Machine Learning with Echo State Networks for Automated Fault Diagnosis in Small Unmanned Aircraft Systems

Diget, Emil Lykke; Hasan, Agus; Manoonpong, Poramate

Published in:
2022 International Conference on Unmanned Aircraft Systems (ICUAS)

DOI:
10.1109/ICUAS54217.2022.9836179

Publication date:
2022

Document version:
Accepted manuscript

Machine Learning with Echo State Networks for Automated Fault Diagnosis in Small Unmanned Aircraft Systems*

Emil Lykke Diget[1], Agus Hasan[2], and Poramate Manoonpong[1]

***Abstract*— Echo State Network (ESN) is one of machine learning methods that can be used to detect anomalies in sensor readings. The method predicts output signals, from which a prediction error can be created. To enable fault-tolerant control, ESN needs to be combined with a robust fault estimation method. Indeed, identifying the source of the faults, whether coming from sensors or actuators, is crucial in safety-critical Unmanned Aircraft Systems (UAS), since it will determine proper control actions when the faults occur. This paper presents a novel method to combine sensor anomaly detection using ESN with actuator fault estimation using adaptive extended Kalman filter (AEKF). Numerical results show the benefit of using the cascaded algorithm in a noisy environment. Furthermore, the presented method is validated using a hexacopter with actuator faults in indoor experiments.**

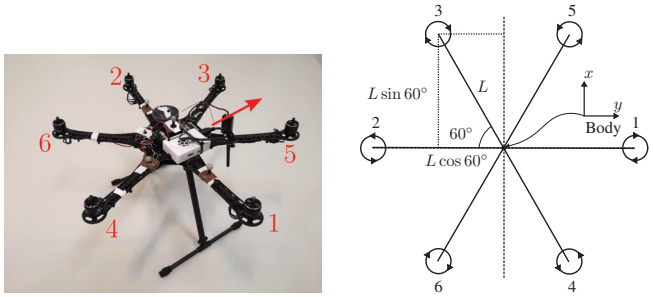
I. INTRODUCTION

Sensors and actuators in Unmanned Aircraft Systems (UAS) or drones are prone to failures. Each failure needs to be detected, localized, and estimated, as soon as possible to enable fault-tolerant control and other automated safety-critical decisions [1]. Several researchers have developed fault diagnosis algorithms for such problems. For example, in recent studies, Zhang [2] presented a diagnosis algorithm for actuator faults using adaptive Kalman filter (AKF) for linear systems. Skriver et al. [3] further improved the AKF method by extending it for nonlinear systems, called the adaptive extended Kalman filter (AEKF). Those approaches are different from traditional Kalman filter-based fault estimation, since the adaptive algorithm estimates the fault directly without considering the parameters as extended states. Hasan et al. [4] proposed an advanced version of the AEKF, called the adaptive eXogenous Kalman filter (AXKF). The method was used to detect a complete actuator failure on a UAS to trigger a parachute fail-safe system [5]. The advanced method was tested in numerical simulated data and data from actual test flights. The work focused on the virtual forces acting on the UAS, i.e. the thrust and torques acting in the x-, y-, and z-directions. While faults or errors can also occur on sensors, all these methods can estimate only the actuator fault level. In other words, they were designed for actuator fault diagnosis only [6].

*This work was supported by Equinor ASA through its gift professorship at NTNU.

[1]E.L. Diget and P. Manoonpong are with Maersk Mc-Kinney Moller Institute, University of Southern Denmark, Campusvej 55, Odense, 5320, Denmark. {eld,poma}@mmmi.sdu.dk

[2]A. Hasan is with Department of ICT and Natural Sciences, Norwegian University of Science and Technology, Larsgardsvegen 2, 6009 Ålesund, Norway. Corresponding email: agus.hasan@ntnu.no



(a) (b)

Fig. 1. The hexacopter used in this study. (a) The real hexacopter. The forward direction is marked by the red arrow and corresponds to the x-axis on (b). The motor numbers are annotated. (b) A free-body diagram of the hexacopter used for modelling. It shows the motor distribution and the coordinate system used in the modelling and the simulation.

For sensor fault diagnosis, Tu et al. [7] developed a method to calculate residual errors between the estimated state and the actual sensor readings from an inertial measurement unit (IMU) and to control a UAS during an attack. A related approach called *anomaly detection* has been introduced to flag sensor readings if they deviate from their normal expected state. Underlying this, Sharma et al. [8] investigated different types of sensor faults and how often they occur in the real world. They presented two different classes of fault detection methods: Rule-based and learning-based methods. The learning-based methods for identifying anomalies have also been studied in [9], [10], where they used Echo State Network (ESN) to predict the next measurement and calculates a prediction error. In [10], the authors found peaks in the prediction error and flagged these as anomalies, while in [9] the authors focused on state prediction rather than anomaly detection.

Based on previous studies, integrated sensor and actuator fault diagnosis has not been addressed. In this paper, we propose a novel approach for automated sensor and actuator fault diagnosis of small UAS. The approach combines a machine learning-based technique for sensor anomaly detection and AEKF for actuator fault estimation. In particular, we use ESN to flag anomalies on the sensory input data. This information is further used by an actuator fault estimator based on AEKF. By cascading these methods, we aim to provide a framework for safer drone operations where possibly dangerous faults in the sensors or actuators can be detected in time before the system fail.

II. METHODS

In this section, first we introduce the dynamic model of the UAS, followed with a brief introduction to the AEKF. The UAS frame used in this paper is a hexacopter. Furthermore, we present an ESN algorithm for sensor anomaly detection. Finally, we introduce our approach to combine the AEKF and ESN for automated sensor and actuator fault diagnosis.

A. Hexacopter Modelling

An illustration of the hexacopter used in this study can be seen in Figure 1. Let us denote $(x(t), y(t), z(t))$ and $(\phi(t), \theta(t), \psi(t))$ as the linear and angular position of the hexacopter in the inertial frame. The hexacopter is modelled as presented extensively in literature, e.g. in [11], [12]. The equations of motion for the hexacopter can be written as follow:

$$\begin{aligned}
\ddot{x}(t) &= -\frac{T(t)}{m}\left(\cos(\phi(t))\cos(\psi(t))\sin(\theta(t))\right. \\
&\quad \left.+\sin(\phi(t))\sin(\psi(t))\right), \quad (1)\\
\ddot{y}(t) &= -\frac{T(t)}{m}\left(\cos(\phi(t))\sin(\theta(t))\sin(\psi(t))\right. \\
&\quad \left.-\sin(\phi(t))\cos(\psi(t))\right), \quad (2)\\
\ddot{z}(t) &= -\frac{T(t)}{m}\left(\cos(\phi(t))\cos(\theta(t))\right)+g, \quad (3)\\
\ddot{\phi}(t) &= \frac{1}{J_{xx}}\left(\dot{\theta}(t)\dot{\psi}(t)\left(J_{yy}-J_{zz}\right)+M_x(t)\right), \quad (4)\\
\ddot{\theta}(t) &= \frac{1}{J_{yy}}\left(\dot{\phi}(t)\dot{\psi}(t)\left(J_{zz}-J_{xx}\right)+M_y(t)\right), \quad (5)\\
\ddot{\psi}(t) &= \frac{1}{J_{zz}}\left(\dot{\phi}(t)\dot{\theta}(t)\left(J_{xx}-J_{zz}\right)+M_z(t)\right), \quad (6)
\end{aligned}$$

where $T(t)$ is the total thrust, and the triplets $M_x(t)$, $M_y(t)$, and $M_z(t)$ are the control torques (moments) generated by differences in the rotor speeds. Here, J_{xx}, J_{yy}, J_{zz} are the moments of inertia around the $x(t)$, $y(t)$ and $z(t)$ axis, respectively. The total thrust and moments are defined as the control input, i.e. $\boldsymbol{U}(t) = \begin{bmatrix}T(t) & M_x(t) & M_y(t) & M_z(t)\end{bmatrix}^\intercal$, and can be written as an individual force applied for each rotor as follow:

$$\boldsymbol{U}(t) = \begin{bmatrix}1 & -L & 0 & -b\\ 1 & L & 0 & b\\ 1 & \frac{1}{2}L & \frac{1\sqrt{3}}{2}L & -b\\ 1 & -\frac{1}{2}L & -\frac{1\sqrt{3}}{2}L & b\\ 1 & -\frac{1}{2}L & \frac{1\sqrt{3}}{2}L & b\\ 1 & \frac{1}{2}L & -\frac{1\sqrt{3}}{2}L & -b\end{bmatrix}^\intercal \begin{bmatrix}F_1(t)\\ F_2(t)\\ F_3(t)\\ F_4(t)\\ F_5(t)\\ F_6(t)\end{bmatrix}, \quad (7)$$

where L is the distance between any rotor to the center of the drone, b is the drag factor, and F_i $(i = 1, \cdots, 6)$ is the individual thrust of each motor. Let us define the state of the drone as:

$$\boldsymbol{X} = \begin{bmatrix}\phi & \dot{\phi} & \theta & \dot{\theta} & \psi & \dot{\psi} & x & \dot{x} & y & \dot{y} & z & \dot{z}\end{bmatrix}^\intercal. \quad (8)$$

The nonlinear model (1)-(6) can be linearized and discretized at its priori estimate $\hat{\boldsymbol{X}}_k$, such that we obtain the following discrete-time system:

$$\boldsymbol{X}_{k+1} = \boldsymbol{F}_k(\hat{\boldsymbol{X}}_k)\boldsymbol{X}_k + \boldsymbol{B}_k\boldsymbol{U}_k + \boldsymbol{E}_k\boldsymbol{\vartheta}. \quad (9)$$

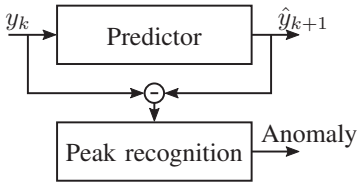


Fig. 2. The anomaly detection process where a predictor predicts the next value. The prediction error is calculated and analyzed using peak recognition which marks possible anomalies.

Here, we introduce $\boldsymbol{\vartheta}$ to model the faults in the motors or actuators. In particular, if $\boldsymbol{E}_k = -\boldsymbol{B}_k\text{diag}(\boldsymbol{U}_k)$, then the nominal control inputs become $\boldsymbol{B}_k\left(\boldsymbol{I} - \text{diag}(\boldsymbol{\vartheta})\right)\boldsymbol{U}_k$. From the last expression, if $\boldsymbol{\vartheta} = \boldsymbol{I}$, then the hexacopter experiences complete actuator failure, and in turn if $\boldsymbol{\vartheta} = \boldsymbol{0}$, where $\boldsymbol{0}$ is the zero matrix, the rotors are working as expected.

B. Actuator Fault Estimation

We use adaptive extended Kalman filter presented in [3] to estimate the magnitude of the actuator faults $\boldsymbol{\vartheta}$. The algorithm uses update law to estimate the fault through the following equation:

$$\boldsymbol{\vartheta}_{k+1} = \boldsymbol{\vartheta}_k + \Gamma_k\left(\boldsymbol{Y}_k - \boldsymbol{C}\boldsymbol{X}_k\right), \quad (10)$$

where $\boldsymbol{Y}_k$ is the measurement vector and the gain Γ_k is obtained from a recursive procedure involving some auxiliary variables and a forgetting factor. Given measurement matrix $\boldsymbol{C}$, the algorithm directly estimates the magnitude of the faults without considering it as an augmented state like in the traditional adaptive Kalman filter approach.

C. Sensor Anomaly Detection

To ensure that the actuator fault estimation can estimate the actuator faults correctly, it has to be ensured that the sensor signals can be trusted. If a sensor experiences abnormal measurements in the data stream, the specific measurement should be flagged as an *anomaly*. An anomaly is defined as a measurement that is significantly different from the estimated one [10]. Such an anomaly detector module analyzes the sensor data in an online fashion. The anomaly detection is inspired by [10] which uses ESN. The first step is to predict the next value at $k+1$ given a measurement at k. The second step is to flag possible anomalies in a process called *peak recognition*. An overview of the process can be seen in Figure 2.

1) Echo State Network: In general, ESN has a wide variety of uses. In [13], an ESN is used to control a fixed wing UAS and achieve better results than traditional controllers. In [14], the authors used ESN as a black-box system identifier of the position and attitude based on throttle-, roll-, pitch- and yaw commands. The benefits of ESN in insect-like robots for force estimation and terrain classification used for decision making on the choice of gait for the robot were shown in [15] and [16]. One of the advantages of ESN is that it is able to estimate non-linear relationships as demonstrated in [17], where a *Specific Growth Rate* for fish is calculated based on

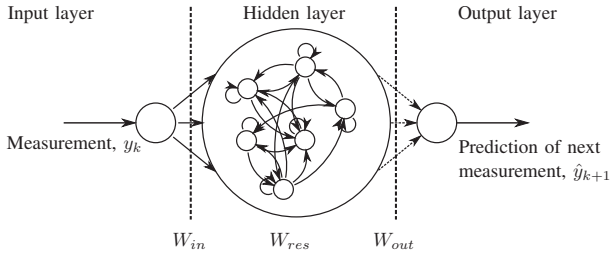


Fig. 3. An illustration of the ESN used in this paper. In this setup the current sensor measurement is used as the input and a prediction of the next measurement is the output. The solid lines are randomly initialized at the beginning, while the dotted lines are trained using the recursive least squares (RLS).

some parameters. In this study, we apply the ESN described in Figure 3 as a predictor for the sensory signals.

The ESN is chosen over other neural networks since the sensor signals are from a dynamic system where temporal relations between data points are important. Furthermore, it has also embedded temporal memory which leads to more robust signal prediction, in case of temporary missing feedback, compared to other techniques. As we will see in this section, the ESN is conceptually simple and also computationally inexpensive. In comparison, training for normal recurrent neural networks can be very difficult and computationally expensive.

Here, the ESN is trained to predict the next measurement. Therefore, the target signal, $\boldsymbol{d}_k$, is the same as the input signal, $\boldsymbol{u}_k$, shifted a step forward in time such that $\boldsymbol{d}_0 = \boldsymbol{u}_1$ and so forth. For this study, we have in total nine sensor signals (three angles, three angular velocities, and three positions) and use a modular approach where each ESN acts as a sensor prediction module for each sensor signal. Each ESN consists of input, hidden, and output layers with sizes N_u, N_x and N_y, respectively.

Three sets of weights are used in the network to define the connections: $\boldsymbol{W}_{in}$, $\boldsymbol{W}_{res}$, and $\boldsymbol{W}_{out}$. $\boldsymbol{W}_{in} \in \mathbb{R}^{N_x \times N_u}$ describes the weights of the connections between the input layer and the reservoir. $\boldsymbol{W}_{res} \in \mathbb{R}^{N_x \times N_x}$ describes the weights of the internal connections in the reservoir. $\boldsymbol{W}_{out} \in \mathbb{R}^{N_y \times N_x}$ describes the weights between the reservoir and the output layer. In this implementation, $N_u = 1$, $N_x = 40$, and $N_y = 1$. The matrices $\boldsymbol{W}_{in}$ and $\boldsymbol{W}_{res}$ are randomly initialized with values from a uniform distribution of $[-0.6; 0.6]$ and $[-1; 1]$, respectively. $\boldsymbol{W}_{in}$ and $\boldsymbol{W}_{res}$ are $50\,\%$ sparse, i.e. $50\,\%$ of the weights are zero. The state of the reservoir is updated using the following update rule:

$$\begin{aligned} \boldsymbol{x}_{k+1} &= (1-\lambda)\boldsymbol{x}_k \\ &\quad + \lambda f_{sys}(\boldsymbol{W}_{in}\boldsymbol{u}_{k+1} + \boldsymbol{W}_{res}\boldsymbol{x}_k + b_0), \\ \boldsymbol{y}_k &= \boldsymbol{W}_{out}\boldsymbol{x}_k, \end{aligned} \tag{11}$$

where $\boldsymbol{x}_k \in \mathbb{R}^{N_x \times 1}$ is the vector of reservoir activation, $\lambda = 0.7$ is the leaking rate, $\boldsymbol{u}_k$ is the current sensor reading, $b_0 = 0.001$ is a constant reservoir bias and $\boldsymbol{y}_k$ is the prediction of the next measurement. f_{sys} is a hyperbolic tangent activation function. To ensure echo state property, we keep the spectral radius of $\boldsymbol{W}_{res}$, $\rho(\boldsymbol{W}_{res}) < 1$ [18, p. 5]. The initial spectral radius is calculated by finding the largest eigenvalue, $\lambda_{\max}$, and normalizing $\boldsymbol{W}_{res}$ with it and then scale it with the desired spectral radius, which in this case is $\rho = 0.6$.

$$\boldsymbol{W}_{res} = \rho \cdot \frac{\boldsymbol{W}_{res,init}}{\lambda_{\max}}. \tag{12}$$

$\boldsymbol{W}_{out}$, the dashed connections in Figure 3, are trained in an online faction using the recursive least squares (RLS) algorithm presented in [19]. At each time step the input, $\boldsymbol{U}_k$ is fed to the network and the output weights are updated according to the RLS algorithm:

$$\boldsymbol{e}_k = \boldsymbol{d}_k - \boldsymbol{y}_k, \tag{13}$$

$$\boldsymbol{K}_k = \frac{\boldsymbol{p}_{k-1}\boldsymbol{x}_k}{\lambda_{RLS} + \boldsymbol{x}^\intercal \boldsymbol{p}_{k-1}\boldsymbol{x}_k}, \tag{14}$$

$$\boldsymbol{p}_k = \frac{1}{\lambda_{RLS}}\left(\boldsymbol{p}_{k-1} - \boldsymbol{K}_k \boldsymbol{x}^\intercal \boldsymbol{p}_{k-1}\right), \tag{15}$$

$$\boldsymbol{W}_{out,k} = \boldsymbol{W}_{out,k-1} + \boldsymbol{K}_k \boldsymbol{e}_k, \tag{16}$$

where $\boldsymbol{d}_k$ is the target signal, $\boldsymbol{y}_k$ is the output according to (11), $\boldsymbol{p}_k$ is the auto-correlation matrix, $\boldsymbol{K}_k$ is the RLS gain vector and λ_{RLS} is the learning rate. The output weight vector, $\boldsymbol{W}_{out}$, is initialized to the zero vector. Initially, the output weights cannot be trusted, so the auto-correlation is something very large defined as $\boldsymbol{p}_0 = \boldsymbol{I}/\beta$, where β is a small constant, in this case 10^{-4}. The learning rate, λ_{RLS}, is set to a constant value less than one, in this case $\lambda_{RLS} = 0.99$. The parameters of the ESN are summed up in Table I.

TABLE I
THE ECHO STATE NETWORK PARAMETERS.

Parameter	Symbol	Value
Number of input neurons	N_u	1
Number of reservoir neurons	N_x	40
Number of output neurons	N_y	1
Reservoir bias	b_0	0.001
Leaking rate	λ	0.7
Reservoir sparsity	–	$50\,\%$
Input to reservoir sparsity	–	$50\,\%$
Reservoir spectral radius	ρ	0.6
RLS learning constant	β	10^{-4}
RLS learning rate	λ_{RLS}	0.99

2) Low-Pass Filter: The low-pass filter (LPF) is considered as a simple way of estimating the next measurement, i.e. it can act as a simple alternative to the ESN based approach and is included for comparison. By using this method it is assumed that the next measurement is close to the ones we have already made. As for the ESN, each sensor signal will have it's own associated low-pass filter acting as a sensor prediction module. In this paper the LPF is an $N = 3$ sized moving average filter, such that the next measurement is predicted as:

$$\hat{\boldsymbol{y}}_{k+1} = \frac{1}{N}\sum_{i=0}^{N-1} \boldsymbol{y}_{k-i}. \tag{17}$$

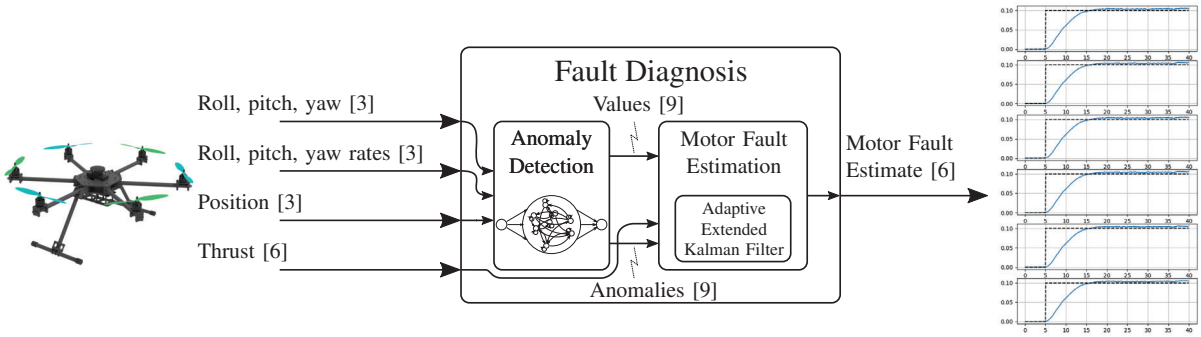


Fig. 4. The topology of the integration of the ESN based sensor anomaly detection and the AEKF-based actuator fault estimation. Each sensor data line has its own anomaly detector attached. If a measurement is anomalous, the corresponding measurement noise covariance is set very large. With a very large measurement covariance the AEKF will rely on the model during an anomaly rather than the measurement. [.] denotes signal dimension and source references.

3) Peak Recognition: The predictor predicts the next value. The previous prediction is compared to the current value; this gives the prediction error, δ_k. It should be close to zero for normal data and anomalous data should result in large differences [10]. The prediction error is then analysed by the peak recognition which outputs whether the measurement is an anomaly or not. The peak recognition used in this paper is based on the one proposed in [10]. The algorithm consists of three stages:

1) Moving average on prediction error:

$$\delta_{avg,k} = \frac{1}{2}(|\delta_{k-1}| + |\delta_k|). \tag{18}$$

2) Apply two different thresholds to the two most recent filtered data points. The second threshold is only applied, if the first is true. If both are true the data point is marked as an anomaly.

$$\delta_{avg,k} > \rho_k, \qquad \delta_{avg,k-1} > \rho_{k-1}, \tag{19}$$

where ρ_k and ρ_{k-1} are detection thresholds.

3) Assume that a single point will at most be marked as four data points. After four data points trailing marked anomalies are removed.

D. Anomalous Measurement

In this study a short anomaly is considered where there is a sudden change in the current to the next value. The anomaly is injected in two ways. The first method uses the the standard deviation, σ, of the whole signal and an amplification factor, β, to calculate an anomalous measurement, $\tilde{\boldsymbol{m}}_k$, using the original measurement, $\boldsymbol{m}_k$:

$$\tilde{\boldsymbol{m}}_k = \boldsymbol{m}_k + \beta\sigma. \tag{20}$$

The second method is to simply multiply the measurement with a constant, α:

$$\tilde{\boldsymbol{m}}_k = \alpha\boldsymbol{m}_k. \tag{21}$$

E. Combination of Sensor Anomaly Detection and Actuator Fault Estimation

The sensor anomaly detection and actuator fault estimation are integrated with each other to spare the AEKF from faulty sensor values. This is done by assigning one anomaly detector module to each of the sensor lines, totalling in nine. Each anomaly detection module outputs a Boolean value whether the measurement is an anomaly or not. The sensor measurements are fed to the AEKF as the measurement vector, $\boldsymbol{y}$, and the list of anomalies is used to make the measurement covariance matrix, $\boldsymbol{R}$. If a measurement is an anomaly, the anomaly flag is true and the associated standard deviation should be very large. If a measurement is normal, the anomaly flag is false and the associated standard deviation should correspond to the sensor standard deviation. An overview of this topology can be seen in Figure 4.

III. EXPERIMENT RESULTS AND DISCUSSIONS

In this section the test results are described and explained. It is divided into numerical simulation section and real experiment sections. In the numerical section, the ESN-based and LPF-based sensor anomaly detectors are compared. The benefit of using sensor anomaly detection in a noisy environment in use with the AEKF is also explored. In the experiment section, the anomaly detection and AEKF is tested against real world data from indoor test flights in three different scenarios.

A. Numerical Simulation

The following tests are simulated numerically. First the two predictors for the anomaly detection are compared. Secondly the effect of shot noise in the sensor data is examined in relation to actuator fault estimation. Here, the anomaly detection is used to flag and remove troublesome sensor readings.

1) Sensor Anomaly Detection: Echo State Network vs. Low-Pass Filter: In this test the ESN-based and the LPF-based anomaly detection are compared. The test is run on

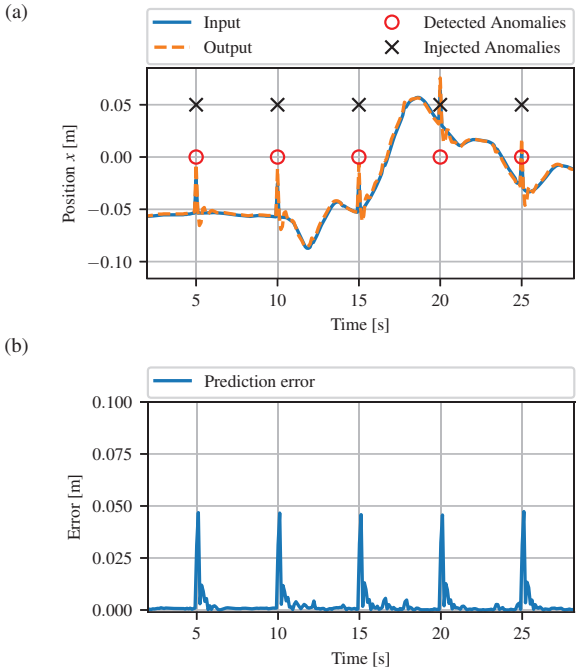


Fig. 5. Anomaly detection using an echo state network as the signal predictor. (a) Input signal, output signal, injected anomalies and detected anomalies. (b) The prediction error. The previous prediction compared to the current value. Should be close to zero when data is normal. Notice how the prediction error is mostly zero during normal operation and high during anomalies.

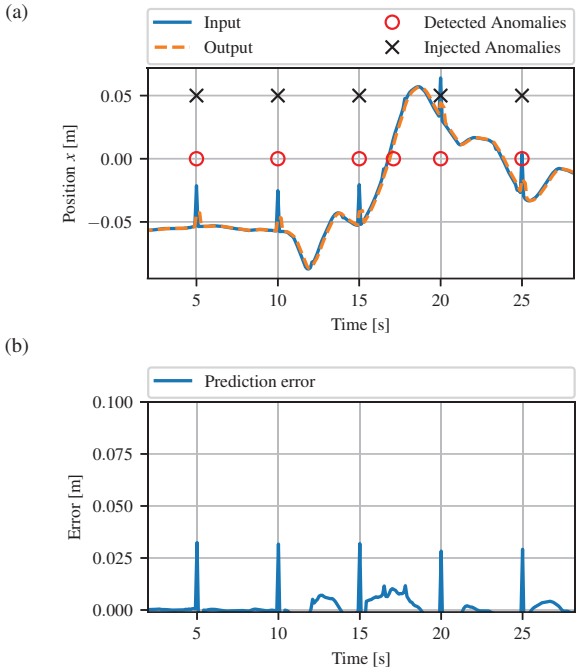


Fig. 6. Anomaly detection using a low pass filter as the signal predictor. (a) Input signal, output signal, injected anomalies and detected anomalies. (b) The prediction error. The previous prediction compared to the current value. Should be close to zero when data is normal. Notice how the prediction error is large during the transitions at 12.5 s and 15–18 s.

a data-set consisting of position in the x-direction. Noise is added to the sensor readings using (20), with $\beta = 1$ every 5 s, which results in spikes. In the timespan from 5 s to 25 s five anomalies are injected. The anomaly detection thresholds are set as follows: $\rho_k = 0.01$ and $\rho_{k-1} = 0.0001$. In Figure 5 the results of the ESN-based anomaly detection can be seen, and the results of the LPF-based can be seen in Figure 6. One of the differences between the methods are that the ESN amplifies the signal when there is an error, while the LPF smooths out the signal. For both solutions clear peaks in the prediction error can be seen, and these are also successfully detected. However, looking at the prediction error it is clear that the LPF has problems with signal changes such as the movements around 12 s and 17 s which leads to a high error. At 17 s this is enough to incorrectly trigger an anomaly detection. The ESN, in comparison, can correctly trigger an anomaly detection where it has a lower error in the normal operation and also a higher error at the anomalies.

2) ESN+AEKF vs. AEKF: Numerical simulations are performed using a hexacopter model (1)-(7) assumed to be equipped with a GPS and IMU to measure the position, the attitude and the angular velocities. The simulation is run for 40 s. The drone is initialized in the position $(0, 0, 10)$ and held in position by keeping the motor thrusts at the ideal levels to maintain a hover. A 10 % fault is introduced on each motor at 5 s. Noise is added to the sensor readings, by multiplying the measurement vector with 1.5 every five seconds, (21), which results in spikes. The parameters used in the simulation are given in Table II.

TABLE II

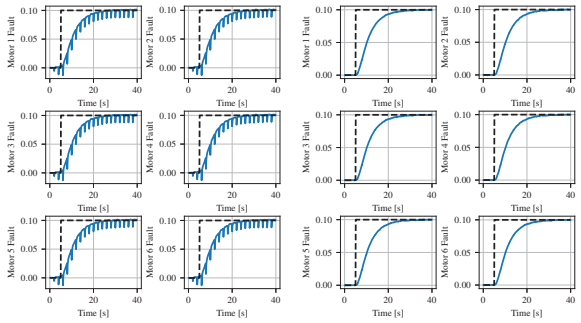
PARAMETERS USED BY THE MOTOR FAULT ESTIMATION IN BOTH THE NUMERICAL AND THE FIELD EXPERIMENTS.

Parameter	Symbol	Value
Mass	m	1.5 kg
Arm length	L	0.268 m
Moment of inertia, x	J_{xx}	0.027 465 Kgm^2
Moment of inertia, y	J_{yy}	0.027 465 Kgm^2
Moment of inertia, z	J_{zz}	0.053 868 Kgm^2
Drag coefficient	b	0.01
Gravity acceleration	g	9.81 $\mathrm{m/s}^2$

The resulting actuator fault estimates can be seen in Figure 7. Without the anomaly detection, the noise on the input data is also visible on the actuator fault estimate. Using the anomaly detection, the fault on the input is removed and the actuator fault estimate is completely smooth.

B. Field Experiment

In this series of three experiments, the presented algorithm is tested on a real system. For the experiments a S550 hexarotor with the PixHawk 2.1 Cube FC running the PX4 flight-stack is used. It can be seen in Figure 1a. The drone weighs around 1.5 kg and has a span from motor center to motor center across the middle of ~ 53.5 cm. The drone is equipped with six T-Motor Air Gear 350 Motor+Propeller



(a) (b)

Fig. 7. Numerical simulation motor fault estimate result. (a) Without anomaly detection. (b) With anomaly detection. Notice how the anomaly detection can remove all the resulting spikes in the motor fault estimate.

combo. The propellers are $9.5'' \times 4.5''$. Firstly, the drone will be flown with all six propellers intact. Secondly, the drone will be flown with five intact propeller and one altered one. In the end a weight will be suspended $\sim 0.5\,\mathrm{m}$ underneath the drone such that it should experience a sudden actuator fault due to the extra weight. For all three tests the drone is commanded to hover at the position $(0, 0, 1)$m. The position and orientation is tracked using the indoor OptiTrack vision based motion capture system. The parameters given in Table II. The ESNs are trained on real sensor data with the parameters given in Table I. The anomaly detection parameters, ρ_k and ρ_{k-1}, are tuned for each sensor value to minimize the number of false negatives and false positives.

1) Six Normal Propellers: When flying with all six propellers intact the motor fault estimation should ideally be zero. The drone starts the flight at about $10\,\mathrm{s}$. Looking at Figure 8 it can be seen that the fault estimate for all propellers indeed remain close to zero. They are not exactly zero which is probably caused by differences between the model and the real world. Some of the motors also have a fault less than zero which would mean that the motor works better than it should, which again can be explained by the same reason as before.

2) One Broken Propeller: In this experiment the propeller on motor 2 is replaced by a broken one. The resulting motor fault estimate of this test is presented in Figure 9. The broken propeller can be seen in the top right of the figure. The fault of motor 2 quickly rises as it should do. Ideally the algorithm should only estimate a fault on motor 2, though it also estimates a fault on the opposite motor, motor 1. Fault estimates other than for motor 2 should be zero. This is due to the hexacopter being an over-determined system, i.e. the motor faults are estimated from the fault on the virtual forces (thrust and the three torques around x, y and z) which have multiple solutions of which the algorithm will find the minimum norm solution.

3) Taking off with Extra Weight: In this experiment a weight of $\sim 300\,\mathrm{g}$ is attached to a rope of length $\sim 0.5\,\mathrm{m}$. The weight increases the total weight of the drone by $\sim \frac{0.3\,\mathrm{kg}}{1.5\,\mathrm{kg}} = 20\,\%$. The weight increase is not known by the system which in turn should interpret it as a $20\,\%$ fault on all the motors. The result of the motor fault estimation can be seen on Figure 10. A line at $20\,\%$ is added to this plot. It can be seen that all the motor faults are estimated to $\sim 20\,\%$ confirming the estimation works effectively. In this test there is also some fluctuation from the desired value which also is explainable by the difference between the real world and the model.

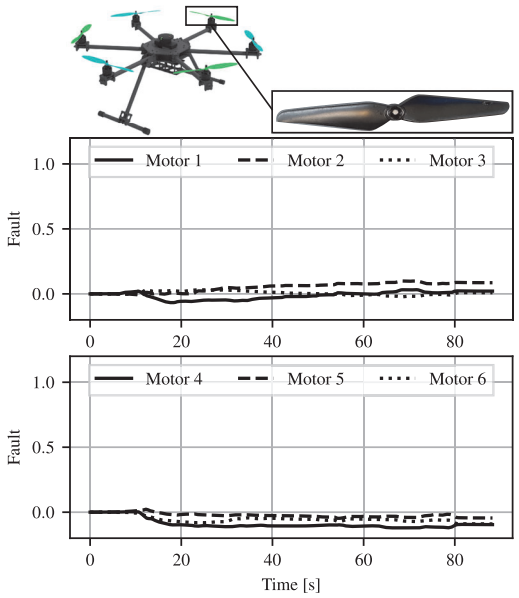


Fig. 8. Estimation of the fault on the six motors with no broken propellers. The drone starts the flight at around $10\,\mathrm{s}$. The fault for all motors moves around zero. A fault less than zero would mean that the motor works better than expected. This is probably due to the difference between the real world and the model. A video showing this experiment can be seen at: `www.manoonpong.com/FaultDiagnosis/video1.mp4`

IV. CONCLUSIONS

This paper presents a method of combining sensor anomaly detection with actuator fault estimation. The method changes the measurement uncertainty from low to high when encountering an anomaly. Numerical simulations and real world experiments show that the method works effectively. In the future, we will use the actuator fault estimate for fault-tolerant control. It could also be interesting to explore other types of neural networks, such as feed forward networks, for anomaly detection. The combined system should also be deployed to a mobile computer on a drone to be used in real life.

ACKNOWLEDGMENT

We would like to thank Jes Hundevadt Jepsen for his help with the experimental work with the drone and the OptiTrack system. This research is partially funded by Equinor's gift professorship at NTNU.

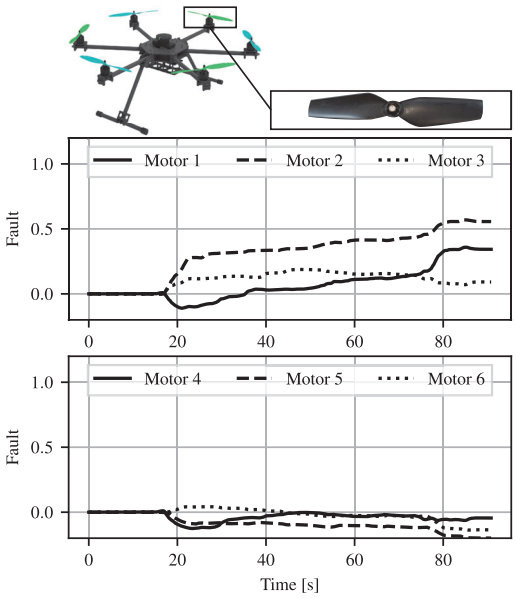


Fig. 9. Estimation of the fault on the six motors with one broken propellers. The drone starts the flight around 18 s. Motor 2 quickly reaches a considerable fault level while it takes a while before motor 1 does it. The fault should only be present for motor 2, and not motor 1. These differences are probably a product of the system being over-determined, i.e. the motor faults, of which there are six (one for each motor), are estimated from the fault on the virtual forces, of which there are four (thrust and torque in x, y, z). A video showing this experiment can be seen at: `www.manoonpong.com/FaultDiagnosis/video1.mp4`

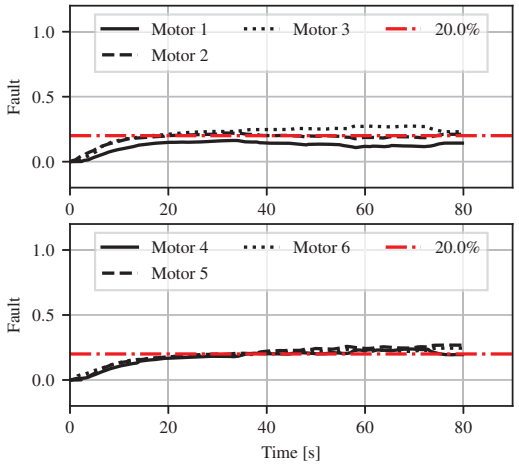


Fig. 10. Estimation of the fault on the six motors while flying with a weight of $\sim$ 300 g. The flight starts at the beginning of the data. Fault levels of all the motors rise to around the expected level of 20 %. A video showing this experiment can be seen at: `www.manoonpong.com/FaultDiagnosis/video1.mp4`

REFERENCES

[1] E. L. Diget, A. Hasan, and P. Manoonpong, "Fault-Tolerant Model Predictive Control for Multirotor UAVs," *American Control Conference*, 2022.

[2] Q. Zhang, "Adaptive Kalman Filter for Actuator Fault Diagnosis," *Automatica, Elsevier, 2018, 93, pp.333-342.*, 2018.

[3] M. Skriver, J. Helck, and A. Hasan, "Adaptive Extended Kalman Filter for Actuator Fault Diagnosis," *4th International Conference on System Reliability and Safety (ICSRS)*, 2019.

[4] A. Hasan, "Adaptive eXogenous Kalman Filter for Actuator Fault Diagnosis in Robotics and Autonomous Systems," *7th International Conference on Control, Mechatronics and Automation (ICCMA)*, pp. 162–167, 2019.

[5] A. Hasan, V. Tofterup, and K. Jensen, "Model-Based Fail-Safe Module for Autonomous Multirotor UAVs with Parachute Systems," *International Conference on Unmanned Aircraft Systems (ICUAS)*, 2019.

[6] M. Tahavori and A. Hasan, "Fault Recoverability for Nonlinear Systems with Application to Fault Tolerant Control of UAVs," *Aerospace Science and Technology*, vol. 107, p. 106282, 2021.

[7] Z. Tu, F. Fei, M. Eagon, D. Xu, and X. Deng, "Flight Recovery of MAVs with Compromised IMU," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019.

[8] A. Sharma, L. Golubchik, and R. Govindan, "On the Prevalence of Sensor Faults in Real-World," 06 2007, pp. 213–222.

[9] O. Obst, X. R. Wang, and M. Prokopenko, "Using Echo State Networks for Anomaly Detection in Underground Coal Mines," *International Conference on Information Processing in Sensor Networks*, 2008.

[10] M. Chang, A. Terzis, and P. Bonnet, "Mote-based Online Anomaly Detection using Echo State Networks," *Distributed Computing in Sensor Systems*, 2009.

[11] A. Ahmed, C. M. Elias, and O. M. Shehata, "Control of a Hexa-Rotor System Using Various X-MPC Techniques," *The 8th International Conference on Control, Mechatronics and Automation*, 2020.

[12] A. G. Rot, A. Hasan, and P. Manoonpong, "Robust Actuator Fault Diagnosis Algorithm for Autonomous Hexacopter UAVs," *IFAC World Congress*, 2020.

[13] B. Pugach, B. Beallo, D. Bement, S. McGough, N. Miller, J. Morgan, L. Rodriguez, K. Winterer, T. Sherman, S. Bhandari, and Z. Aliyazicioglu, "Nonlinear Controller for a UAV using Echo State Network," *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2017.

[14] A. Vargas, M. Ireland, and D. Anderson, "System Identification of Multi-Rotor UAV's using Echo State Networks," *AUVSI's Unmanned Systems 2015*, 2015.

[15] S. Dasgupta, D. Goldschmidt, F. Wörgötter, and P. Manoonpong, "Distributed recurrent neural forward models with synaptic adaptation and CPG-based control for complex behaviors of walking robots," *Frontiers in Neurorobotics*, vol. 9, p. 10, 2015.

[16] P. Borijindakul, N. Jinuntuya, and P. Manoonpong, "Haptic Feedback with a Reservoir Computing-Based Recurrent Neural Network for Multiple Terrain Classification of a Walking Robot," *Intelligent Robotics and Applications.*, 2019.

[17] K. Rungruangsak-Torrissen and P. Manoonpong, "Neural computational model GrowthEstimate: A model for studying living resources through digestive efficiency," *PLoS ONE 14*, 2018.

[18] M. Lukoševičius, "A Practical Guide to Applying Echo State Networks," *Neural Networks: Tricks of the Trade, Reloaded*, 2012.

[19] H. Jaeger, "Adaptive Nonlinear System Identification with Echo State Networks," *NIPS*, 06 2003.