Learning-based Minimally-Sensed Fault-Tolerant Adaptive Flight Control

Michael O'Connell*, Joshua Cho*, Matthew Anderson, and Soon-Jo Chung

***Abstract*— This paper presents a novel sparse failure identification method along with rapid control reconstitution using deep neural networks for detecting and compensating for motor failures in multirotor aircraft. The presented method leverages a reformulation of the Neural-Fly online adaptation algorithm and a unique control allocation update approach to prevent motor saturation and improve tracking performance in the presence of modeling errors and actuator faults. Experimental flight results demonstrate the ability of the method to maintain control of an aircraft by isolating motor failures and reallocating control in under one second, whilst also reducing the trajectory tracking error by 48 % compared to the baseline. When direct motor speed sensing is available, the proposed allocation algorithm and control architecture enables almost instantaneous failure compensation. The findings of this study contribute to the development of robust fault detection and compensation strategies for over-actuated aircraft, enhancing aircraft safety and reliability in a wide range of applications.**

***Index Terms*— Fault tolerant control, adaptive control, remotely piloted aircraft**

I. Introduction

AERIAL robots, such as uninhabited aerial vehicles (UAVs) and passenger-fairing electric Vertical Take-off and Landing (eVTOL) aircraft, are becoming increasingly commonplace, emphasizing the need for robust Fault Detection, Isolation, and Recovery (FDIR) methods. Operating these aircraft pose significant challenges and risks, particularly when navigating above densely populated areas where a major malfunction could jeopardize lives and assets. Thus, there is a great interest in improving the survivability of these aircraft to events such as motor and actuator failures.

One method for increasing the tolerance of an aircraft to a motor failure is to use highly-redundant control actuation schemes. Although this redundancy intends to secure sufficient control authority for recovery following a failure, it amplifies the complexity of the control system. The usual approach involves assessing the system using fault tree analysis, accounting for all predictable failure situations. However, as system complexity rises, the chance of unforeseen failures also escalates. This poses a significant challenge in designing and implementing robust fault-tolerant control (FTC) systems equipped to handle potential failure scenarios. This paper presents a set of new methods for robust, data-driven fault-tolerant control called Neural-Fly for Fault Tolerance (NFFT). The core method consists of an adaptive fault diagnosis algorithm and a rapid fault compensation scheme (Fig. 1) that detects actuator failures without direct motor sensing and reallocates control without the need for a hard-coded response. These methods are validated via real-world flight tests and show up to a $48\,\%$ improvement in high-speed position tracking when compared to the baseline. The major contributions of NFFT are 1) a sparse fault identification algorithm that works in conjunction with rapid adaptation of deep neural networks for adaptive flight control, Neural-Fly [1], without the need for direct motor sensing; 2) a novel control allocation scheme that effectively leverages both the learned dynamics and failure identification to redesign the control allocation matrix and maximize control authority; and 3) demonstrations through real-world flight tests providing compelling examples of the benefits of this work.

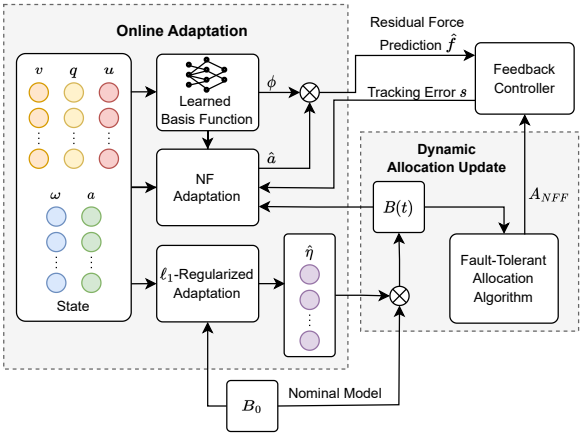

Fig. 1. Neural-Fly for Fault Tolerance (NFFT) combines residual force estimation with sparse actuator failure identification without direct sensing of the actuator states through ℓ_1-regularized adaptive update policy. The system compensates for faults by dynamically updating the control allocation through online optimization based on the observed aircraft behavior.

Related work. The current pool of fault-tolerant control research commonly divides the problem of recovering from faults into two discrete steps: fault diagnosis, which encompasses detection and isolation, and FTC. Recent surveys [2], [3] offer extensive insights into a range of fault diagnosis methodologies. Yet, a significant portion of the research focuses solely on FTC, without considering the preceding steps for fault diagnosis. For known faults that leave the aircraft under-actuated, [4]–[6] employ strategies to let the aircraft rotate freely about a fixed axis in order to maintain position. For over-actuated systems, control allocation algorithms, highlighted by [7]–[13] prove particularly valuable. Not only do these algorithms aid in designing robust vehicle configurations [8], [9], but they also serve as a method for real-time response to identified faults [7]–[9], [12], [13]. Similarly, [14] presents methods for analyzing vehicle performance limits under specific failures.

Some works address both fault diagnosis and FTC si-

The first two authors contributed equally. The authors are with the Division of Engineering and Applied Science, California Institute of Technology.

This work was funded by Supernal and in part funded by Defense Advanced Research Projects Agency (DARPA).

multaneously. Early studies [15], [16] utilized the Interacting Multiple Model approach, propagating the probabilities of various failure scenarios and subsequently using these probabilities to fuse the ideal estimated state or control command for each identified failure. Newer studies prioritize identifying the most likely fault and performing discrete switches between nominal and fault-compensating control methodologies [17]–[19]. Some work [20] builds upon adaptive control tools, though they require persistent excitation of the failure detection signal. These limitations are addressed by [21], which records key previous measurements, and [22], which uses both output tracking performance and direct sensor measurements to design robust control schemes. Some recent works explore learning-based methods for fault diagnosis [23], [24] and recovery. [25] presents FTC scheme utilizing supervisory reinforcement learning (SRL) while [26] employs adaptive NN-based FTC with automatically regulated neural network weight parameters.

It is important to highlight that numerous robust control and adaptive control algorithms, initially designed for different purposes, can be effectively utilized in fault diagnosis and FTC. This holds particularly true for benign failures like partial propeller damage, icing, unexpected battery voltage drop and blade pitch locking, for which methods such as [1], [27]–[31] prove to be useful. These strategies deliver impressive performance under nominal conditions for highly over-actuated vehicles but lose effectiveness in severe cases where system controllability is dramatically altered due to a complete actuator failure.

Despite the existence of these methods for FDIR and FTC, a substantial gap in knowledge persists in dealing with unforeseen failures not accounted for during the design phase of fault mitigation systems. This study bridges the gap by deriving our NFFT method, which utilizes only the aircraft's response to control commands to isolate a failure, eliminating the need for direct sensing of a predetermined, enumerated set of potential failure modes. This NFFT method is also designed to maintain the original performance of the system, enabling safe recovery during any operational mode. Further, by building on our prior work [1], NFFT facilitates precise and agile control even amidst time-varying disturbances and faults by utilizing rapid adaptation of deep learning networks with stability guarantees.

II. Nominal Control and Control Allocation

Consider a multirotor with dynamics of the form

$$\dot{x} = f(x) + B(t)u + g(x,u,t) + d(t) \tag{1}$$

where $x \in \mathbb{R}^n$ is the n-dimensional state vector, $f(x)$ is the estimated nominal dynamics model, $B(t)$ is an unknown control actuation matrix that is approximated by $B_0 \approx B(t)$ when the system is operating nominally, $u \in \mathbb{R}^m$ is the m-dimensional control input, $g(x,u,t)$ is the unknown residual dynamics, and $d(t)$ is external disturbance. Without loss of generality, assume that $u \in [0,1]^m$ and the maximum control force $\tau_{\max,i} = \max_u |(B_0 u)_i|$.

The measurement y of $g(x,u,t)$ with noise ϵ is given as

$$y = g(x,u,t) + \epsilon, \tag{2}$$

For small $g(x,u,t)$ and $d(t)$ in the nominal operating condition, the system is exponentially stabilized by

$$u = B_0^{\text{-}R}(-K\tilde{x} + \dot{x}_d - f(x)), \tag{3}$$

where K is a positive definite gain matrix, x_d is the desired state, $\tilde{x}$ is $x - x_d$, and $B_0^{\text{-}R}$ is any right inverse of B_0 such that $B_0 B_0^{\text{-}R} = I$.

In practice, $B_0^{\text{-}R}$ acts as a control allocation matrix that maps the required forces and torques to the control input u and for over-actuated systems, $B_0^{\text{-}R}$ is not unique. For a given estimated actuation matrix, $\hat{B}$, finding the control allocation matrix reduces to finding A such that ideally $A = \hat{B}^{\text{-}R}$ and $\hat{B}A = I$. If $\hat{B}A$ is not diagonal, the system will have some control-induced cross-coupling between the control axes. If $\hat{B}A$ is diagonal but not the identity matrix, then there will not be any control-induced cross coupling, however, the system will not achieve the expected control efforts.

III. Neural-Fly Adaptation with Fault Tolerance

Under actuator faults, the nominal control actuation matrix is altered, and is expressed in the form

$$B(t) = B_0 H(t), \quad H(t) = \text{diag}\,(\eta(t)) \tag{4}$$

where $\eta(t) = (\eta_1, \cdots, \eta_m)^\top$; $\eta_i(t) \in [0,1]$ is the effectiveness of the ith actuator at time t (0 denotes complete failure).

We present the following model to estimate the residual dynamics $g(x,u,t)$ from (1)

$$\hat{g}_{\text{NFFT}}(x,u,t) = B_0(\hat{H} - I)u + \phi(x,u)\hat{a}(t) \tag{5}$$

where $\hat{g}_{\text{NFFT}} \approx g(x,u,t)$ is a combination of $B_0(\hat{H} - I)u$ which is the estimate of model mismatch due to actuator faults, and $\phi(x,u)\hat{a}(t)$ denotes Neural-Fly's learned residual model of the time-varying aerodynamic forces [1]. Consequently, the control law u for (1) has the form

$$u = u_{\text{NFFT}} = (B_0\hat{H})^{\text{-}R}(-K\tilde{x} + \dot{x}_d - f(x) - \phi\hat{a}) \tag{6}$$

The method for obtaining the estimated motor effectiveness matrix $\hat{H}$ is presented in Sec. III-A. The method for obtaining the learned residual model $\phi\hat{a}$ is presented in Sec. III-B. Finally, the method for determining the optimal control allocation $A_{\text{NFFT}} = (B_0\hat{H})^{\text{-}R}$ in the presence of fault is presented in Sec. III-C.

A. Sparse Failure Identification

For discussion of FTC when residual and disturbance forces are negligible (such as during hover), we employ the simpler model of g and control law u for (1):

$$\begin{aligned} \hat{g}_{\text{FT}}(x,u) &= B_0\hat{H}u - B_0 u = B_0(\hat{H} - I)u, \\ u &= u_{\text{FT}} = (B_0\hat{H})^{\text{-}R}(-K\tilde{x} + \dot{x}_d - f(x)) \end{aligned} \tag{7}$$

In (7), $\hat{g}_{\text{FT}} \approx g(x,u,t)$ represents the mismatch between the expected and actual propeller force outputs due to actuator

faults, and $\hat{H}$ is the diagonal matrix of estimated actuator effectiveness $\hat{\eta}$.

The goal of this scheme is to estimate $\hat{\eta}$ online without directly sensing the actuator states, allowing computation of the optimal control allocation $A_{\text{FT}} = (B_0\hat{H})^{-R}$ to maintain nominal performance characteristics in the presence of actuator failure. We employ an ℓ_1-regularized adaptive update policy for such estimation. This is an effective regularization term for sparse parameter estimation because it encourages sparse solutions without requiring a hard constraint on the number of non-zero parameters or iteration through many non-zero parameter combinations.

1) Continuous Update Law: Consider the following least squares cost function for continuous time t

$$J(\hat{\eta}) = \int_0^t \mathrm{e}^{-\omega_f(t-r)} \left\| y - \hat{g}_{\text{FT}} \right\|^2 \mathrm{d}r + 2\gamma\|\hat{\eta} - 1\|_1 \quad (8)$$

where y is the noisy measurement from (2), ω_f is the exponential forgetting rate, and γ is the regularization factor. Equation (8) can be rearranged by defining $\bar{\eta} = \hat{\eta} - 1$, $U = \operatorname{diag}(u)$, and $\hat{g}_{\text{FT}} = B_0 U\bar{\eta}$ to yield

$$J(\bar{\eta}) = \int_0^t \mathrm{e}^{-\omega_f(t-r)} \left\| y - B_0U(r)\bar{\eta} \right\|^2 \mathrm{d}r + 2\gamma\|\bar{\eta}\|_1 \quad (9)$$

We approximate $\|\bar{\eta}\|_1$ such that for i actuators, $\|\bar{\eta}\|_1 \approx \sum_i \sqrt{\bar{\eta}_i^2 + \epsilon} = \|\bar{\eta}\|_{1,\epsilon}$ and $\lim_{\epsilon\to 0}\sum_i \sqrt{\bar{\eta}_i^2+\epsilon} = \|\bar{\eta}\|_1$.

Then the cost function (9) is approximated $J(\bar{\eta},\epsilon)$ s.t. $\lim_{\epsilon\to 0} J(\bar{\eta},\epsilon) = J(\bar{\eta})$, where $J(\bar{\eta},\epsilon)$ is given by $J(\bar{\eta},\epsilon) = \int_0^t \mathrm{e}^{-\omega_f(t-r)}\left(y^\top y - 2y^\top B_0U\bar{\eta} + \bar{\eta}^\top U B_0^\top B_0 U\bar{\eta}\right)\mathrm{d}r 2\gamma\sum_i \sqrt{\bar{\eta}_i^2+\epsilon}$ (10)

Since this cost function is convex in $\bar{\eta}$, the minimum value is obtained when $\frac{\partial J}{\partial \bar{\eta}} = 0$, leading to

$$\frac{\partial J}{\partial \bar{\eta}} = \int_0^t \mathrm{e}^{-\omega_f(t-r)}\left(-2UB_0^\top y + 2UB_0^\top B_0U\bar{\eta}\right)\mathrm{d}r \quad (11)$$

$$\bar{\eta} = P_\eta \int_0^t \mathrm{e}^{-\omega_f(t-r)} U(r)B_0^\top y \mathrm{d}r \quad \text{where } P_\eta = \quad (12)$$

$$\left(\gamma\cdot\operatorname{diag}\left[\tfrac{1}{\sqrt{\bar{\eta}_1^2+\epsilon}},\tfrac{1}{\sqrt{\bar{\eta}_2^2+\epsilon}},\cdots\right] + \int_0^t \mathrm{e}^{-\omega_f(t-r)}U(r)B_0^\top B_0U(r)\mathrm{d}r\right)^{-1}$$

If $\bar{\eta}_i = 0$, $\exists i$ (typically for ℓ_1 norm minimization), then $\frac{1}{\sqrt{\bar{\eta}_i^2+\epsilon}} = \frac{1}{\sqrt{\epsilon}}$. Otherwise, as $\lim_{\epsilon\to 0}\frac{1}{\sqrt{\bar{\eta}_i^2+\epsilon}} = \frac{1}{|\bar{\eta}_i|}$.

By using $\dot{P}_\eta = -P_\eta\frac{\mathrm{d}\left(P_\eta^{-1}\right)}{\mathrm{d}t}P_\eta$, we obtain

$$\dot{P}_\eta = \omega_f P_\eta - P_\eta\left(\gamma\mathcal{G}(\bar{\eta},\epsilon) + U(t)^\top B_0^\top B_0U(t)\right)P_\eta \quad (13)$$

where $\mathcal{G}(\bar{\eta},\epsilon) = \operatorname{diag}\left[\frac{\bar{\eta}_1+\omega_f\bar{\eta}_1^2+\omega_f\epsilon}{\left(\bar{\eta}_1^2+\epsilon\right)^{3/2}}, \frac{\bar{\eta}_2+\omega_f\bar{\eta}_2^2+\omega_f\epsilon}{\left(\bar{\eta}_2^2+\epsilon\right)^{3/2}},\cdots\right]$.

Now, we can derive a recursive update law for $\bar{\eta}$:

$$\begin{aligned}\dot{\bar{\eta}} = {} & P_\eta\left(U(t)^\top B_0^\top y(t) - \omega_f\int_0^t \mathrm{e}^{-\omega_f(t-r)}U(r)B_0^\top y(r)\mathrm{d}r\right) \\ & + \left(\omega_f P_\eta - P_\eta\left(\gamma\mathcal{G}(\bar{\eta},\epsilon) + U(t)B_0^\top B_0U(t)\right)P_\eta\right) \\ & \cdot\left(\int_0^t \mathrm{e}^{-\omega_f(t-r)}U(r)B_0^\top y(r)\mathrm{d}r\right) \quad (14)\end{aligned}$$

$$\dot{\bar{\eta}} = P_\eta U(t)^\top B_0^\top\left(y - B_0U(t)\bar{\eta}\right) - \gamma P_\eta\mathcal{G}(\bar{\eta},\epsilon)\cdot\bar{\eta} \quad (15)$$

Integrating (15) over time t produces the minimum solution to (10).

2) Stability Analysis: Assume that we design a control allocation matrix A such that $B_0\hat{H}A = \mathbb{I}$.

The closed loop dynamics are

$$\begin{aligned}\dot{x} &= f(x) + B_0Hu + g(x,u,t) + d(t) \\ &= -K\tilde{x} + \dot{x}_d - B_0\tilde{H}u + d_2(t) \quad (16)\\ \dot{\tilde{x}} &= -K\tilde{x} - B_0U\tilde{\eta} + d_2(t) \quad (17)\end{aligned}$$

where $\tilde{\eta} = \hat{\eta} - \eta$, $\tilde{x} = x - x_d$, $\tilde{H} = \hat{H} - H$ and $d_2(t) = g(x,u,t) + d(t)$.

By direct online optimization (e.g., the solution given in (12)), the ℓ_1 sparsity condition is fully enforced for m control inputs. In contrast, numerical integration using (13) and (15) is subject to integration errors and violation of the sparsity condition, while eradicating the need for online optimization.

Theorem 1: For bounded disturbance d, the tracking error $\tilde{x}$ using u_{FT}, given by (7) and (15), will be exponentially bounded in the sense of finite-gain $\mathcal{L}_p$ stability [32].

Proof: Consider the following Lyapunov function: $\mathcal{V} = \begin{bmatrix}\tilde{x} & \tilde{\eta}\end{bmatrix}^\top \mathcal{M}\begin{bmatrix}\tilde{x} & \tilde{\eta}\end{bmatrix}$ where $\mathcal{M} = \begin{bmatrix} I & 0 \\ 0 & P_\eta^{-1}\end{bmatrix}$. The derivative is computed as follows:

$$\begin{aligned}\dot{\mathcal{V}} &= 2\tilde{x}^\top\dot{\tilde{x}} + 2\tilde{\eta}^\top P_\eta^{-1}\dot{\tilde{\eta}} + \tilde{\eta}^\top\frac{\mathrm{d}\left(P_\eta^{-1}\right)}{\mathrm{d}t}\tilde{\eta} \\ &= -\begin{bmatrix}\tilde{x}\\ \tilde{\eta}\end{bmatrix}^\top\begin{bmatrix}2K & B_0U \\ UB_0^\top & U^\top B_0^\top B_0U + \gamma\mathcal{G}(\bar{\eta},\epsilon) + \omega_f P_\eta^{-1}\end{bmatrix}\begin{bmatrix}\tilde{x}\\ \tilde{\eta}\end{bmatrix} \\ &\quad + 2\begin{bmatrix}\tilde{x}\\ \tilde{\eta}\end{bmatrix}^\top\begin{bmatrix} d_2 \\ U^\top B_0^\top d - \gamma\mathcal{G}(\bar{\eta},\epsilon)\cdot\eta - P_\eta^{-1}\dot{\eta}\end{bmatrix} \quad (18)\end{aligned}$$

Since P_η^{-1} being bounded and uniformly positive definite, there exists some $\alpha > 0$ such that

$$-\begin{bmatrix}2K & B_0U \\ UB_0^\top & U^\top B_0^\top B_0U + \gamma\mathcal{G}(\bar{\eta},\epsilon) + \omega_f P_\eta^{-1}\end{bmatrix} \le -2\alpha\mathcal{M} \quad (19)$$

Note that (19) is satisfied if $2K - 2\alpha I > 0$ and

$$UB_0^\top B_0U + \gamma\mathcal{G} + (\omega_f - 2\alpha)P_\eta^{-1} - UB_0^\top\left(2K - 2\alpha I\right)^{-1}B_0U > 0 \quad (20)$$

A sufficient condition for (20) is $2K - 2\alpha I - I > 0$, $\omega_f - 2\alpha \ge 0$ since $P_\eta^{-1} > 0$ is true by definition, $\gamma\mathcal{G}(\eta,\epsilon) > 0$ and $UB_0^\top B_0U - UB_0^\top\left(2K - 2\alpha I\right)^{-1}B_0U > 0$ are also true. Using (18), (19) and the Cauchy-Schwartz inequality, $\dot{\mathcal{V}}$ can be bounded as follows:

$$\dot{\mathcal{V}} \le -2\alpha\mathcal{V} + 2\sqrt{\mathcal{V}}D, \quad (21)$$

where $D = \left\|\begin{bmatrix} d_2 \\ P_\eta^{1/2}UB_0^\top d - \gamma P_\eta^{1/2}\mathcal{G}(\bar{\eta},\epsilon)\eta - P_\eta^{-1/2}\dot{\eta}\end{bmatrix}\right\|$.

Consider the related system $\mathcal{W}$ where $\mathcal{W} = \sqrt{\mathcal{V}}$ and $2\mathcal{W}\dot{\mathcal{W}} = \dot{\mathcal{V}}$. Then, from the Comparison Lemma [32]

$$\sqrt{\mathcal{V}} \le w(t) \le \mathrm{e}^{-\alpha t}\left(w(0) - \sup_t\frac{D(t)}{\alpha}\right) + \sup_t\frac{D(t)}{\alpha} \quad (22)$$

Hence, $\sqrt{\mathcal{V}}$ and also $\|\tilde{x}\|$ exponentially converges to the ball $\sup_t\frac{D}{\alpha}$. ∎

Remark 1: Seemingly, the proof is complete at this point. However, we have not yet shown that D is bounded. By assumption, η, d, and $\dot{\eta}$ are uniformly bounded, and B_0, γ, ω_f, and ϵ are constants. D is uniformly bounded if $P_\eta^{1/2}$, $P_\eta^{-1/2}$, $\bar{\eta}$, and U are initially bounded and continuous.

Note that $P_\eta^{1/2}$ and $P_\eta^{-1/2}$ are uniformly bounded if P_η^{-1} is uniformly positive definite and uniformly bounded, respectively. Uniform positive definiteness is guaranteed by uniform positive definiteness of $\bar{\eta}$. Uniform boundedness is guaranteed by uniform boundedness of U and $\bar{\eta}$. $\bar{\eta}$ is uniformly bounded if η is uniformly bounded and $\tilde{\eta}$ is uniformly bounded. Also, U is uniformly bounded if $\tilde{x}$ is uniformly bounded and $(B_0\bar{\eta})^{\text{-}R}$ is uniformly bounded.

While precise conditions for uniform boundedness of $(B_0\bar{\eta})^{\text{-}R}$ is difficult to write out, it is clear that $\bar{\eta} \to 0$ as $\gamma \to \infty$. We also observe that $P_\eta^{1/2} \sim U, \bar{\eta}$, so for small γ, D will be dominated by the term $P_\eta^{1/2} U B_0^\top d$. For sufficiently large γ, $(B_0\bar{\eta})^{\text{-}R}$ is bounded. Lastly, for very large γ, no adaptation will occur and the system will maintain the baseline performance. Thus, there is an inherent design trade off between the degree of regularization and the nominal modeling errors not captured by the effectiveness adaptation model.

3) Discrete Update Law: The analogous discrete time least squares cost function for time step k is

$$J_k(\bar{\eta}) = \sum_{i=0}^{k} e^{-\omega_f(t_k - t_i)} (y_i - \hat{g})^\top (y_i - \hat{g}) + \gamma \|\bar{\eta}\|_1 \tag{23}$$

$$= \sum_{i=0}^{k} e^{-\omega_f(t_k - t_i)} \left(y_i^\top y_i - 2 y_i^\top \hat{g} + \hat{g}^\top \hat{g} \right) + \gamma \|\bar{\eta}\|_1$$

$$= \left(\sum_{i=0}^{k} e^{-\omega_f(t_k - t_i)} y_i^\top y_i \right) - 2 \left(\sum_{i=0}^{k} e^{-\omega_f(t_k - t_i)} y_i^\top B_0 U_i \right) \bar{\eta}$$

$$+ \bar{\eta}^\top \left(\sum_{i=0}^{k} e^{-\omega_f(t_k - t_i)} U_i B_0^\top B_0 U_i \right) \bar{\eta} + \gamma \|\bar{\eta}\|_1$$

where $\hat{g} = \hat{g}_{\text{FT}}$ from (7), and (23) is a convex function of $\bar{\eta} = \hat{\eta} - 1$. So we can easily solve for the optimal $\hat{\eta}$, using a number of optimization problem solving tools such as [33], [34]. During online computation, we quickly incorporate a new measurement by scaling the old summation terms by the exponential forgetting term $e^{(-\omega_f(t_k - t_{k-1}))}$ and adding the k'th term.

In addition, we apply the following thresholds and linear fit to the optimally solved effectiveness factors.

$$\hat{\eta}_i = \begin{cases} 1 & \text{if } \hat{\eta}_i \geq \zeta \\ \frac{\hat{\eta}_i - \xi}{\zeta - \xi} & \text{if } \xi \leq \hat{\eta}_i < \zeta \\ 0 & \text{if } \hat{\eta}_i < \xi \end{cases} \tag{24}$$

where ζ and ξ are upper and lower threshold parameters, respectively, such that $\zeta, \xi \in [0, 1]$ and $\zeta > \xi$. This scheme allows for a more aggressive update of $B_0 \hat{H}(t)$ by treating actuators with a sufficiently low effectiveness estimate as completely failed, and treating those with a sufficiently high effectiveness as fully functional, with smooth shaping in between for stability. Additionally, this output shaping captures the behavior of real-world motor failures as motors are highly unlikely to be functional but significantly degraded.

B. Neural-Fly Dynamics Adaptation for Real-Time Learning

As will be shown in Sec. IV, the performance of effectiveness ($\bar{\eta}$) adaptation based on discrete time cost function (23) degrades when the aircraft is in forward flight and aerodynamic forces are introduced. To address this limitation, we follow prior work [1] to combine sparse failure identification with Neural-Fly to form NFFT (6).

Consider Neural-Fly's learned residual model of the time-varying aerodynamic forces, and its adaptive controller

$$\hat{g}_{\text{NF}}(x, u, t) = \phi(x, u)\hat{a}(t),$$
$$u = u_{\text{NF}} = B_0^{\text{-}R}(-K\tilde{x} + \dot{x}_d - f(x) - \phi\hat{a}) \tag{25}$$

For both u_{NFFT} (6) and u_{NF} (25), the basis function $\phi(x, u)$ is trained through the DAIML algorithm [1] in the offline learning phase, while the time-varying linear weights $\hat{a}$ are updated online using the Kalman filter-based composite adaptation law and covariance update equation

$$\dot{\hat{a}} = -\lambda\hat{a} - P_a\phi^\top R^{-1}(\phi\hat{a} - y) + P_a\phi^\top s \tag{26}$$

$$\dot{P}_a = -2\lambda P_a + Q - P_a\phi^\top R^{-1}\phi P_a \tag{27}$$

where P_a is a covariance-like matrix for automatic gain tuning, s is the composite tracking error, and λ, R, and Q can be tuned similar to Kalman filtering.

Although $\hat{g}_{\text{NF}}$ alone yields reasonable aerodynamic residual force estimates and position tracking performance during nominal flight, actuator failures cause the plant to be severely altered, resulting in erroneous force estimates and flight performance. To further highlight this, we investigate the stability requirement for Neural-Fly. [35] showed that spectral normalization of a learned residual model for g guarantees the existence of a stabilizing control solution and robustness of the augmented control. [1] extended the learned-residual control framework to online learning of dynamics, allowing effective and robust adaptation of a pretrained model to a time-varying conditions. The learned model of the dynamics is incorporated into the control scheme with the following iteratively updated control law for iteration step k:

$$u_k = B_0^{\text{-}R}\tau_{d,k}, \ \tau_{d,k} = \dot{x}_d - K\tilde{x} - f(x) - \hat{g}_{\text{NF}}(x, u_{k-1}, t) \tag{28}$$

where the fixed point iteration $\hat{g}_{\text{NF}}(x, u_{k-1}, t) \approx \hat{g}_{\text{NF}}(x, u_k, t)$ is used to capture the non-affine control problem that arises from the learned model. To ensure stability, the fixed point iteration requires that the Lipschitz constant of the iteration is less than one, which is true when

$$\sigma(B_0^{\text{-}R}) \cdot \mathcal{L}_u(\hat{g}(x, u, t)) < 1 \tag{29}$$

where $\sigma(B_0^{\text{-}R})$ is the spectral norm of $B_0^{\text{-}R}$ and $\mathcal{L}_u(\hat{g}(x, u, t))$ is the Lipschitz constant of $\hat{g}(x, u, t)$ with respect to u. Furthermore, [35] showed that the exponential convergence rate of the closed-loop system depends on the

convergence rate of this fixed-point iteration. In particular, the exponential convergence rate, α, is proportional to $(\lambda_{\max}(K) - \rho)$, where ρ bounds the one-step difference in the control input, such that $\|u_k - u_{k-1}\| \leq \rho\|\tilde{x}_k\| \approx \rho\|\tilde{x}_{k-1}\|$. For actuator failures in the form of (4), the Lipschitz constant is given by $\mathcal{L}_u(g(x,u,t)) = \max_{i,t}(\eta_i - 1)$, and, therefore, an effective learned model of the faulty dynamics will have a large Lipschitz constant. Because of this Lipschitz constant requirement, the control law in (28) can break down.

Therefore, we arrive at aforementioned (5), (6) combining (7) which estimates the actuator effectiveness and dynamically updates the corresponding control allocation, and (25) which estimates the aerodynamic residual forces through offline learning and composite adaptation.

The discrete-time least-squares cost function (23) is still valid for this combined approach, except now $\hat{g} = \hat{g}_{\text{NFFT}}$ (5). This approach prevents the system from erroneously estimating actuator effectiveness based on a naïve measurement of the residual dynamics that does not take into account aerodynamic disturbance forces such as drag.

Theorem 2: Similar to Theorem 1 and (18), the tracking error $\tilde{x}$ using u_{NFFT} of (6) will be exponentially bounded.

Proof: The derivative of the Lyapunov function $\mathcal{V}$ for u_{NFFT} is now extended to take the form

$$\begin{aligned}\dot{\mathcal{V}} &= 2\tilde{x}^\top\dot{\tilde{x}} + 2\tilde{\eta}^\top P_\eta^{-1}\dot{\tilde{\eta}} + 2\tilde{a}^\top P_a^{-1}\dot{\tilde{a}} + \tilde{\eta}^\top\dot{P}_\eta^{-1}\tilde{\eta} + \tilde{a}^\top\dot{P}_a^{-1}\tilde{a} \\ &= -\begin{bmatrix}\tilde{x}\\ \tilde{\eta}\\ \tilde{a}\end{bmatrix}^\top \begin{bmatrix} 2K & 0 & 0 \\ 0 & U^\top B_0^\top B_0 U + \gamma\mathcal{G} + \omega_f P_\eta^{-1} & \phi^\top(I+R^{-1})B_0U \\ 0 & U^\top B_0^\top(I+R^{-1})\phi & \phi^\top R^{-1}\phi + P_a^{-1}QP_a^{-1}\end{bmatrix}\begin{bmatrix}\tilde{x}\\ \tilde{\eta}\\ \tilde{a}\end{bmatrix} \\ &\quad + 2\begin{bmatrix}\tilde{x}\\ \tilde{\eta}\\ \tilde{a}\end{bmatrix}^\top \begin{bmatrix} d \\ U^\top B_0^\top d - \mathcal{G}\cdot\eta - P_\eta^{-1}\dot{\eta} \\ \phi^\top R^{-1}d - P_a^{-1}\lambda a - P_a^{-1}\dot{a}\end{bmatrix} \end{aligned} \tag{30}$$

where $\tilde{x} = x - x_d$, $\tilde{\eta} = \hat{\eta} - \eta$, and $\tilde{a} = \hat{a} - a$. Similar to the stability proof of u_{FT}, we can find the bounded input and bounded output stability (finite-gain $\mathcal{L}_p$ stability with sufficiently large Q). This follows from $\phi^\top R^{-1}\phi + P_a^{-1}QP_a^{-1} - 2\alpha P_a^{-1} > \frac{1}{\kappa}UB_0^\top(I+R^{-1})\phi\phi^\top(I+R^{-1})B_0U$ and κ is the lower-bound of the first 2×2 given in (19) and since Q, λ can be chosen such that $P_a^{-1}QP_a^{-1} - 2\alpha P_a^{-1} > 0$, since $P_a^{-1} \geq 2\lambda/\|Q\| \cdot I$ if $P_a(0) \leq \frac{Q}{2\lambda}$ by Theorem 3. ∎

Theorem 3: $P_a^{-1} \geq 2\lambda/\|Q\| \cdot I$, $\forall t$ if $P_a(0) \leq \frac{Q}{2\lambda}$.

Proof: Consider $\mathcal{V}_{P_a}(t,y) = y^\top P_a y$, for $\|y\| \equiv 1$.

If $P_a(0) \leq \frac{Q}{2\lambda}$, then,

$\dot{\mathcal{V}}_{P_a} = y^\top(-2\lambda P_a + Q - P_a\phi^\top R^{-1}\phi P_a)y$

$\leq y^\top(-\lambda P_a + Q)y \implies \mathcal{V}_{P_a} \leq \|\frac{Q}{2\lambda}\|\|y\|^2 = \|\frac{Q}{2\lambda}\|$.

Therefore, $P_a^{-1} \geq 2\lambda/\|Q\| \cdot I$ as required by Theorem 2. ∎

C. Control Allocation through Online Optimization

Before we present a novel control allocation algorithm, we compare three prior approaches for control allocation. Because these methods do not explicitly consider the time-varying nature of the system, we will simply denote the control actuation matrix of interest as B.

A natural choice for the control allocation matrix, A, is the Moore-Penrose right pseudoinverse, which yields the minimum norm control input given any desired torque command. However, this solution does not account for actual power usage or control saturation, leading to potentially non-actionable solutions.

For a symmetric system, the allocation matrix that yields maximum control authority along each control axis, independently, is

$$A_{\text{mca}} = \text{sign}(B^\top) \tag{31}$$

This allocation scheme does not work under a single motor failure, where $B = B_0H(t)$, as it will cause $(B_0H)\text{sign}((B_0H)^\top)$ to become non-diagonal. This leads to cross-coupling in the different control axis and significantly degrades tracking performance.

The allocation algorithm proposed in [9] imposes constraints to a cost function to ensure that the solution is a valid control allocation matrix for B and that the thrust factors are non-negative. However, under an outboard motor failure for the system in Sec. IV, this method yields some thrust factors equal to 0 with non-zero torque factors. Thus, there are infinitesimally small torque commands that can cause the control to saturate.

Due to the limitations of prior approaches, we propose a novel allocation algorithm that directly maximizes the control authority at the nominal operating point. Let the commanded net-zero force thrust be $h \in [0,1]$, and A_T, A_R, A_P, A_Y be the columns of A that produce required actuator command given desired vertical thrust, roll torque, pitch torque, and yaw torque, respectively. Finally, let B_T, B_R, B_P, B_Y be the corresponding rows of B that govern the vertical thrust, and roll, pitch, and yaw moments.

The thrust for a given set of motor speeds is given by B_Tu. Thus, to achieve the maximum thrust with no torque, $u = A\tau_d = A_T$. To achieve the maximum torque along the roll axis with net-zero force, $u_d = A\tau_d = hA_T + A_R$. A similar approach can be taken for the pitch and yaw axes. Since the vehicle is asymmetric when experiencing a failure, we must consider both the positive and negative torque along each axis. Accounting for actuation limits, this leads to

$$\begin{aligned}\bar{A}_{\text{FT}} = &\underset{A}{\text{argmax}} \sum_{i\in\{R,P,Y\}} B_i(hA_T + A_i) - \sum_{i\in\{R,P,Y\}} B_i(hA_T - A_i) \\ &+ B_T(hA_T) - \|A - A_0\|_F \\ \text{s.t.}\quad & B_{R,P,Y}A_T = 0,\ 0 \leq A_T \leq 1, && \left.\begin{matrix}\\ \end{matrix}\right\}\ \text{thrust constraint} \\ & B_{T,P,Y}(hA_T + A_R) = [h,0,0]^\top, && \left.\begin{matrix}\\ \end{matrix}\right\}\ \text{roll} \\ & 0 \leq (hA_T \pm A_R) \leq \mathbb{1}, && \left.\begin{matrix}\\ \end{matrix}\right\}\ \text{constraint} \\ & B_{T,R,Y}(hA_T + A_P) = [h,0,0]^\top, && \left.\begin{matrix}\\ \end{matrix}\right\}\ \text{pitch} \\ & 0 \leq (hA_T \pm A_P) \leq \mathbb{1}, && \left.\begin{matrix}\\ \end{matrix}\right\}\ \text{constraint} \\ & B_{T,R,P}(hA_T + A_Y) = [h,0,0]^\top, && \left.\begin{matrix}\\ \end{matrix}\right\}\ \text{yaw} \\ & 0 \leq (hA_T \pm A_Y) \leq \mathbb{1}, && \left.\begin{matrix}\\ \end{matrix}\right\}\ \text{constraint}\end{aligned} \tag{32}$$

For failure scenarios, $B\bar{A}_{\text{FT}} \neq I$, however, $B\bar{A}_{\text{FT}}$ is diagonal. We rescale $\bar{A}_{\text{FT}}$ to ensure that $B\bar{A}_{\text{FT}} = I$, yielding the final control allocation matrix

$$A_{\text{FT}} = \bar{A}_{\text{FT}}\left(B\bar{A}_{\text{FT}}\right)^{-1} \tag{33}$$

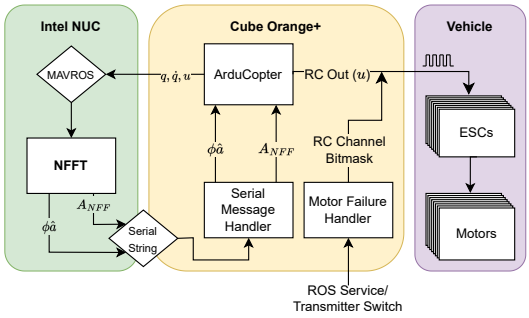

Fig. 2. The software schematic showing how the allocation and feedforward terms are sent to the flight controller.

Under nominal conditions, this exactly reproduces the solution from (31) and under a single motor failure, this algorithm maintains maximum control authority while preserving the nominal performance characteristics of the system, maximizing the probability of controllability of the aircraft. Furthermore, this formulation is not only convex, but also is a linear program, allowing it to be solved efficiently in real time with a numerical solver.

IV. Experimental Validation

A. Hardware Setup

Neural-Fly for Fault Tolerance was deployed on a custom-built octocopter (Fig. 2, left). The aircraft configuration is based upon the Autonomous Flying Ambulance [36], an eVTOL UAV designed to maximize controllability when one or more motors fail. The ocotocopter has a maximum motor-to-motor distance of $1.57\,\mathrm{m}$ and a mass of $9.4\,\mathrm{kg}$. The aircraft runs ArduCopter [37] on a Cube Orange+ and uses IMUs, barometers, magnetometers and an RTK GNSS to provide the state estimate. The motors are controlled via Dshot which provides motor telemetry for verifying the estimates of our algorithms. The aircraft is configured as a Dynamic Scripting Matrix frame class, allowing the control allocation matrix to be updated in-flight.

NFFT is run entirely onboard the aircraft using an NUC (AMD Ryzen 7 4800U CPU, 32 GB RAM) companion computer with ROS 2 and MAVROS for interfacing (Fig. 2). The transfer of the control allocation matrix and learned residual forces to ArduCopter uses a dedicated UART link and Lua scripting on the Cube interfaces this stream to the flight stack. A second Lua script is used to trigger motor failures, taking inputs from either a ROS service call or the R/C transmitter. The only change required to the base ArduCopter 4.4 code was to allow setting the PID feedforward terms via Lua scripting.

This architecture allows the computationally-costly algorithms to be run on the NUC whilst still guaranteeing the runtime performance of the flight stabilization loops. On the NUC, the adaptation algorithm incorporates new measurements in under $1\,\mathrm{ms}$, fast enough to enable measurements to be used as the raw sensor data is available. The computation of the control actuation matrix and residual forces take less than $11\,\mathrm{ms}$, and resolving the control allocation matrix takes less than $6\,\mathrm{ms}$. While the adaptation loop is run at $50\,\mathrm{Hz}$, the control allocation matrix is updated at only $10\,\mathrm{Hz}$, the approximate limit of how quickly the Lua script can accept and process data. The updated control allocation matrix row for the failed motor is kept at the nominal value to persistently excite the failure for the sparse failure identification algorithm.

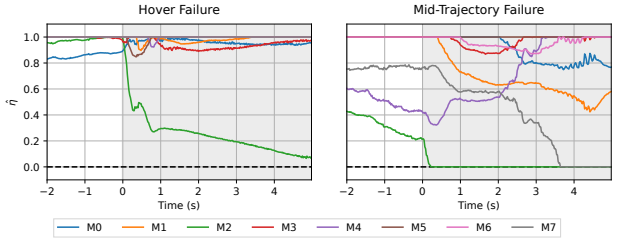

Fig. 3. Motor effectiveness estimate $\hat{\eta}$ without learned residual aerodynamic forces following rear left outer failure during hover (left) and during forward flight (right). The shaded area shows when the failure is active.

Fig. 4. Onboard footage shows the extreme departure from the desired attitude during outboard failures when NFFT is not running.

B. Experimental Performance

As discussed in Sec. III-B, although discrete ℓ_1-regularized actuator effectiveness adaptation (23) with the simple model of the unknown dynamics (7) is able to estimate reasonable actuator effectiveness factors during hover, it breaks down when the aircraft begins to maneuver and generate larger aerodynamic forces (Fig. 3). To mitigate this issue, the proposed NFFT controller (6) employs learned residual aerodynamic force prediction through online composite adaptation (26) and (27), in addition to the ℓ_1-regularized adaptation and the control allocation algorithm (32) and (33).

The training dataset used for DAIML [1] to learn representation ϕ of the unmodeled aerodynamic effects was collected during a 10-minute, manually flown flight along an arbitrary trajectory with the baseline controller. Velocity, quaternion attitude, and motor speed commands were collected at $50\,\mathrm{Hz}$ and were used as the input states. Importantly, periods with motor failures did not form part of the training dataset, nor did direct measurement of the motor speeds. Control performance was evaluated by analyzing the aircraft response after an actuator failure during $7.5\,\mathrm{m/s}$ forward flight tracking a rectangular trajectory (Fig. 4).

Fig. 5(a) shows the motor effectiveness factor estimate following a rear left outboard motor failure while tracking a trajectory. The adaptation algorithm correctly identifies the failed motor and estimates lowered effectiveness. Fig. 5(b) shows how the proposed allocation algorithm quickly reallocates actuator contributions among the remaining motors

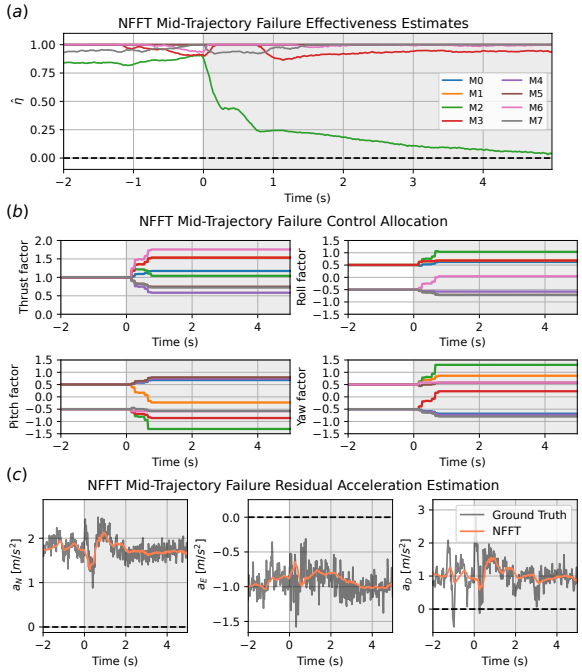

Fig. 5. Full NFFT adaptation and control allocation performance during rear left outer moto failure while tracking a trajectory at ground speed $7.5\,\mathrm{m/s}$. The shaded areas show the failure region. (a) Motor effectiveness $\hat{\eta}$ estimate (b) Change in control allocation factors following the failure. Each "factor" represents corresponding column of A_{NFF}. (c) $\phi\hat{a}$ estimate compared to the measured ground truth.

TABLE I

MAXIMUM/MEAN PEAK ATTITUDE ERRORS DURING FAILURES

	Peak R Error		Peak P Error		Peak Y Error	
Controller	max	mean	max	mean	max	mean
Inboard						
Baseline	4.75°	4.38°	8.05°	7.79°	8.55°	5.97°
Neural-Fly	4.66°	4.37°	7.80°	7.24°	7.77°	6.63°
NFFT	4.07°	3.98°	6.89°	6.43°	5.45°	5.24°
Omniscient	2.46°	2.01°	3.18°	2.74°	2.01°	1.69°
Outboard						
Baseline	8.53°	8.37°	9.64°	8.13°	14.95°	12.88°
Neural-Fly	10.42°	9.92°	10.98°	9.51°	21.13°	15.69°
NFFT	8.22°	7.45°	8.73°	7.38°	7.75°	6.99°
Omniscient	4.22°	3.57°	3.80°	3.36°	2.32°	1.91°

to maintain the original performance characteristics, reaching steady state in 0.66 seconds. The online aerodynamic residual force estimates $\hat{f} = \phi\hat{a}$, derived from the learning-based adaptation algorithm (Fig. 5c) are applied directly to the baseline PID position controller as feedforward to supplement the integral term, which has a slower response to unmodeled dynamics and time-varying disturbances.

Fig. 6 shows the aircraft attitude response following an inboard and an outboard actuator failure at $7.5\,\mathrm{m/s}$ forward flight, with errors averaged over four failure events for each case. Table I shows the maximum and mean peak attitude error for each case. The baseline ArduCopter response does not include any adaptation algorithm and relies on the built-in PID controller to stabilize the aircraft. The Neural-Fly controller in (25) applies learned residual dynamics as position control feedforward. The Neural-Fly controller, which relies

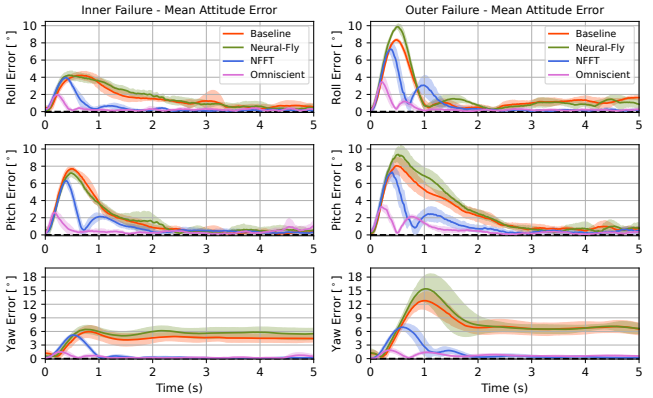

Fig. 6. Baseline, Neural-Fly, NFFT, and failure-omniscient controller attitude tracking performance. Solid curves show the mean tracking error over four 5-second failure events during forward flight. The shaded areas show standard deviation of the mean error.

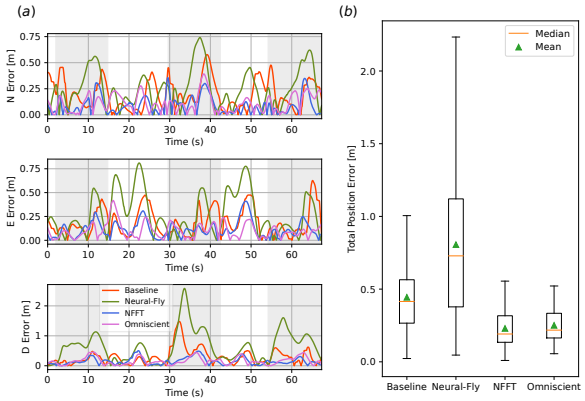

Fig. 7. Position error during rectangular trajectory tracking with outer motor failures. (a) Measured north, east, down position error while flying at ground speed $7.5\,\mathrm{m/s}$. Failures are toggled at three different points during the experiment are represented in gray regions. (b) Data distribution from the corresponding flights.

only on residual force adaptation, exhibits degraded attitude tracking performance compared to the baseline when the original control actuation matrix is significantly altered due to the failure. On extreme failure cases such as outer motor failures, NFFT on average reduces the peak attitude error by $11\,\%$, $9\,\%$, and $46\,\%$, for roll, pitch, and yaw axes, respectively. NFFT also significantly reduces the recovery time, with the exception of roll error during outer motor failure. It is noteworthy that while the baseline and Neural-Fly are unable to recover from the initial yaw deviation, NFFT recovers to within 0.5° yaw error in just 1.86 seconds. The failure-omniscient controller directly senses the motor failure and uses the same control allocation algorithm as NFFT to stabilize the system in real time. As such, it outperforms NFFT and can be considered an ideal system response to a known motor failure via optimal control allocation.

The learned feedforward term leads to improvement in aircraft position tracking performance under actuator failure only when combined with the optimized control allocation (Fig. 7(a), NFFT). If the motor failure is not accounted for, the abrupt change in the system plant significantly degrades

the controller's ability to track the desired position (Fig. 7(a), Neural-Fly). Fig. 7(b) shows the $74\,\%$ increase in mean position tracking error compared to the baseline while running Neural-Fly only, while the combined NFFT approach leads to a $48\,\%$ reduction in the mean position error. NFFT yields position tracking performance on par with that of the failure-omniscient controller, despite the faster control reallocation and attitude recovery (Fig. 6) due to perfect state information. Such results demonstrates the advantage of introducing fault-tolerant control scheme as an extension of Neural-Fly.

V. Conclusion

In this paper, we presented a new method for detecting and compensating for motor failures in multirotor aircraft without knowledge of the motor states. This method, coupled with Neural-Fly, is able to simultaneously predict unmodeled forces and motor failures, enabling precise tracking performance even under significant plant changes and uncertainties. Our proposed sparse failure identification method successfully isolates a failed motor within one second and correctly reallocates actuator effort without any direct observation of the motor. We also introduced the maximum-control-authority allocation scheme which has shown significant improvements over previous work, particularly in maximizing and maintaining control authority in the presence of faults. Our research represents a significant step forward in developing robust fault detection and compensation strategies for aircraft, enabling ever-safer skies for tomorrow.

References

[1] M. O'Connell, G. Shi, X. Shi, K. Azizzadenesheli, A. Anandkumar, Y. Yue, and S.-J. Chung, "Neural-Fly enables rapid learning for agile flight in strong winds," *Sci. Robot.*, 2022.

[2] G. K. Fourlas and G. C. Karras, "A Survey on Fault Diagnosis Methods for UAVs," in *2021 Int. Conf. Unmanned Aircraft Systems*, 2021, pp. 394–403.

[3] R. Puchalski and W. Giernacki, "UAV Fault Detection Methods, State-of-the-Art," *Drones*, vol. 6, no. 11, p. 330, 2022, number: 11 Publisher: Multidisciplinary Digital Publishing Institute.

[4] M. W. Mueller and R. D'Andrea, "Stability and control of a quadrocopter despite the complete loss of one, two, or three propellers," in *Int. Conf. on Robot. and Automat.*, 2014.

[5] N. P. Nguyen and S. K. Hong, "Fault Diagnosis and Fault-Tolerant Control Scheme for Quadcopter UAVs with a Total Loss of Actuator," *Energies*, p. 1139, 2019.

[6] J. Yeom, G. Li, and G. Loianno, "Geometric Fault-Tolerant Control of Quadrotors in Case of Rotor Failures: An Attitude Based Comparative Study," *arXiv:2306.13522*, 2023.

[7] M. W. Oppenheimer, D. B. Doman, and M. A. Bolender, "Control Allocation for Over-actuated Systems," in *Mediterranean Conf. Control and Autom.*, 2006, pp. 1–6.

[8] X. Shi, K. Kim, S. Rahili, and S.-J. Chung, "Nonlinear Control of Autonomous Flying Cars with Wings and Distributed Electric Propulsion," in *IEEE Conf. Decis. and Control*, 2018.

[9] K. Kim, S. Rahili, X. Shi, S.-J. Chung, and M. Gharib, "Controllability and Design of Unmanned Multirotor Aircraft Robust to Rotor Failure," in *AIAA Scitech Forum*, 2019.

[10] T. A. Johansen and T. I. Fossen, "Control allocation—A survey," *Automatica*, vol. 49, no. 5, pp. 1087–1103, 2013.

[11] O. Harkegard, "Efficient active set algorithms for solving constrained least squares problems in aircraft control allocation," in *Proc. IEEE Conf. Decis. and Control*, 2002, pp. 1295–1300.

[12] M. Mousaei, J. Geng, A. Keipour, D. Bai, and S. Scherer, "Design, Modeling and Control for a Tilt-rotor VTOL UAV in the Presence of Actuator Failure," in *IEEE/RSJ Int. Conf. Intel. Robot. and Syst.*, 2022, pp. 4310–4317.

[13] A. Baldini, R. Felicetti, A. Freddi, S. Longhi, and A. Monteriù, "Actuator Fault-Tolerant Control Architecture for Multirotor Vehicles in Presence of Disturbances," *J Intell Robot Syst*, pp. 859–874, 2020.

[14] S. Fuhrer, S. Verling, T. Stastny, and R. Siegwart, "Fault-tolerant Flight Control of a VTOL Tailsitter UAV," in *Int. Conf. Robot. Autom.*, 2019, pp. 4134–4140.

[15] C. Rago, R. Prasanth, R. Mehra, and R. Fortenbaugh, "Failure detection and identification and fault tolerant control using the IMM-KF with applications to the Eagle-Eye UAV," in *Proc. IEEE Conf. Decis. and Control*, vol. 4, 1998, pp. 4208–4213.

[16] H. Blom and Y. Bar-Shalom, "The interacting multiple model algorithm for systems with Markovian switching coefficients," *IEEE Trans. Automat. Contr.*, vol. 33, no. 8, pp. 780–783, 1988.

[17] M. Saied, H. Shraim, C. Francis, I. Fantoni, and B. Lussier, "Actuator fault diagnosis in an octorotor UAV using sliding modes technique: Theory and experimentation," in *Eur. Control Conf.*, 2015, pp. 1639–1644.

[18] A. Freddi, S. Longhi, and A. Monteriù, "Actuator fault detection system for a mini-quadrotor," in *IEEE Int. Symp. Ind. Electronics*, 2010, pp. 2055–2060.

[19] D. Rotondo, A. Cristofaro, T. A. Johansen, F. Nejjari, and V. Puig, "Detection of icing and actuators faults in the longitudinal dynamics of small UAVs using an LPV proportional integral unknown input observer," in *Conf. Control and Fault-Tolerant Syst.*, 2016, pp. 690–697.

[20] A. Vahidi, A. Stefanopoulou, and H. Peng, "Recursive least squares with forgetting for online estimation of vehicle mass and road grade: theory and experiments," *Vehicle Syst. Dyn.*, vol. 43, no. 1, pp. 31–55, 2005.

[21] G. Chowdhary and E. Johnson, "Concurrent learning for convergence in adaptive control without persistency of excitation," in *IEEE Conf. Decis. and Control*, 2010, pp. 3674–3679.

[22] E. N. Johnson and A. J. Calise, "Limited Authority Adaptive Flight Control for Reusable Launch Vehicles," *J. Guid., Control, and Dyn.*, vol. 26, no. 6, pp. 906–913, Nov. 2003.

[23] M. Saied, B. Lussier, I. Fantoni, H. Shraim, and C. Francis, "Fault Diagnosis and Fault-Tolerant Control of an Octorotor UAV using motors speeds measurements," *IFAC-PapersOnLine*, pp. 5263–5268, 2017.

[24] G. Iannace, G. Ciaburro, and A. Trematerra, "Fault Diagnosis for UAV Blades Using Artificial Neural Network," *Robotics*, p. 59, 2019.

[25] I. Ahmed, M. Quinones-Grueiro, and G. Biswas, "Adaptive fault-tolerant control of octo-rotor UAV under motor faults in adverse wind conditions," in *AIAA SciTech Forum*, 2023.

[26] S. Mokhtari, A. Abbaspour, K. K. Yen, and A. Sargolzaei, "Neural Network-Based Active Fault-Tolerant Control Design for Unmanned Helicopter with Additive Faults," *Remote Sens.*, 2021.

[27] M. Faessler, A. Franchi, and D. Scaramuzza, "Differential Flatness of Quadrotor Dynamics Subject to Rotor Drag for Accurate Tracking of High-Speed Trajectories," *IEEE Robot. Autom. Lett.*, 2018.

[28] S. Mallikarjunan, B. Nesbitt, E. Kharisov, E. Xargay, N. Hovakimyan, and C. Cao, "L1 Adaptive Controller for Attitude Control of Multirotors," in *AIAA Guid., Navigation, and Control Conf.*, 2012.

[29] D. Hanover, P. Foehn, S. Sun, E. Kaufmann, and D. Scaramuzza, "Performance, Precision, and Payloads: Adaptive Nonlinear MPC for Quadrotors," *arXiv:2109.04210*, 2021.

[30] E. Tal and S. Karaman, "Accurate Tracking of Aggressive Quadrotor Trajectories Using Incremental Nonlinear Dynamic Inversion and Differential Flatness," in *IEEE Conf. Decis. and Control*, 2018.

[31] J. Mao, J. Yeom, S. Nair, and G. Loianno, "From Propeller Damage Estimation and Adaptation to Fault Tolerant Control: Enhancing Quadrotor Resilience," *arXiv:2310.13091*, 2023.

[32] H. Khalil, *Nonlinear Systems*, 3rd ed. Prentice Hall, 2002.

[33] S. Diamond and S. Boyd, "CVXPY: a python-embedded modeling language for convex optimization," *J. of Machine Learn. Res.*, 2016.

[34] A. Agrawal, R. Verschueren, S. Diamond, and S. Boyd, "A Rewriting System for Convex Optimization Problems," *arXiv:1709.04494*, 2019.

[35] G. Shi et al., "Neural Lander: Stable Drone Landing Control using Learned Dynamics," *Int. Conf. Robot. Autom.*, 2019.

[36] E. Tang, P. Spieler, M. Anderson, and S.-J. Chung, "Design of the Next-Generation Autonomous Flying Ambulance," in *AIAA Scitech Forum*, 2021.

[37] ArduPilot, "ArduPilot." [Online]. Available: https://ardupilot.org/