Intelligent Fault Diagnosis Method for UAVs based on FV-SVM

Dheer Samtani Shreyas Pakhare Utkarsh Raj Singh and Jatin Khandal

Abstract—The advancements of small UAVs necessitate intelligent approaches towards the issue of fault diagnosis. The hardware limitations for these small vehicles call for the usage of analytical redundancy to be implemented in contrast to the conventional hardware redundancy approach. An approach of implementing machine learning practices to diagnose faults on a quadcopter has been utilized which overcomes the burden needed in model-based fault analysis. In this study, first, a model of a quadcopter is simulated to generate data and test the designed algorithms. A supervised classification method, SVM (Support Vector Machines) is used to classify the faulty and nominal flight conditions as well as to classify the type of failure. The features selected are the gyro and accelerometer measurements and the inputs to the four motors of the quadcopter. The faults considered are various actuator failures namely - locked-in place, loss of effectiveness, total system failure and hard-over fault. The training is held offline due to the need for labelled data. The results show that for simulated measurements, SVM gives an accuracy of 96% for binary classification and 95% for multi-class classification.

I. INTRODUCTION

With a growing indigenous base of drone manufacturers and exporters, and incentives from the government, industry sentiment is positive. With this affirmative sentiment, the need to ensure a safer flight has become more prominent in recent times. A solution to this is fault diagnosis and detection. An approach to implementing machine learning practices to diagnose faults on a small quadcopter is selected. The reason behind selecting the machine learning approach is to avoid the burden of accurately modelling the entire process is quite complicated. A drawback of the model-based approaches is that they require an accurate model of the aircraft for successful detection. In a small UAV system, which is susceptible to various uncertainties/disturbances and usually lacks an accurate model, using a model-based approach might fail.

In data-driven methods, detailed knowledge about the internal dynamics of the system is not necessary. The data available is the source of information about the behaviour of the system. Supervised learning, which requires labelling the fault cases previously in the training data, is usually used for data-centric fault detection [13]. In case of an unlabeled fault, the result is expected as a probability distribution of the available normal modes, identified fault labels and a probable unknown fault.

First, a model of an aircraft is simulated, to generate the data regarding the flight parameters which are then labelled with the type of fault or not faulty based on gyrometer, accelerometer measurements and the input to the four motors of the quadcopter.

Understandably, Re-configurable control systems rely heavily on the successful detection of the faults or estimation of the dynamics of the vehicle. Failure of such will result in degradation of the performance (sometimes even catastrophic) rather than improvement. Therefore, a successful detection and diagnosis method is always desired.

II. RELATED WORK

Detection strategy can be mainly based on two methods: 1) Model-based, and 2) Data-driven.

A drawback of the model-based approaches is that they require an accurate model of the aircraft for successful detection.

In data-driven methods, detailed knowledge of the internal dynamics of the system is not necessary. Supervised learning, which requires labelling the fault cases previously in the training data, is usually used for data-centric fault detection [7]. In case of an unlabeled fault, the result is expected as a probability distribution of the available normal modes, identified fault labels and a probable unknown fault.

In [8], the author applies an SVM classifier to detect and diagnose faults in drones with an actuator failure for fixed-wing drones and uses the accelerometer, and gyrometer readings along with commanded deflection input for the wings whereas this study is for a quadcopter wherein instead of commanded deflection, input to the four motors are used as features. [9] continuously regresses an estimation of the aircraft actuator's health. They have used a deep learning structure that comprises one-dimensional CNN followed by LSTM layers. The health information is embedded in the NDI-based control system. [10] also proposed a deep learning-based fault detection and identification method using CNN and LSTM. Sindhwani et al. [11], while using a highly redundant VTOL vehicle, learned the dynamics estimate function from 5000 mission flights and used this function as a detector for anomalies on a new set of 5000 mission flights. They have captured three different anomaly types. [12] compared model-based and data-driven fault detection methods by using real-flight data gathered from flight tests. An important initiative has been made by [13], where their ALFA Flight data set has been open-sourced. Furthermore, in [14], they present the good use of their data set for online anomaly detection. We will be continuing this initiation by open-sourcing our flight data set and all related programs.

III. OVERVIEW AND CONTRIBUTIONS

We plan to open-source the dataset generated for the parrot mini-drone quadcopter on GitHub.

IV. FLIGHT TEST PLATFORM

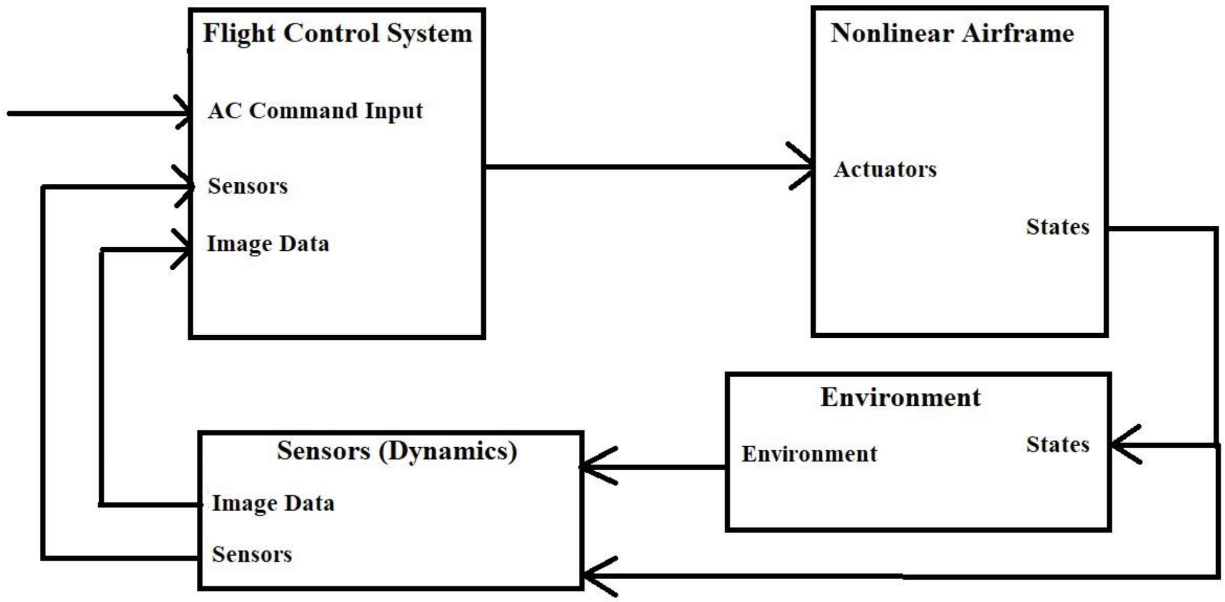
A] Software

We have used the Parrot Mini drone Quadcopter for the simulation of the flight inside Simulink. The model for the line following drone which we have used was taken from the open-sourced GitHub repository of [1].

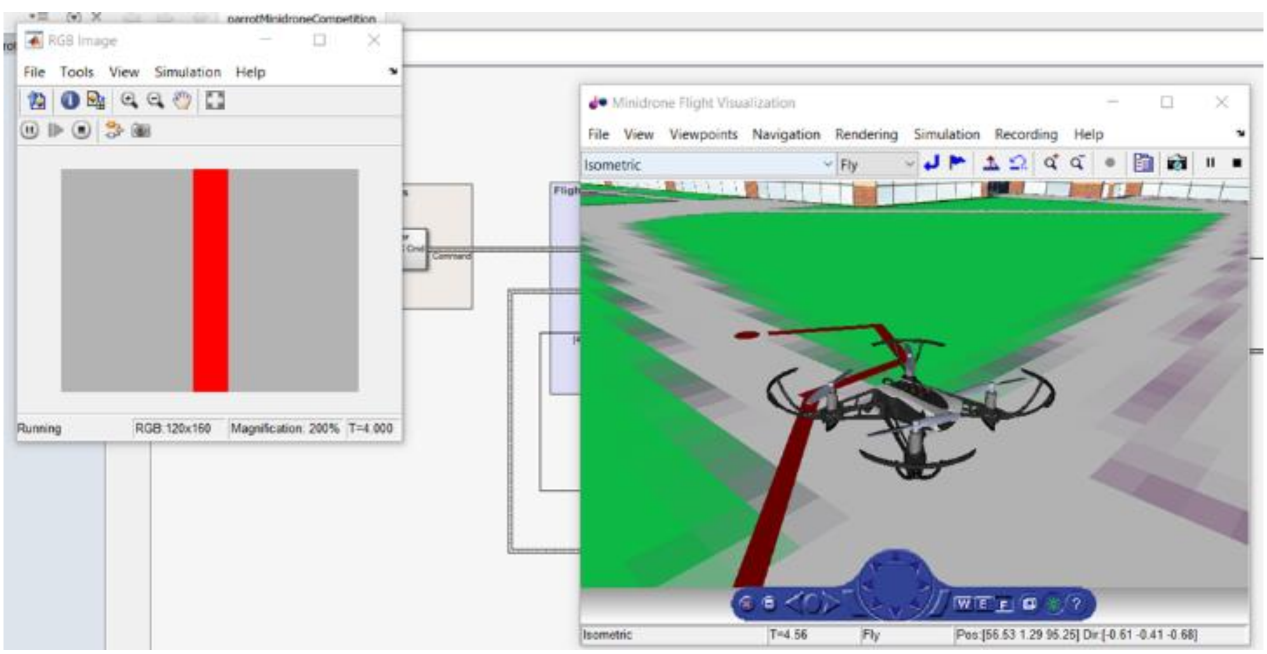
A view of the control system model in Simulink is shown below which has various components such as the flight command system, flight control system, multi-copter model, environment model, sensors model and flight visualization block.

Modifications inside the flight control system were made for generating the data set with the aforementioned faults.

Block Model of Quadcopter



A pictorial representation of the flight simulation



V] FAULT DETECTION METHODOLOGY

A. *SVM*

Support Vector Machine is one of the most popular off-the-shelf frameworks for supervised learning. The most forthcoming properties of SVMs are:

1) SVMs construct a decision boundary that has the largest distance to the example points, called **the Maximum margin separator**. This helps with generalization.

2) SVMs can use **kernel-trick**, which is to represent the linear separator in a high-dimensional space so that it can represent a non-linear separator in the original space.

After an SVM classifier is trained, the performance of the classification is evaluated with a variety of different metrics, depending on the nature of the classification problem.

A brief reminder about the performance metrics that are used in this work:

- **Accuracy** is the ratio between correctly classified points to a total number of points.

- **Precision** is the ratio between correctly predicted faults to the total number of detection (both true and false). It indicates how reliable is the model when it detects a fault.

- **Recall** is the ratio between correctly predicted faults to the total number of existing fault points. It

indicates how reliable the model is in detecting faults.

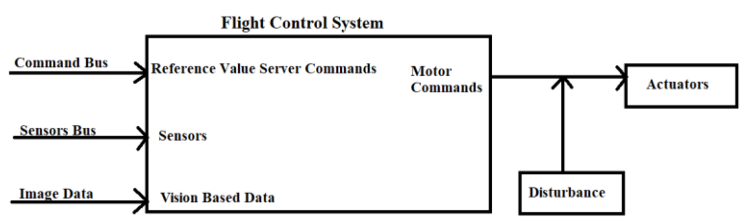
- **F1-score** is the harmonic mean of the precision and recall.

- **Confusion matrix** is a performance measurement for machine learning classification problems where the output can be two or more classes. It is a table with 4 different combinations (in the case of n classes, it is n^2 combinations) of predicted and actual values.

B. Modeling of Actuator Failure

Common actuator failure cases for an aircraft include: 1) locked-in place, 2) floating around trim, 3) hard-over, and 4) loss of effectiveness. We have also implemented the total system failure fault. We have utilized the linear relation between torques for roll, pitch, yaw and thrust and the inputs applied to the different motors for injection of various kinds of faults by changing the commanded actuator inputs generated from the control system by introducing a **multiplication factor D** and then passing the resulting output to the multi-copter.

Introducing Fault



1.Locked-in-place fault D=0

Locked-in place fault can be modelled by making D = 0 and changing the output to the input of the system at a previous instant.

2. Loss of Effectiveness Fault D<1

Actuator loss of effectiveness can be modelled by changing D and reducing it to less than one wherein the actual commanded input cannot be obtained at the output.

3. Hard-Over Fault D>1

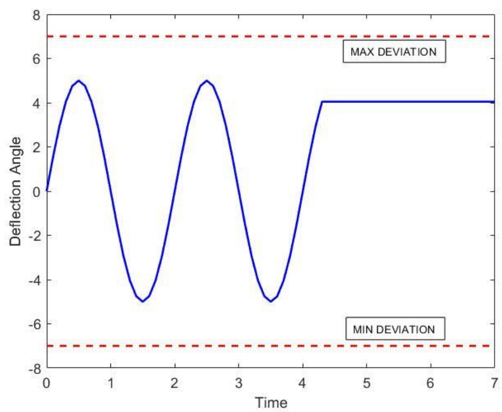
In this fault, the actuators run almost at their maximum limit and keep moving on. This has been implemented by multiplying the input by a high factor such as 3.

4. Total System Failure Fault D = 0.

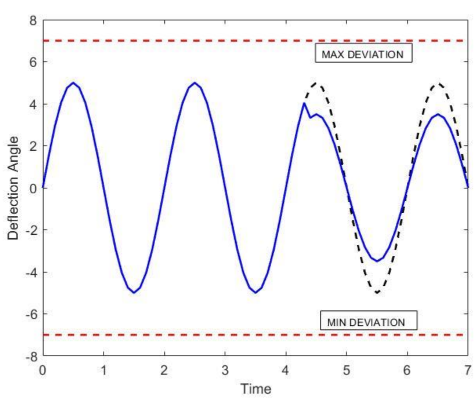
In case the system shuts down totally. This has been implemented by reducing the input to zero.

The figure given below explains the faults being implemented

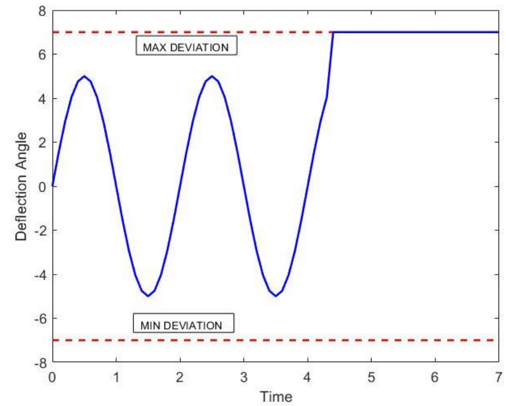
Locked-In Place Fault



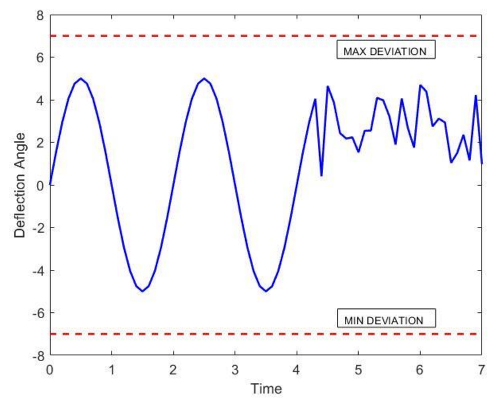
Loss of Effectiveness Fault



Hard-Over Fault



Total System Failure Fault



MATLAB Code for implementing various faults

```
function y = fcn(u,t)
lockedin = u;

% t = get_param('flightControlSystem','SimulationTime');

if t==6.5
    lockedin = u;
end
% Locked IN Place Fault - New input is not realised by the motor.
if t>6.5 && t<=8.5
    output = lockedin;
%     Loss of effectiveness Fault
elseif t>12.5 && t<=14.5
    output = 0.7*u;
%     Total System Failure Fault
elseif t>20.2 && t<=20.4
    output = 0*u;
%     Hard-Over Fault. Can Maybe increase gain further more.
elseif t>25.5 && t<27.5
    output = 3*u;
else
    output =u;
end

y = output ;
```

Generation of Feature Trajectories

It is usually desirable to reduce the number of input features to both reduce the computational cost of modelling and, in some cases, to improve the performance of the model by removing unrelated information. We decided that a basic feature list will suffice to show the feasibility of the proposed idea. Therefore, without complex Feature Engineering, we directly used the linear accelerations (axyz), angular rates (ωxyz) and the autopilot commanded inputs for the four motor actuators, adding up to 10 features.

SVMs assume that all features are centered around 0 and have variance in the same order. To prevent the domination of any single attribute in the feature vector, we standardize features by removing the mean and scaling to unit variance (using the default normalize function in MATLAB). Centering and scaling happen independently on each attribute of the feature vector by computing the relevant statistics on the samples in the training set. Mean and standard deviation are then stored to be used later during the inference pre-processing phase.

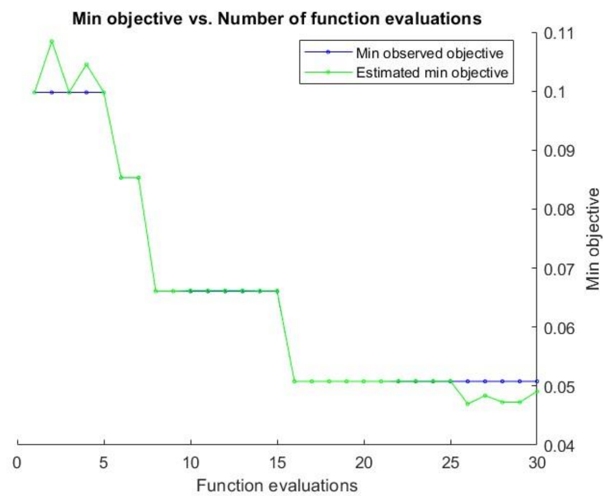
VI] RESULTS

An accuracy of 94.51 % was achieved with the multi-class SVM. The hyperparameter tuning was done with Bayesian Optimization.

Best Results were obtained for one vs one coding with hyperparameter values as C = 986.22 (C denotes Box Constraint which denotes the cost of misclassified points.) and γ = 224.33 (Kernel Scale which is a scaling parameter for input data) with an estimated objective function value of 0.049085.

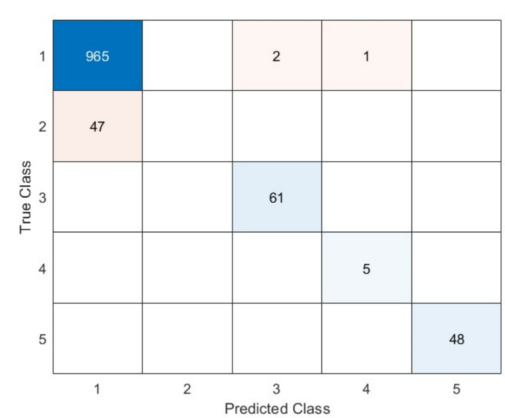
In Bayesian Optimization, we build a probability model of the objective function to be minimized and use it to select the most promising hyperparameters to evaluate in our true objective function.

A surrogate probability model of the objective function is built and we find hyperparameters that perform best on the surrogate function. We then apply them to our true objective function and update our surrogate model incorporating new results.



The above plot shows that as the number of iterations increased the cost of the objective function decreased and the value predicted by our surrogate function starts matching with that of our objective function.

Confusion matrix for multiclass classifier



The horizontal axis of the confusion matrix represents the class (1 to 5 as we go from left to right) as predicted by the SVM model, and the vertical axis represents the actual class (1 to 5 as we go from top to bottom) for the input. Hence, we desire all the off-diagonal elements to be 0 for all the samples in the test dataset to be correctly classified by the model.

The following is the representation assigned to the classes,

1 - No fault, 2- locked in fault, 3- loss of effectiveness fault, 4- total system failure fault, 5- hard over fault.

The locked-in-place fault was simulated during a phase of flight where the quadcopter was moving in a straight line as a result high error exists for it.

VII] ACKNOWLEDGEMENTS

This work was supported by Dr M. Felix Orlando and Mr. Kaushik Halder. We are grateful to them for their guidance.

References

[1] M. Terlizzi, "Vision-Based Pure Pursuing Algorithm," GitHub repository. [Online]. Available: https://github.com/mar4945/Vision-Based-Pure-Pursuing-Algorithm.

[2] Model-based Design of a Line-tracking Algorithm for a Low-cost Mini Drone through Vision-based Control BY PAOLO CEPPI

[3] Real-time Fault Detection on Small Fixed-Wing UAVs using Machine Learning – Murat Bronz; Elgiz Baskaya; Daniel Delahaye; Stephane Puechmore – 2020 AIAA/IEEE 39th Digital Avionics Systems Conference (DASC) IEEE Xplore.

[4] Fault Detection & Diagnosis for small UAVs via Machine Learning – Elgiz Baskaya; Murat Bronz; Daniel Delahaye – 2017 IEEE/AIAA 36th Digital Avionics Systems Conference (DASC) IEEE Xplore.

[5] A Vision-Based Algorithm for a Path Following Problem – Mario Terlizzi Giuseppe Silano Luigi Russo Muhammad Aatif; Amin Basiri; Valerio Mariani; Luigi Iannelli; Luigi Glielmo 2021 International Conference on Unmanned Aircraft Systems (ICUAS)

[6] ATL Drone Module – Get, Set, Fly! – Dr Ayesha Chaudhary, Atal Innovation Mission, NITI Aayog.

[7] J. Stutz, "On data-centric diagnosis of aircraft systems," IEEE Transactions on Systems, Man and Cybernetics, 2010.

[8] E. Baskaya, "Fault detection and diagnosis for drones using machine learning," PhD dissertation, Universite Paul Sabatier-Toulouse III, ´ 2019.

[9] B. Eroglu, M. C. Sahin, and N. K. Ure, "Autolanding control system design with deep learning-based fault estimation," Aerospace Science and Technology, p. 105855, 2020.

[10] V. Sadhu, S. Zonouz, and D. Pompili, "On-board deep-learning-based unmanned aerial vehicle fault cause detection and identification," arXiv preprint arXiv:2005.00336, 2020.

[11] V. Sindhwani, H. Sidahmed, K. Choromanski, and B. Jones, "Unsupervised anomaly detection for self-flying delivery drones," 2020.

[12] P. Freeman, R. Pandita, N. Srivastava, and G. J. Balas, "Model-based and data-driven fault detection performance for a small UAV," IEEE/ASME Transactions on Mechatronics, vol. 18, no. 4, pp. 1300–1309, 2013.

[13] A. Keipour, M. Mousavi, and S. Scherer, "Alfa: A dataset for UAV fault and anomaly detection," arXiv preprint arXiv:1907.06268, 2019.

[14] ——, "Automatic real-time anomaly detection for autonomous aerial vehicles," in 2019 International Conference on Robotics and Automation (ICRA). IEEE, 2019, pp. 5679–5685.