle of attack
β	The angle of side slip
p	The roll rate about X body axis
q	The pitch rate about Y body axis
r	The yaw rate about Z body axis
θ, ϕ, ψ	The Euler angles
I_X, I_Y, I_Z	The moment of inertia about X, Y and Z body axes
I_{XZ}	The product of inertia with respect to X and Z body axes
$\bar{q}$	The free-stream dynamic pressure
S	The wing area
m	The mass
T	The total instantaneous engine thrust
b	The wing span
$\bar{c}$	The wing mean aerodynamic chord
H_e	The engine angular momentum
g	The acceleration due to gravity
X_E, Y_E, Z_E	The X, Y and Z position with respect to the Earth
$C_{X,t}, C_{Y,t}, C_{Z,t}$	The total X, Y and Z force aerodynamic coefficients
$C_{l,t}$	The total rolling-moment aerodynamic coefficient
$C_{m,t}$	The total pitching-moment aerodynamic coefficient
$C_{n,t}$	The total yawing-moment aerodynamic coefficient

The loss function used in this paper is the mean squared error (MSE) shown in the following equation:

$$l(\hat{y}, y) = \sum_{k=1}^K E(k) = \frac{1}{2n} \sum_{k=1}^K \sum_{i=1}^n \| \hat{y}_i(k) - y_i(k) \|^2. \quad (6)$$

Here, n is the total number of outputs. One of the most widely used gradient descent back-propagation algorithms is the stochastic gradient descent (SGD) algorithm [34]. In this paper, two commonly used variations of stochastic gradient descent optimizing algorithms: root mean square propagation (RMSprop) [34] and adaptive momentum estimation (Adam) [35], are compared.

F-16 fighter jet equation of motion

F-16 fighter jet’s EOM is presented below. The model is a 6-Degree of freedom (DOF) rigid body equation referenced to a body-fixed axis system. During modeling, the following assumptions are made: the aircraft is a rigid body, the earth is considered to be flat, the mass of the aircraft is assumed to

be constant, the mass distribution of the aircraft is symmetric to the XZ plane of the body axis reference frame (Table 4). The flight EOM is from [36,37].

Force equations:

$$\begin{aligned}\dot{u} &= rv - qw - g \sin \theta + \frac{\bar{q}S}{m} C_{X,t} + \frac{T}{m} \\ \dot{v} &= pw - ru - g \cos \theta \sin \phi + \frac{\bar{q}S}{m} C_{Y,t}\end{aligned}\quad (7)$$

$$\begin{aligned}\dot{w} &= qu - pv + g \cos \theta \cos \phi + \frac{\bar{q}S}{m} C_{Z,t} \\ V_t &= \sqrt{u^2 + v^2 + w^2} \\ \alpha &= \tan^{-1} \left(\frac{w}{u} \right)\end{aligned}\quad (8)$$

$$\begin{aligned}\beta &= \sin^{-1} \left(\frac{v}{V_t} \right) \\ \dot{V}_t &= \frac{u\dot{u} + v\dot{v} + w\dot{w}}{V_t} \\ \dot{\alpha} &= \frac{u\dot{w} - w\dot{u}}{(V_t \cos \beta)^2} \\ \dot{\beta} &= \frac{V_t \cos \beta \dot{v} - v \cos \beta \dot{V}_t}{(V_t \cos \beta)^2}\end{aligned}\quad (9)$$

Moment equations:

$$\begin{aligned}\dot{p} &= \frac{I_Y - I_Z}{I_X} qr + \frac{I_{XZ}}{I_X} (\dot{r} + pq) + \frac{\bar{q}Sb}{I_X} C_{l,t} \\ \dot{q} &= \frac{I_Z - I_X}{I_Y} pr + \frac{I_{XZ}}{I_Y} (r^2 - p^2) + \frac{\bar{q}S\bar{c}}{I_Y} C_{m,t} - H_e r \\ \dot{r} &= \frac{I_X - I_Y}{I_Z} pq + \frac{I_{XZ}}{I_Z} (\dot{p} - qr) + \frac{\bar{q}Sb}{I_Z} C_{n,t} - H_e q\end{aligned}\quad (10)$$

Kinematic equations:

$$\begin{aligned}\dot{\phi} &= p + \tan \theta (q \sin \phi + r \cos \phi) \\ \dot{\theta} &= q \cos \phi - r \sin \phi \\ \dot{\psi} &= \frac{q \sin \phi + r \cos \phi}{\cos \theta}\end{aligned}\quad (11)$$

Navigation equations:

$$\begin{aligned}\dot{X}_E &= u \cos \psi \cos \theta + v (\cos \psi \sin \theta \sin \phi - \sin \psi \cos \phi) \\ &\quad + w (\cos \psi \sin \theta \cos \phi + \sin \psi \sin \phi) \\ \dot{Y}_E &= u \sin \psi \cos \theta + v (\sin \psi \sin \theta \sin \phi + \cos \psi \cos \phi) \\ &\quad + w (\sin \psi \sin \theta \cos \phi - \cos \psi \sin \phi) \\ \dot{Z}_E &= -u \sin \theta + v \cos \theta \sin \phi + w \cos \theta \cos \phi\end{aligned}\quad (12)$$

The aerodynamic force and moment coefficients determine the drag, lift, side-forces, rolling moment, pitching

moment, and yawing moment of the F-16 fighter jet. In this article, we used aerodynamic coefficients and wind-tunnel data from [37]. The control inputs are the thrust, elevator, aileron, and rudder. Maneuvering the F-16 fighter jet is accomplished by changing the aerodynamic forces and moments using the elevator, aileron, and rudder.

In this article, we used the positions X_E, Y_E, Z_E , the roll angle (ϕ), the pitch angle (θ), the yaw angle (ψ), the total velocity (V_t), the angle of attack (α), the angle of side-slip (β), the roll rate (p), the pitch rate (q) and the yaw rate (r) as the state of the F-16 fighter jet. We used the accelerations normalized by g in the x, y, z directions (a_{nx}, a_{ny}, a_{nz}), the Mach number, the free-stream dynamic pressure ($\bar{q}$) and the static pressure (P_s) as the output of the F-16 fighter jet. We used the thrust, the elevator, the aileron and the rudder as the input of the F-16 fighter jet. Our desire is to develop a data-driven model of the plant that can be used not only for prediction of states but also for control.

Experiment setup

Figure 3 describes the schematics of this research. In the following subsections, we describe the data collection, the network setup, and the carried out experiments.

Data collection and preprocessing

The data are collected from the F-16 simulation developed by [38]. The data include joint lateral and longitudinal dynamics of the model, allowing a multi-dimensional analysis. The aerodynamic data used in the simulation are derived from low-speed static and dynamic (force oscillation) wind-tunnel test conducted with sub-scale models of the F-16 wind-tunnel facilities at the NASA Ames and Langley Research Centers [38]. The navigation equation for the model is taken from [36]. The equations for the aerodynamic force and moment coefficients are taken from [37].

The model incorporates all the kinematic and aerodynamic characteristics of real F-16 aircraft [38]. The flight dynamic, aerodynamic coefficients, and wind-tunnel data used can be found in [37]. We used the simulation model with the wind-tunnel data as our real plant. Another alternative for collecting data is to use small fixed-wing unmanned aircraft, but it was shown that this method does not work well because the small fixed-wing unmanned aerial vehicles (UAV) are susceptible to atmospheric disturbances and are equipped with lower-quality instrumentation, which gives noisier measurements that do not provide enough information for modeling [39]. We have chosen to use a simulation model developed from wind-tunnel tests because of the high cost of generating data by maneuvering around with a real F-16. To defend our choice further, we stress that the data were not just generated

Fig. 3 Schematic of the research

from a set of abstract equation of motions but from physical system that has real aerodynamic characteristics of the F-16 fighter jet.

The data are collected using several combinations of doublet control inputs (elevator, aileron and rudder) following the recommendation by [40]. The thrust input is kept at the trim value while collecting data. The data are collected at trimmed total velocity and altitude (V_t and h) in a steady

wings-level flight condition. A doublet input is applied at the trimmed condition, and 22 parameters are measured for 40 s in a time step of 0.1 s. This procedure is repeated at various trimmed altitudes and heights. The collected parameters are all the states, the outputs and the inputs of the F-16 fighter jet. The first row of every data matrix is the initial condition (trimmed value). More than 600 data matrices of size 401×22 are collected. The data collected are from an

altitude ranging from 5000 to 40,000 feet and total velocity ranging from 300 to 900 feet/s.

It is important to process the raw dataset before training. The collected dataset is processed using standardization method. The singular value decomposition (SVD) is used to check the data collection is not monotonous [41]. The plot of the Singular values and the cumulative Singular values is shown in Fig. 4.

Network setup

The sRNN, LSTM, GRU and the combinations LSTM-GRU, sRNN-GRU and sRNN-LSTM are evaluated to model the F-16 fighter jet in the steady winged condition. During the training, the data is divided in a 60% : 40% ratio for training and validation purposes. During the training phase, we used the validation dataset for the early stopping technique. The early stopping technique stopped the training when the training error became smaller than the validation error. This method uses cross-validation and prevents the network's weight from overfitting to the training data. The networks are implemented using an input layer, three hidden layers, and a linear output layer. A time step of 10 is used for all networks. The input to the networks is the combination of all the states and the input of F-16 fighter jet. The desired output of the network is the combination of the states and the outputs of the F-16 fighter jet.

The training is carried out using root mean square propagation (RMSprop) and adaptive momentum estimation (Adam) algorithms. The batch size for training is set to 400 because it takes 400 samples for each transient oscillation to be dampened. In addition to this, dropout regularization of 20% is used so that the training will not over-fit the training dataset.

Description of the experiment

All the experiments are carried out using CUDA-enabled GeForce-RTX2070 Super with compute capability of 7.5. All the networks are trained multiple times for the experiments. The experiments are:

1. First experiment In the first experiment, only one of the three RNNs is used for the hidden layer (i.e., all the three hidden layers are set up to be only LSTMs or GRUs or sRNNs with 20% dropout regularization). In this experiment, the networks are trained using the training dataset prepared by considering noisy sensor assumptions. During the noisy sensor data assumption, a Gaussian noise with zero mean and variance of 0.15 is added to the whole dataset to simulate adverse flight condition that results in a noisy sensor measurements. A standard deviation of 0.15 is greater than sensor measurement noise in normal flight conditions. We chose a standard deviation of 0.15 to sim-

ulate a flight condition where all sensor measurements are highly affected by noise that is greater than the normal flight condition sensor noise. During this test, all the networks are evaluated by their training MSE, validation MSE and the time it takes to train the network.

2. Second experiment In the second experiment, the AOA prediction capability of the trained networks in the first experiment is further evaluated using a noisy test dataset input that is assumed to have a faulty sensor. In the first test, the total velocity measurement is assumed to be faulty; in the second test, both the total velocity and the pitch rate measurement are assumed to be faulty and in the third test, the AOA measurement is assumed to be faulty. We chose to evaluate the trained network under faulty total velocity measurement and faulty pitch rate measurement because the total velocity and the pitch rate measurement are very essential when evaluating the angle of attack of an aircraft. When the sensors are assumed to be faulty, the feature of the values in the test dataset is replaced by Gaussian noise. For each of the above tests, correlation and MSE between the predicted AOA and target AOA is used to evaluate the trained networks' ability to give good AOA predicted values in flight conditions with faulty sensor measurements.
3. Third experiment Finally, LSTM-GRU, sRNN-GRU, and sRNN-LSTM combined networks are trained using the training dataset prepared with noisy sensor assumption, and their AOA prediction ability is tested by a noisy test dataset input that is assumed to have a faulty sensor. The three sensor faults considered in the second experiment is used to evaluate the combined trained networks. We evaluated various combinations of LSTM-GRU, sRNN-LSTM, and sRNN-GRU combined networks. In this article, we only present the result of the sRNN-GRU combined networks because one of the sRNN-GRU combined networks showed better performance compared to the other networks when used to predict AOA in flight conditions with faulty sensor measurements.

Result and analysis

First experiment: training result

The results of the training and validation for the first experiment have been summarized in Tables 5, 6 and 7.

The sRNN takes more time to train, has higher training and validation MSE than the rest of the networks. Although the sRNN has higher training and validation MSE compared to the other networks, the validation accuracy is very close to the GRU and the LSTM networks. The highest validation accuracy for the sRNN is 92%, for the LSTM is 92% and for the GRU is 94%. This shows that although the sRNN takes

Fig. 4 Checking the data collected is not monotonous **Table 5** sRNN training and validation

Algorithm	Training	Validation	Training runtime (s)
RMSprop	Error	Error	112
	18×10^{-4}	18×10^{-04}	
	Accuracy	Accuracy	
Adam	82%	88%	140
	Error	Error	
	18×10^{-4}	18×10^{-04}	
	Accuracy	Accuracy	
	82%	92%	

Table 6 LSTM training and validation

Algorithm	Training	Validation	Training runtime (s)
RMSprop	Error	Error	54
	15×10^{-4}	19×10^{-04}	
	Accuracy	Accuracy	
Adam	84%	92%	75
	Error	Error	
	15×10^{-4}	14×10^{-04}	
	Accuracy	Accuracy	
	85%	92%	

Table 7 GRU training and validation

Algorithm	Training	Validation	Training runtime (s)
RMSprop	Error	Error	24
	15×10^{-4}	15×10^{-04}	
	Accuracy	Accuracy	
Adam	83%	92%	40
	Error	Error	
	15×10^{-4}	12×10^{-04}	
	Accuracy	Accuracy	
	86%	94%

Fig. 5 sRNN trained with Adam training and validation error

a longer time to train, its validation accuracy is close to the other networks.

The validation error in some of the networks is smaller than the training error. In this experiment, the reason for getting a smaller validation error than the training error can be attributed to the use of dropout regularization (Fig. 5). Applying regularization helps the model not to over-fit the dataset during training, but it might also make the validation loss less than the training loss.

Second experiment: AOA prediction with faulty sensor assumption result

The result is shown in Table 8. During the faulty total velocity measurement test and faulty total velocity and pitch rate measurement test, the sRNN performs better than the other networks (see Table 8). In the first experiment, the sRNN did not perform well compared to the other networks as it can be seen from the training MSE results (in Tables 5, 6 and

Table 8 Linear regression result with the assumption of sensor failure

Algorithm and test	LSTM	GRU	sRNN
RMSprop no faulty measurement but highly noisy	Correlation	Correlation	Correlation
	0.99	0.99	0.97
	MSE (deg)	MSE (deg)	MSE (deg)
	0.36	0.84	1.11
RMSprop faulty total velocity measurement test	Correlation	Correlation	Correlation
	0.81	0.92	0.94
	MSE (deg)	MSE (deg)	MSE (deg)
	10.3	2.5	3.0
RMSprop faulty total velocity and pitch rate measurement test	Correlation	Correlation	Correlation
	0.73	0.92	0.96
	MSE (deg)	MSE (deg)	MSE (deg)
	12.8	2.8	2.4
RMSprop faulty AOA measurement test	Correlation	Correlation	Correlation
	0.98	0.94	0.84
	MSE (deg)	MSE (deg)	MSE (deg)
	1.11	3.4	8.4
Adam no faulty measurement but highly noisy	Correlation	Correlation	Correlation
	0.99	0.99	0.97
	MSE (deg)	MSE (deg)	MSE (deg)
	0.15	0.3	1.12
Adam faulty total velocity measurement test	Correlation	Correlation	Correlation
	0.97	0.98	0.98
	MSE (deg)	MSE (deg)	MSE (deg)
	3.0	0.64	0.61
Adam faulty total velocity and pitch rate measurement test	Correlation	Correlation	Correlation
	0.92	0.91	0.96
	MSE (deg)	MSE (deg)	MSE (deg)
	5.04	2.84	1.80
Adam faulty AOA measurement test	Correlation	Correlation	Correlation
	0.95	0.88	0.83
	MSE (deg)	MSE (deg)	MSE (deg)
	2.3	5.3	7.0

7), but in the faulty measurement tests in the second experiment, it gives the best approximation of AOA during faulty total velocity measurement test and faulty total velocity and pitch rate measurement test. During the faulty total velocity and pitch rate measurement test, the correlation between the predicted value and the target for the sRNN trained with RMSprop is 0.96, for the GRU trained with RMSprop is 0.92 and the LSTM trained with RMSprop is 0.73. However, the sRNN has the worst prediction during the faulty AOA measurement test (see Fig. 6).

The predicted value of the LSTM has the worst correlation and MSE value during the faulty total velocity measurement test and faulty total velocity and pitch rate measurement test. However, the LSTM has the best prediction during the faulty

AOA measurement test. The MSE between the predicted value and the target during the faulty AOA measurement test is 1.11 degrees for the LSTM trained with RMSprop, 3.4 degrees for the GRU trained with RMSprop, and 8.4 degrees for the sRNN trained with RMSprop.

The GRU has an average performance in all the three faulty sensor tests. This results can be attributed to the architecture of the GRU. The GRU is an encoder–decoder RNN. The input encoding of RNN is a positive factor when used in environments or situations that result in sensor failure.

In almost all network faulty sensor tests, training with Adam gives lesser MSE between the predicted and target values than training with RMSprop. This is because Adam has all the attributes of RMSprop and it also has a momentum

Fig. 6 Predicted and real value of AOA using sRNN trained with Adam

Fig. 7 Predicted and real value of AOA using Network 1 trained with Adam

component. RMSprop contributes the exponential decaying average of past squared gradients, while the momentum accounts for the exponentially decaying average of past gradients [34].

Third experiment: training and AOA prediction result of the combined networks

We evaluated various combinations of LSTM-GRU, sRNN-LSTM and sRNN-GRU combined networks. But here, we present the result for the sRNN-GRU combined networks because one of the sRNN-GRU combined networks showed better performance. Table 9 shows the sRNN-GRU combinations that are evaluated and Table 10 shows the result of faulty sensor measurement tests of the second experiment for the sRNN-GRU combined networks.

Table 9 sRNN-GRU combined architecture

Network 1	Network 2	Network 3
Layer 1:Input	Layer 1:Input	Layer 1:Input
Layer 2:sRNN	Layer 2:sRNN	Layer 2:GRU
Layer 3:sRNN	Layer 3:GRU	Layer 3:sRNN
Layer 4:GRU	Layer 4:sRNN	Layer 4:sRNN
Layer 5:Linear (Output layer)	Layer 5:Linear (Output layer)	Layer 5:Linear (Output layer)

During the faulty sensor tests on the sRNN-GRU combined architecture, Network 1 (see Table 10) showed good results with lower MSE values and higher correlations. Comparing the result for the combined networks in Table 10 and the results in Table 8 during the faulty AOA measurement test, Network 1 has a correlation value 95% and MSE of 2.3 degrees (see Fig. 7), the sRNN has a correlation value 83% and MSE of 7.0 degree and the GRU has correlation value 88% and MSE of 5.3 degrees. Moreover, the sRNN-GRU combined architecture performed relatively better than the sRNN and the GRU during the faulty velocity measurement test and the faulty velocity and pitch rate measurement test.

The effectiveness of the sRNN-GRU combined architecture results from the individual strengths of the sRNN and the GRU. The sRNN uses Eq. (2) to come up with a state to represent the input-output dataset. The GRU is an encoder-decoder network where one RNN encodes and the other RNN decodes according to Eqs. (4) and (5). Therefore, instead of applying encoding-decoding in each layer as the network with three-layer GRU does, Network 1 uses encoding-decoding just before the linear output layer. This gives Network 1 an advantage over the other networks. Although sRNN-GRU combined architecture performs well in the faulty measurement tests, it takes a longer time (150 s) to train compared to the LSTM and the GRU.

Conclusion

This work has shown F-16 aircraft modeling using a recurrent neural network and its use for AOA prediction with faulty sensor measurements. F-16 aircraft model is developed using sRNN, LSTM, GRU and LSTM-GRU, sRNN-LSTM and sRNN-GRU combined networks. The result obtained shows F-16 fighter jet can be modeled using RNNs, and the developed model can be effectively used for predicting AOA in flight conditions with a faulty sensor. The sRNN-GRU combination is an effective network that can be used in flight conditions with a faulty sensor. The architecture can be used for predictive control techniques.

Table 10 sRNN-GRU faulty sensor measurement test result

Test	Network 1	Network 2	Network 3
No faulty measurement but highly noisy	Correlation	Correlation	Correlation
	0.99	0.99	0.97
	MSE (deg)	MSE (deg)	MSE (deg)
	0.2	0.3	0.4
Faulty total velocity measurement test	Correlation	Correlation	Correlation
	0.98	0.98	0.98
	MSE (deg)	MSE (deg)	MSE (deg)
	1.0	0.6	0.5
Faulty total velocity and pitch rate measurement test	Correlation	Correlation	Correlation
	0.94	0.95	0.92
	MSE (deg)	MSE (deg)	MSE (deg)
	3.05	2.0	2.2
Faulty AOA measurement test	Correlation	Correlation	Correlation
	0.95	0.88	0.83
	MSE (deg)	MSE (deg)	MSE (deg)
	2.3	5.3	7.0

The AOA estimation in this work employs RNN to estimate AOA under steady flight conditions with noisy and faulty sensors. Furthermore, the method we use efficiently illustrates that AOA can be estimated utilizing properly functioning sensors. However, neural network-based techniques have the disadvantage of requiring a big dataset and lack interpretability because it is a black-box model. In addition to this, the study does not address the detection and isolation of faulty sensors. The next step in this study will be to develop algorithms that use fewer datasets and can estimate AOA under flying situations with faulty sensors. MUSIC, ESPRIT, and Koopman [42] are all viable options. Furthermore, we will also be working on the detection and isolation of defective sensors in the future.

Declarations

Conflict of interest On behalf of all the authors, the corresponding author states that there is no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. FAA (2020) Summary of the FAA's Review of the Boeing 737 MAX: Return to Service of the Boeing 737 MAX Aircraft
2. Jackson B, Hoffler KD, Sizoo DG, Ryan W (2017) Experience with sensed and derived angle of attack estimation systems in a general aviation airplane. In: AIAA information systems-AIAA Infotech @ Aerospace. American Institute of Aeronautics and Astronautics, p 0065
3. Ly JK (2017) Angle of attack determination using inertial navigation system data from flight tests. Master's thesis, University of Tennessee
4. Langelaan JW, Alley N, Neidhoefer J (2011) Wind field estimation for small unmanned aerial vehicles. *J Guid Control Dyn* 34(4):1016–1030
5. Cho A, Kim J, Lee S, Kee C (2011) Wind estimation and airspeed calibration using a UAV with a single-antenna GPS receiver and pitot tube. *IEEE Trans Aerosp Electron Syst* 47(1):109–117
6. Seren C, Ezerzere P, Hardier G (2015) Model-based techniques for virtual sensing of longitudinal flight parameters. *Int J Appl Math Comput Sci* 25(1):23–38
7. Popowski S, Dabrowski W (2015) Measurement and estimation of the angle of attack and the angle of sideslip. *Aviation* 19(1):19–24
8. Lerro A, Battipede M, Gili P, Brandl A (2017) Advantages of neural network based air data estimation for unmanned aerial vehicles. *Int J Mech Aerosp Ind Mechatron Manuf Eng* 11(5):1016–1025
9. Lichota P, Lasek M (2013) Maximum likelihood estimation: a method for flight dynamics-angle of attack estimation. In: Proceedings of the 14th international Carpathian control conference (ICCC). IEEE, pp 218–221
10. Bestetti M (2018) Performance analysis of time reversal-MUSIC for soft fault location in power cables
11. Gao W, Wang X-H, Wang B-Z (2014) Time-reversal ESPRIT imaging method for the detection of single target. *J Electromagn Waves Appl* 28(5):634–640
12. Garrard WL, Jordan JM (1977) Design of nonlinear automatic flight control systems. *Automatica* 13(5):497–505
13. Bureau AAI (2019) Aircraft accident investigation preliminary report: Ethiopian airlines group b737–8 (MAX) registered ET-AVJ,

- 28 NM South East of Addis Ababa, Bole International Airport, March 10, 2019. Federal Democratic Republic of Ethiopia, Ministry of Transport, Aircraft Accident Investigation Bureau, Addis Ababa
14. KNKT (2018) Aircraft Accident Investigation Report—PT. Lion Mentari Airlines—Boeing 737-8 (MAX); PK-LQP—Tanjung Karawang, West Java—Republic of Indonesia—29 October 2018. Komite Nasional Keselamatan Transportasi Republic of Indonesia
 15. G. F. B. of Aircraft Accident Investigation (2015) Interim Report, BFU 6X014-14
 16. BEA (2008) Accident on 27 November 2008 off the Coast of Canet-Plage (66) to the Airbus A320-232 Registered D-AXLA Operated by XL Airways Germany. Office of Investigations and Analysis for the Safety of Civil Aviation
 17. Lombaerts T, Chu P, Mulder JAB, Stroosma O (2010) Fault tolerant flight control, a physical model approach. PhD thesis, Delft University of Technology (TU Delft)
 18. Goupil P (2011) AIRBUS state of the art and practices on FDI and FTC in flight control system. *Control Eng Pract* 19(6):524–539
 19. Yin S, Xiao B, Ding SX, Zhou D (2016) A review on recent development of spacecraft attitude fault tolerant control system. *IEEE Trans Ind Electron* 63(5):3311–3320
 20. Darvishi H, Ciunzo D, Eide ER, Rossi PS (2020) Sensor-fault detection, isolation and accommodation for digital twins via modular data-driven architecture. *IEEE Sens J*
 21. Mandal S, Santhi B, Sridhar S, Vinolia K, Swaminathan P (2017) Nuclear power plant thermocouple sensor-fault detection and classification using deep learning and generalized likelihood ratio test. *IEEE Trans Nucl Sci* 64(6):1526–1534
 22. Cho K, Van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y (2014) Learning phrase representations using RNN encoder-decoder for statistical machine translation. [arXiv:1406.1078](https://arxiv.org/abs/1406.1078)
 23. Li X, Wu X (2016) Long short-term memory based convolutional recurrent neural networks for large vocabulary speech recognition. [arXiv:1610.03165](https://arxiv.org/abs/1610.03165)
 24. Wang J-S, Chen Y-P (2006) A fully automated recurrent neural network for unknown dynamic system identification and control. *IEEE Trans Circuits Syst I Regul Pap* 53(6):1363–1372
 25. Ogunmolu O, Gu X, Jiang S, Gans N (2016) Nonlinear systems identification using deep dynamic neural networks. [arXiv:1610.01439](https://arxiv.org/abs/1610.01439)
 26. Sorensen KL, Blanke M, Johansen TA (2015) Diagnosis of wing icing through lift and drag coefficient change detection for small unmanned aircraft. *IFAC-PapersOnLine* 48(21):541–546
 27. Mathisen SH, Fossen TI, Johansen TA (2015) Non-linear model predictive control for guidance of a fixed-wing UAV in precision deep stall landing. In: 2015 international conference on unmanned aircraft systems (ICUAS). IEEE, pp 356–365
 28. Sherstinsky A (2020) Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Phys D* 404:132306
 29. Bengio Y, Simard P, Frasconi P (1994) Learning long-term dependencies with gradient descent is difficult. *IEEE Trans Neural Netw* 5(2):157–166
 30. Hochreiter S, Schmidhuber J (1997) Long short-term memory. *Neural Comput* 9(8):1735–1780
 31. Pascanu R, Mikolov T, Bengio Y (2013) On the difficulty of training recurrent neural networks. In: International conference on machine learning. PMLR, pp 1310–1318
 32. Graves A, Mohamed A-R, Hinton G (2013) Speech recognition with deep recurrent neural networks. In: 2013 IEEE international conference on acoustics, speech and signal processing. IEEE, pp 6645–6649
 33. LeCun YA, Bottou L, Orr GB, Müller K-R (2012) Efficient backprop. In: *Neural networks: tricks of the trade*. Springer, pp 9–48
 34. Ruder S (2016) An overview of gradient descent optimization algorithms. [arXiv:1609.04747](https://arxiv.org/abs/1609.04747)
 35. Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. [arXiv:1412.6980](https://arxiv.org/abs/1412.6980)
 36. Stevens BL, Lewis FL (2003) *Aircraft control and simulation*. Wiley, New York
 37. Nguyen LT (1979) Simulator study of stall/post-stall characteristics of a fighter airplane with relaxed longitudinal static stability, vol 12854. National Aeronautics and Space Administration (1979)
 38. Russell R (2003) Non-linear F-16 simulation using Simulink and MATLAB. University of Minnesota
 39. Simmons BM, McClelland HG, Woolsey CA (2019) Nonlinear model identification methodology for small, fixed-wing, unmanned aircraft. *J Aircr* 56(3):1056–1067
 40. Jategaonkar RV (2015) *Flight vehicle system identification: a time-domain methodology*, 2nd edn. American Institute of Aeronautics and Astronautics Inc, Reston
 41. Brunton SL, Kutz JN (2019) *Data-driven science and engineering: machine learning, dynamical systems, and control*. Cambridge University Press, Cambridge
 42. Brunton SL, Budisic M, Kaiser E, Kutz JN (2021) Modern Koopman theory for dynamical systems. [arXiv:2102.12086](https://arxiv.org/abs/2102.12086)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.