An Automated Approach for the Design of a Fault Tolerant Controller

Liam Vile, Halim Alwi and Christopher Edwards

***Abstract*— In this paper a fault tolerant sliding mode control allocation law is proposed. The controller is designed through a particle swarm optimization method which ensures the closed loop system is optimal for multiple design objectives - including robustness to uncertainties in the actuator fault and failure information. The approach is tested for the control of blended wing body aircraft's lateral dynamics.**

I. Introduction

In the open literature, Sliding Mode Control (SMC) and Control Allocation (CA) is a popular combination for Fault Tolerant Control (FTC). SMC is a robust control methodology which can be designed to guarantee robustness against an unknown (but bounded) matched uncertainty [1], whereas CA is a method of reconfiguring the distribution of the control signal to the physical actuators, independently of the closed-loop system's performance [2]. This combination has been shown to be effective for the FTC of fixed-wing aircraft [3], [4], quad-copters [5], [6] and automobiles [7] as well as many other over-actuated systems. In these works the CA mechanism uses knowledge of any actuator faults and failures to redistribute the control effort and maintain the system's performance. In reality exact knowledge of actuator faults and failures is difficult to obtain and so only an approximation of the actuator's health (generated by a Fault Detection and Isolation scheme) is available which introduces uncertainty into the closed-loop system. Several works, such as [3] and [4], have considered the effects of this uncertainty on the system's stability and provide rigorous bounds on the uncertainty for which stability can be guaranteed.

Typically the process of control design is a laborious one, requiring a designer who is knowledgeable, with regards to the underlying control methodology, to manually tune the controller gains for a particular system. Not only is this a time consuming (and potentially expensive) process, but the resulting controller is not guaranteed to be an optimal choice. In recent years Multiple Objective Optimisation (MOO) has gained popularity to help expedite this process. In [8] a Genetic Algorithm (GA) approach is proposed for the design of a SMC to control the water levels of two inter-connected tanks. In this case the objectives of minimising tracking error, rise-time and overshoot were individually weighted and conflated into a Single Objective Optimisation (SOO) problem. A similar approach is taken in [9] where the design of a decoupled SMC and PID controller is conducted by using a GA to minimise the magnitude of the system's response, this was then tested on the inverted pendulum problem. The design of a fuzzy SMC, for the speed control of an induction machine, is considered in [10] where a GA is used to minimise the tracking error by manipulating the fuzzy logic membership functions.

To reduce a MOO problem into a SOO problem requires careful consideration of the relative weighting for each objective. This is non-trivial and requires an understanding of the system. The resulting weighted sum objectives also lose the physical meaning of its original individual objectives. Furthermore, solving a SOO only provides a single solution and therefore fails to give the designer a picture of the trade offs that occur between individual objectives. Instead in the following works a Pareto optimal design approach is used to produce a set of optimal controllers. This provides a more complete picture of the optimisation problem and can be used to prevent short sighted decision making. To design a controller for a Biped Robot, the work in [11] and [9] generate a Pareto front using the tracking error and control signal size as optimisation criteria: [11] uses a GA to design the fuzzy logic membership functions (for a fuzzy SMC) whilst [9] uses a Particle Swarm Optimisation (PSO) to design the sliding surface of a conventional SMC. Three different performance characteristics were optimised in [12] for an active suspension controller using PSO. A comparison of multiple MOO algorithms was conducted for this particular problem in [13] and argued that PSO was the most effective - therefore, PSO will form the basis of the optimisation used in this paper.

This paper presents a PSO method for the development of a fault tolerant sliding mode control allocation design where the underlying control law is based on the work presented in [4]. The novelty of this paper is in using an automated optimisation process to maximise the robustness of the controller against uncertainty in the fault reconstruction, whilst also minimising the tracking error and the size of the control signals. The proposed method provides an automatic tuning tool for a control designer, and provides a wider view of the multiple objectives allowing for better decision making.

II. Problem Statement

Consider the following system

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{1}$$

where $x(t) \in \mathbb{R}^n$ and $u(t) \in \mathbb{R}^m$ respectively denote the system's state and input. The effects of actuator faults and failures can be described by introducing the diagonal matrix $W = diag(w_1, ..., w_m)$ such that (1) becomes

$$\dot{x}(t) = Ax(t) + BWu(t) \tag{2}$$

College of Engineering, Mathematics and Physical Sciences, University of Exeter, Exeter, UK, EX4 4QF, lev201, h.alwi, c.edwards@exeter.ac.uk

where the individual components of W satisfy $0 \leq w_i \leq 1$ and represent the health of the i^{th} actuator. When $w_i = 0$ the i^{th} actuator has completely failed whereas when $w_i = 1$ the actuator is healthy - a value in between these two extremes represents an actuator fault. Due to the difficulty of monitoring actuator health, knowledge of W is typically imperfect. Commonly a Fault Detection and Isolation system (see for example [14] and [15]) is used to provide an approximation $\hat{W} \in \mathbb{R}^{m\times m}$ which is related to the true fault W through

$$W = (I - \Delta)\hat{W} \tag{3}$$

where $\Delta = diag(\delta_1, ..., \delta_m)$ represents the error in the fault estimation. Substituting (3) into (2) produces the following faulty uncertain system.

$$\dot{x}(t) = Ax(t) + B(I - \Delta)\hat{W}u(t) \tag{4}$$

The problem considered in this paper is one of designing a feedback control law for $u(t)$ such that the closed-loop system is optimally robust to the uncertainty Δ while ensuring small tracking errors and minimal control efforts.

III. Control Allocation

The controlled outputs of the system in (4) are given as

$$y(t) = Cx(t) \tag{5}$$

where $C \in \mathbb{R}^{l\times n}$. Assuming the system is over-actuated $m > l$ and a reduced order control can be defined. Firstly partition the B matrix from (4) to achieve the form

$$B = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} \tag{6}$$

where $B_1 \in \mathbb{R}^{(n-l)\times m}$ and $B_2 \in \mathbb{R}^{l\times m}$ where $\|B_1\|$ is 'small' in comparison to $\|B_2\|$ and so it can be said that the B_2 channels provide the majority of the control effort - this may require reordering the system's state. To simplify the following design procedure it is assumed that $\|B_2\| = 1$ (this is always achievable through scaling the last l states of the system, see [3]). Using the partitioned matrix (6), the signal $u(t)$ can be described in terms of a reduced order 'virtual' control signal through

$$u(t) = \hat{W}B_2^T(B_2\hat{W}^2B_2^T)^{-1}v(t) \tag{7}$$

Substituting (7) into system (4) yields the 'virtual' system

$$\dot{x}(t) = Ax(t) + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}(I - \Delta)B_2^+v(t) \tag{8}$$

where B_2^+ is the right pseudo-inverse of B_2 given by

$$B_2^+ = \hat{W}^2B_2^T(B_2\hat{W}^2B_2^T)^{-1}v(t) \tag{9}$$

In the case of perfect fault estimation, $W = \hat{W}$ and $\Delta = 0$, it can be verified that (4) reduces to

$$\dot{x}(t) = Ax(t) + \begin{bmatrix} B_1B_2^+ \\ I \end{bmatrix}v(t) \tag{10}$$

demonstrating that $v(t)$ is perfectly mapped to the bottom l states of the system when W is known.

IV. Sliding Mode Control Design

A. Sliding Surface Analysis

Define a coordinate transformation $x \mapsto \hat{x} = T_rx$ where

$$T_r = \begin{bmatrix} I & -B_1B_2^T \\ 0 & I \end{bmatrix} \tag{11}$$

In the new coordinates system equation (8) becomes

$$\dot{\hat{x}}(t) = \hat{A}\hat{x}(t) + \begin{bmatrix} B_1B_2^NB_2^+ \\ I \end{bmatrix}v(t) + \begin{bmatrix} B_1B_2^N\Delta B_2^+ \\ B_2\Delta B_2^+ \end{bmatrix}v(t) \tag{12}$$

where $\hat{A} = T_rAT_r^{-1}$ and

$$B_2^N = I - B_2^TB_2 \tag{13}$$

Under 'perfect' conditions, where $W = \hat{W} = I$ and $\Delta = 0$, the system in (12) reduces to the following regular form [1]:

$$\dot{\hat{x}}(t) = \hat{A}\hat{x}(t) + \underbrace{\begin{bmatrix} 0 \\ I \end{bmatrix}}_{\hat{B}_v}v(t) \tag{14}$$

Assuming that the matrix pair ($\hat{A}$,$\hat{B}_v$) is controllable, a control law for $v(t)$ can be constructed using sliding mode design principles [1]. Firstly define a switching function

$$s(t) = \begin{bmatrix} M & I \end{bmatrix}\begin{bmatrix} \hat{x}_1(t) \\ \hat{x}_2(t) \end{bmatrix} \tag{15}$$

where $\hat{x}_1(1) \in \mathbb{R}^{(n-l)}$ and $\hat{x}_2(t) \in \mathbb{R}^l$ are partitions of $\hat{x}(t)$ and $M \in \mathbb{R}^{l\times(n-l)}$ represents the design freedom (the process of choosing M will be discussed later on in the paper). Define a further change of coordinates as $\hat{x} \mapsto \tilde{x} = T_s\hat{x}$ where

$$T_s = \begin{bmatrix} I & 0 \\ M & I \end{bmatrix} \tag{16}$$

In the new coordinate system (12) becomes

$$\begin{bmatrix} \dot{\hat{x}}_1(t) \\ \dot{s}(t) \end{bmatrix} = \tilde{A}\begin{bmatrix} \hat{x}_1(t) \\ s(t) \end{bmatrix} + \begin{bmatrix} B_1B_2^N(I - \Delta)B_2^+ \\ I - B_2\Delta B_2^+ \end{bmatrix}v(t) \tag{17}$$

where

$$\tilde{A} = T_s\hat{A}T_s^{-1} = \begin{bmatrix} \tilde{A}_{11} & \tilde{A}_{12} \\ \tilde{A}_{21} & \tilde{A}_{22} \end{bmatrix} \tag{18}$$

and in particular

$$\tilde{A}_{11} = \hat{A}_{11} + \hat{A}_{12}M \tag{19}$$

It can be verified that if ($\hat{A}$, $\hat{B}_v$) is controllable then so is ($\hat{A}_{11}$, $\hat{A}_{12}$) and therefore it can be assumed that $\tilde{A}_{11}$ is stable through the design of M. During an ideal motion $s(t) = \dot{s}(t) = 0$, substituting these values into the bottom partition of (17) and rearranging for the 'virtual' control $v(t)$ yields the equivalent control needed to maintain sliding

$$v_{eq}(t) = -\Psi^{-1}(t)\tilde{A}_{21}\hat{x}_1(t) \tag{20}$$

where

$$\Psi(t) = I + MB_1B_2^N(I - \Delta)B_2^+ - B_2\Delta B_2^+ \tag{21}$$

Substituting $v_{eq}(t)$ for $v(t)$ in the top partition of (17) yields

$$\dot{\hat{x}}_1(t) = \big(\tilde{A}_{11} - B_1B_2^N(I - \Delta)B_2^+\Psi^{-1}(t)\tilde{A}_{21}\big)\hat{x}_1(t) \tag{22}$$

which describes the sliding motion of the top $n-l$ states of the system. This reduced order system can be further rewritten as

$$\left.\begin{aligned}\dot{\hat{x}}_1(t) &= \tilde{A}_{11}\hat{x}_1(t) - B_1 B_2^N \tilde{u}(t)\\ \tilde{y}(t) &= \tilde{A}_{21}\hat{x}_1\\ \tilde{u}(t) &= (I-\Delta)B_2^+\Psi^{-1}(t)\tilde{y}(t)\end{aligned}\right\} \tag{23}$$

where Ψ is defined in (21). The open-loop relationship between $\tilde{u}(t) \mapsto \tilde{y}(t)$ in (23) has the following transfer function

$$\tilde{G}(s) = \tilde{A}_{21}(sI - \tilde{A}_{11})^{-1}B_1 B_2^N \tag{24}$$

which is stable (since $\tilde{A}_{11}$ is stable by design) and therefore has a finite infinity norm which is defined as

$$\gamma_2 = \|\tilde{G}(s)\|_\infty \tag{25}$$

For the following stability analysis, define the following norms

$$\gamma_1 = \|MB_1B_2^N\|, \qquad \gamma_0 > \|B_2^+\| \tag{26}$$

where B_2^+ is defined in (9).

Remark: Due to the boundedness properties of pseudo-inverses [16], a finite value of γ_0 is guaranteed to exist for any $\hat{W} \in \mathcal{W}$ where $\mathcal{W}$ is the allowable fault set. In this paper $\mathcal{W}$ is chosen to ensure that for all $\hat{W} \in \mathcal{W}$, $\det(B_2\hat{W}B_2^T) \neq 0$. To find an appropriate value of γ_0, the entire search space of $\mathcal{W}$ is explored using GA.

Proposition 1: For any fault/failure combination $\hat{W} \in \mathcal{W}$ the sliding motion (22) is stable if the imprecision in the fault reconstruction satisfies

$$\|\Delta\| \leq \Delta_{max} \leq \frac{1-\gamma_0(\gamma_1+\gamma_2)}{\gamma_0(1+\gamma_1+\gamma_2)} \tag{27}$$

Proof The proof is similar to Proposition 1 in [4]. ■

B. Sliding Mode Control Laws

In this paper the virtual control is chosen to have the following structure

$$v(t) = \underbrace{-\tilde{A}_{21}\hat{x}_1(t) - \tilde{A}_{22}s(t)}_{v_l(t)} \underbrace{-\rho(t,x)\frac{s(t)}{\|s(t)\|}}_{v_n(t)} \tag{28}$$

where $v_l(t)$ and $v_n(t)$ respectively represent the linear and non-linear components of the control law.

Proposition 2: Choosing the scalar function $\rho(t,x)$ in (28) as any function that satisfies

$$\rho(t,x) \geq \frac{\gamma_0\gamma_1(1+\Delta_{max}) + \gamma_0\Delta_{max})\|v_l(t)\| + \eta}{1-\gamma_0\gamma_1(1+\Delta_{max}) - \gamma_0\Delta_{max}} \tag{29}$$

where η is a positive design scalar, guarantees that

$$s(t)^T\dot{s}(t) \leq -\eta\|s(t)\| \tag{30}$$

and therefore sliding is guaranteed to happen in a finite time and is maintained for all subsequent time.

Proof The proof is similar to Proposition 2 in [4]. ■

C. Practical Control Laws

To introduce reference tracking, this paper utilises an integral action method. This is achieved by augmenting the system in (8) with the integral action states

$$\dot{x}_r(t) = \underbrace{\bar{r}(t) - Cx(t)}_{e(t)} \tag{31}$$

such that $\bar{x}(t) = col(x_r(t), x(t))$. Here $\bar{r}(t)$ is a differentiable reference signal. The new augmented system is given by

$$\dot{\bar{x}}(t) = \bar{A}\bar{x}(t) + \bar{B}W(t)u(t) + B_r\bar{r}(t) \tag{32}$$

where the system matrices are

$$\bar{A} = \begin{bmatrix}0 & -C\\ 0 & A\end{bmatrix} \qquad \bar{B} = \begin{bmatrix}0\\ B\end{bmatrix} \qquad B_r = \begin{bmatrix}I\\ 0\end{bmatrix} \tag{33}$$

Through direct evaluation it can be proven that if (A, B) are controllable and (A, B, C) has no invariant zeros at, or near, the origin, the augmented system $(\bar{A}, \bar{B})$ is controllable [1]. The design of the control law (28) and (7) remain unchanged with the exception of including an additional term in the linear component of (28) such that

$$v_l(t) = -\tilde{A}_{21}\hat{x}_1(t) - \tilde{A}_{22}s(t) - MB_r\bar{r}(t) \tag{34}$$

To ensure a continuous control signal, and therefore avoid 'chattering' [1], a sigmoidal approximation is used for the non-linear component in (28) such that

$$v_n = \rho(t,x)\frac{s(t)}{\|s(t)\| + \delta} \tag{35}$$

where δ is a small positive design scalar.

The next section discusses the design problem - in particular how to select the design matrix M from equation (15).

V. Particle Swarm Optimisation

Consider the following generic MOO problem

$$\begin{aligned}min \quad & \vec{f}(\vec{q}) = [f_i(\vec{q}), ..., f_p(\vec{q})]^T\\ s.t. \quad & \vec{q} = \left[q_1, ..., q_r\right]^T \in \Omega\end{aligned} \tag{36}$$

where the search space Ω is defined as

$$\Omega = \{q_{i_{min}} \leq q_i \leq q_{i_{max}} \quad \text{for} \quad i = 1, ..., r\} \tag{37}$$

where $q_{i_{min}}$ and $q_{i_{max}}$ are bounds on the element q_i.

Definition: Consider two potential solutions to (36) denoted by $\vec{q}_1 \in \mathbb{R}^r$ and $\vec{q}_2 \in \mathbb{R}^r$; $\vec{q}_1$ is said to *dominate* $\vec{q}_2$ if $\forall i \in \{1, ..., p\}$, $f_i(\vec{q}_1) \leq f_i(\vec{q}_2) \quad \wedge \quad \exists i \in \{1, ..., p\} : f_i(\vec{q}_1) < f_i(\vec{q}_2)$ - this relationship is denoted by $\vec{q}_1 \preceq \vec{q}_2$. Conversely if $\exists i \in \{1, ..., p\} : f_i(\vec{q}_1) < f_i(\vec{q}_2) \quad \wedge \quad \exists i \in \{1, ..., p\} : f_i(\vec{q}_1) > f_i(\vec{q}_2)$ then $\vec{q}_1$ and $\vec{q}_2$ are said to be *mutually non-dominating* [17].

This section will present a Particle Swarm Optimisation (PSO) algorithm to find the set of mutually non-dominating solutions to the optimisation problem (36), this set is commonly known as the Pareto front. At the end of this section a method of utilising this algorithm to design a robust SMC, as described in the earlier sections, is discussed. To explicitly calculate a Pareto front for (36) is impractical since it would

require every value of $\vec{q} \in \Omega$ to be evaluated and then compared with each other. A more efficient method is to evaluate a random population of potential solutions (formed from a sub-set of $\vec{q} \in \Omega$) and generate an approximate Pareto front. An evolutionary algorithm can then be used to iteratively adjust the population such that the approximated Pareto front converges to the true Pareto front. The PSO algorithm achieves this through allowing each member of the population to travel around the search space Ω. The direction and speed of travel for each member is influenced by both the position of members on the current Pareto front, and the best position found by the individual member.

A. Algorithm

The algorithm presented here is based on the work presented in [18].

1) Assign values to the following variables:
   - Population size: $\mathbf{z} \in \mathbb{Z}^+$
   - Grid resolution: $\mathbf{h} \in \mathbb{Z}^+$
   - Max Member Velocity: ν_{max}
   - Maximum iterations: $\mathbf{i}_{max}$
2) Set $\mathbf{i} = 0$.
3) Randomly generate an initial population
$$\mathcal{P} = \{\vec{q}_{\mathbf{1}}, ..., \vec{q}_{\mathbf{z}}\}$$
where each member $\vec{q}_{\mathbf{n}} \in \Omega$.
4) Randomly assign each member of $\mathcal{P}$ a velocity $\vec{\nu}_{\mathbf{n}} \in \mathbb{R}^r$ where $\|\vec{\nu}_{\mathbf{n}}\| < \nu_{max}$.
5) Evaluate $f(\vec{q}_{\mathbf{n}})$ for every member in $\mathcal{P}$ and store the mutually non-dominating members into the repository $\mathcal{R}$.
6) Re-evaluate $\mathcal{R}$ and remove any dominated members[1].
7) Linearly divide the fitness space explored by the current repository $\mathcal{R}$ into $\mathbf{h}$ hypercubes of the same size. Assign weights to each hypercube based on the number of members that occupy it. If the $\mathbf{n}^{th}$ hypercube is empty then it is ignored, otherwise the weight can be calculated as
$$\sigma_{\mathbf{n}} = \frac{10}{Number \;\; of \;\; Members \;\; in \;\; Cube \;\; \mathbf{n}}$$
8) For each member $\vec{q}_{\mathbf{n}}$ in $\mathcal{P}$, update its best value $\vec{q}_{\mathbf{n}}^{best}$ according to:
   a) If $\vec{q}_{\mathbf{n}}^{best}$ is undefined then let $\vec{q}_{\mathbf{n}}^{best} = \vec{q}_{\mathbf{n}}$.
   b) If $\vec{q}_{\mathbf{n}} \preceq \vec{q}_{\mathbf{n}}^{best}$ then let $\vec{q}_{\mathbf{n}}^{best} = \vec{q}_{\mathbf{n}}$.
   c) Otherwise, do nothing.
9) Compute the new velocity of each member using the following formula:
$$\vec{\nu}_{\mathbf{n}} = c_1\vec{\nu}_{\mathbf{n}} + c_2(\vec{q}_{\mathbf{n}}^{best} - \vec{q}_{\mathbf{n}}) + c_3(\mathcal{R}_{\mathbf{n}} - \vec{q}_{\mathbf{n}}) \qquad (39)$$
where $\mathcal{R}_{\mathbf{g}}$ represents a random member of the repository. The index $\mathbf{g}$ is chosen by selecting a random member in a hypercube. The particular hypercube is selected through a Roulette Wheel search - using the values $\sigma_{\mathbf{n}}$ as a weighting[2]. The values c_1, c_2 and c_3 are positive design scalars. In the event that $\|\vec{\nu}_{\mathbf{n}}\| > \nu_{max}$ then the magnitude of the velocity vector is scaled so that $\|\vec{\nu}_{\mathbf{n}}\| = \nu_{max}$.
10) Update the position of each member in $\mathcal{P}$ through the formula:
$$\vec{q}_{\mathbf{n}} = \vec{q}_{\mathbf{n}} + \vec{\nu}_{\mathbf{n}}$$
If a members new position falls outside of Ω then set its new position is set to the point at which it left Ω, its current velocity is then set to $\vec{\nu}_{\mathbf{n}} = -\vec{\nu}_{\mathbf{n}}$.
11) If $\mathbf{i} = \mathbf{i}_{max}$, the maximum iterations has been completed, stop. Else set $\mathbf{i} = \mathbf{i} + 1$ and return to Step 5.

In step 1 the following should be considered when selecting parameters $\mathbf{z}$, $\mathbf{h}$ and ν_{max}. A larger population size $\mathbf{z}$ will lead to greater diversity in $\mathcal{P}$, this may be desirable to ensure that no search areas are overlooked, but this will increase the computational cost of the optimisation. Similarly a larger value of $\mathbf{h}$ improves the diversity, through directing members towards less populated areas, whilst also increasing the computational cost. The value of $\mathbf{h}$ should not be chosen large enough so that each member has its own hypercube, in this case the influence of population density when selecting $\mathcal{R}_g$ will be diminished. The value ν_{max} doesn't particularly affect the speed of each iteration, but it can effect the speed of convergence. A smaller value of ν_{max} will enable the members to exploit more of their local area, whilst a larger value will promote members to explore further afield. Too much exploitation of a members locality can lead to a poor convergence rate, whereas too much exploration can result in prematurely converging to a false Pareto front.

The parameter c_1 in (39) is commonly referred to as the member's inertia, whereas the parameters c_2 and c_3 are commonly referred to as confidence factors. As with physical systems, a greater inertia means that it is harder (takes a longer time) to change the members direction, this means that the member is influenced less by both their own best position and $\mathcal{R}_g$. The confidence factors c_2 and c_3 change the influence that the members own best position and $\mathcal{R}_g$ respectively have on its velocity. If $c_2 > c_3$ then more exploitation is encouraged whereas $c_2 < c_3$ leads to greater exploration. Commonly in the literature c_2 and c_3 are chosen so that $c_2 = c_3$ (see for example [13]).

B. Sliding Surface Design

Using the PSO optimisation discussed in the previous subsection it is possible to iterate upon controller designs and, by analysing simulation results, produce a set of optimal controllers. In this paper a controllers optimality is based on minimising control effort whilst maximising the tracking performance and robustness to uncertainty in the fault estimation. To design the sliding surface (and thus the controller

[1]Due to the computational cost, the set $\mathcal{R}$ should be restricted to a certain amount of members. The work in [18] provides a method of approaching this which aims to maintain population diversity.

[2]Note that the more populous a hypercube is, the smaller the weighting $\sigma_{\mathbf{n}}$ associated with it will be. Therefore, the more populous hypercubes have a smaller chance of being selected during the Roulette Wheel Selection. This promotes diversity in the population by encouraging exploration of less populated areas.

$$A = \begin{bmatrix} 0 & 0 & 1 & 0.074 \\ 0.048 & -0.023 & 0.087 & -1.004 \\ 0 & -1.806 & -2.803 & 0.504 \\ 0 & 0.384 & -0.171 & -0.046 \end{bmatrix} \qquad B = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.005 & -0.008 & -0.001 & -0.001 & 0.001 & 0.001 & 0.008 & 0.005 \\ 0.266 & -2.914 & -1.513 & -1.761 & 1.761 & 1.513 & 2.914 & 0.266 \\ -0.153 & 0.222 & 0.050 & 0.069 & -0.069 & -0.050 & -0.222 & -0.153 \end{bmatrix} \tag{38}$$

performance) the matrix M in (15) can be chosen through a quadratic minimisation process [1] in which the performance index is optimised subject to a symmetric positive weighting matrix $Q \in \mathbb{R}^{n \times n}$. Consider selecting

$$Q = diag(\vec{q}) \tag{40}$$

where $\vec{q}$ is defined in (36). For a given Q, solving an associated Riccatti equation specifies the hyperplane matrix M. To achieve the three objectives discussed above, consider the fitness function

$$\vec{f}(\vec{q}) = \begin{bmatrix} -f_\Delta(\vec{q}) & f_e(\vec{q}) & f_u(\vec{q}) \end{bmatrix}^T \tag{41}$$

where the individual components are defined as

$$f_\Delta(\vec{q}) = \Delta_{max} \tag{42}$$

$$f_e(\vec{q}) = \frac{1}{t_{max}} \sum_{i=1}^{l} \int_0^{t_{max}} \|e_i(t)\| dt \tag{43}$$

$$f_u(\vec{q}) = \frac{1}{t_{max}} \sum_{i=1}^{m} \int_0^{t_{max}} \|u_i(t)\| dt \tag{44}$$

where Δ_{max} is defined in (27), t_{max} denotes the maximum time used to analyse the system's response, $u(t)$ is defined by the sliding mode control allocation law (7) and (28) and $e(t)$ represents the tracking error as defined in (31). The expressions $e_i(t)$ and $u_i(t)$, in (43) and (44), denote the i^{th} component of the respective vectors. Minimising this cost function in (41) will maximise the tracking performance and the controller robustness whilst minimising the control effort used.

VI. A Design Example

A. Model

To demonstrate the effectiveness of the proposed PSO based tuning, the control of a non-linear Blended Wing Body's (BWB's) lateral dynamics is considered. Due to the lack of a traditional tailplane the BWB (shown in Figure 1) has both poor control authority and poor stability characteristics in the lateral axis - this makes the problem of designing an appropriate controller difficult. The lateral dynamics can be described by the following equations

$$\left.\begin{aligned} \dot{V} &= W_0 p - U_0 r + g \cos\theta_0 \sin\phi + C_Y(x,u)\bar{q}S/\bar{m} \\ \dot{\phi} &= p + r\cos\phi\tan\theta_0 \\ \dot{\psi} &= r\cos\phi\sec\theta_0 \\ \dot{p} &= C_M(x,u)b\bar{q}S/I_{xx} \\ \dot{r} &= C_L(x,u)b\bar{q}S/I_{zz} \end{aligned}\right\} \tag{45}$$

where V denotes the horizontal velocity (ms^{-1}), ϕ and ψ respectively denote the roll and yaw angles (rad) which have corresponding rates q and r ($rads^{-1}$). The values W_0, U_0 and θ_0 represent the vertical velocity (ms^{-1}), horizontal velocity (ms^{-1}) and the pitch angle (rad). Since only the lateral dynamics are considered here these values are considered to be fixed. The aerodynamic coefficients $C_i(x,u)$ are functions of the control input u and the system state x. These are derived from computational analysis using the Vortex Lattice Method (VLM) *TORNADO* described in [20]. The aircraft's weight (Kg) is given by $\bar{m}$, its wingspan (m) is given by b, and I_i denotes the moment of inertia (Kgm^2) around a pairing of body axes. The dynamic pressure (Pa) is denoted by $\bar{q}$. For more information on the aircraft's geometric parameters see [19].

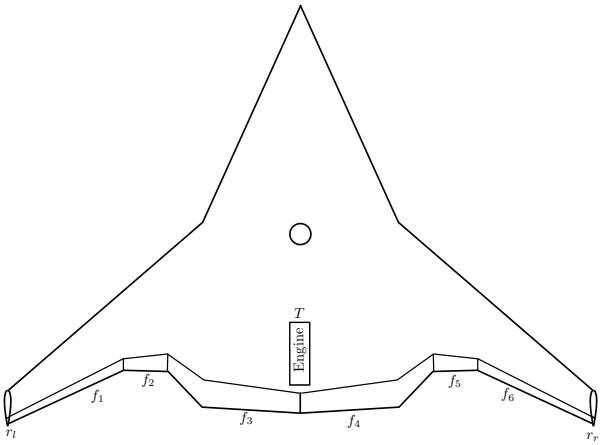


Fig. 1. Blended Wing Body aircraft diagram (adapted from [19])

Linearising the dynamics in (45) at a 'wings-level' flight condition at an airspeed of $200ms^{-1}$ and an altitude of $3000m$, a linear model of the following form can be obtained

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned} \tag{46}$$

where the A and B matrices are given in (38) and the output matrix is chosen as

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{47}$$

The system state $x(t)$ and the controlled outputs $y(t)$ are

$$x(t) = \begin{bmatrix} \phi & \beta & p & r \end{bmatrix}^T \tag{48}$$

$$y(t) = \begin{bmatrix} \phi & \beta \end{bmatrix}^T \tag{49}$$

where ϕ, p and r have been previously defined and β is the side-slip angle (rad). The system's input $u(t)$ is given by

$$u(t) = \begin{bmatrix} \delta r_l & \delta f_1 & \delta f_2 & \delta f_3 & \delta f_4 & \delta f_5 & \delta f_6 & \delta r_r \end{bmatrix}^T \tag{50}$$

where each element corresponds to perturbations (rad) of the matching rudder and elevon surfaces in Figure 1.

B. Optimisation Results

Using the tuning algorithm in §V, as each member of the population $\mathcal{P}$ is evaluated, a controller is created using the procedure described in §IV with the linearised model described in §VI-A. The design freedom M in (15) is chosen through quadratic minimisation with respect to the weighting matrix Q, which is defined as a function of $\vec{q} \in \Omega$ in (40). In this example $q_{i_{min}} = 1$ and $q_{i_{max}} = 75$ for all i since this is considered a region that would normally be manually explored. The controllers performance is analysed using the fitness functions in (42)-(44) by simulating the closed loop performance of the non-linear model. During the simulations the reference signal $r(t)$ is set to roll the aircraft to $20deg$ before returning back to 'wings-level' flight (see for example Figure 3a). For each control design the value of $\rho(t, x)$ and δ from (35) are set as 2 and 0.025 respectively. A value of γ_0 - defined in (26) - is calculated as 2.86 for the set of faults where at least one rudder (r_l or r_r) and one elevon (f_1-f_6) have an approximated health of 0.5. The population size **z** and the grid resolution **h** are chosen as 300 and 8000 respectively whilst the maximum velocity ν_{max} is chosen as 5. The weightings c_1, c_2 and c_3, used to evaluate a member's new velocity (39), are selected as 0.4, 2 and 2.

The PSO was run using the code adapted from [21] for a total of 500 iterations (i.e. $\mathbf{i}_{max} = 500$) - through experimentation this proved to be long enough to ensure convergence without using excessive computation. The resulting Pareto front is shown in Figure 2 from which 4 controllers are chosen, these are associated with the members $\vec{q}_\Delta$, $\vec{q}_e$, $\vec{q}_u$ and $\vec{q}_{opt}$. The members $\vec{q}_\Delta$, $\vec{q}_e$ and $\vec{q}_u$ are chosen to represent the optimal solutions to the respective fitness functions: $f_\Delta(\vec{q})$, $f_e(\vec{q})$ and $f_u(\vec{q})$. These represent the extreme points of the search space explored by the repository $\mathcal{R}$. The member $\vec{q}_{opt}$ is chosen through the Pareto $80/20$ principle and represents an optimal trade-off between all the objective functions [17].

The Pareto front in Figure 2 demonstrates that the PSO found a typical optimal control trade off between the functions $f_u(\vec{q})$ and $f_e(\vec{q})$. The Pareto front also shows no clear relationship between the robustness of the controller and its position within the optimal control curve - each controller has a value of $f_\Delta(\vec{q}) \in [0.2985 \quad 0.335]$. This has resulted in a Pareto front that closer resembles a 3D line (see bottom right of Figure 2) as opposed to a surface plot.

VII. Simulation Results

To demonstrate the range of controllers produced by the PSO, the simulation results associated with the optimal controllers $\vec{q}_\Delta$, $\vec{q}_e$, $\vec{q}_u$, and $\vec{q}_{opt}$ during the PSO process are presented in Figure 3. The controllers $\vec{q}_e$ and $\vec{q}_u$ are shown to be polar opposites: $\vec{q}_e$ offers the best tracking performance (at a cost to the size of the control signal) whilst $\vec{q}_u$ offers the smallest control signals (at a cost to the tracking performance). Looking at the Pareto front in Figure 2, this is to be expected since they are different ends of the optimal control curve. The optimal choice $\vec{q}_\Delta$ may offer the largest robustness ($\Delta_{max} = 0.335$) but has both poor reference tracking and large control signals - indicating

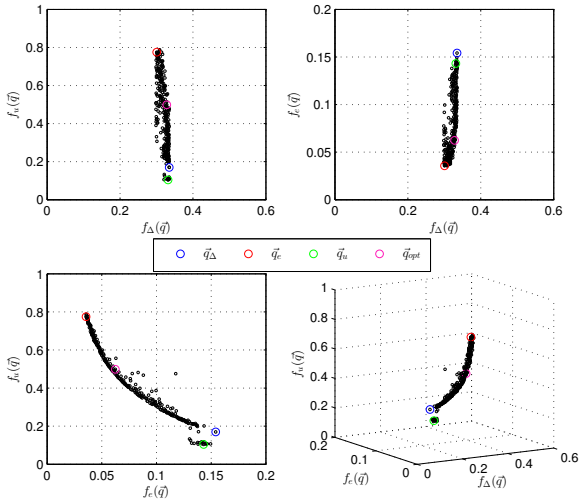


Fig. 2. Pareto Front produced by PSO

that choosing a controller just to be as robust as possible is inadvisable. The optimal trade-off controller $\vec{q}_{opt}$ is shown to have both good tracking performance (close to $\vec{q}_e$) and small control signal size (close to $\vec{q}_u$) whilst also having a robustness close to ($\vec{q}_\Delta$) - $\Delta_{max} = 0.328$ for $\vec{q}_{opt}$ compared to 0.335 for $\vec{q}_\Delta$.

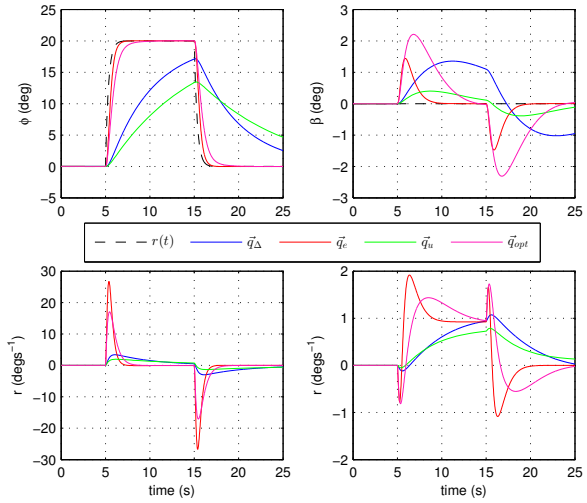
Figure 4 shows simulation results of $\vec{q}_{opt}$ undergoing the same manoeuvre used during the PSO whilst subject to a series of uncertain faults and failures. In this example $W = diag(1,\ 0.2,\ 0,\ 0.5,\ 1,\ 0,\ 1,\ 0)$ and the diagonal elements of Δ are chosen as a sinusoidal wave of magnitude 0.328 (Δ_{max} for $\vec{q}_{opt}$), with random frequencies and phase angles for each input channel. The state response in Figure 4a shows that the system's performance remains unchanged despite the faults and failures. In Figure 4b it can be seen that, during the fault and failure scenario, the failed actuators f_2, f_5 and r_r receive none of the control effort, whilst the faulty actuators f_1 and f_3 received less of the control effort when compared to the nominal fault free case. The healthy actuators r_l, f_4 and f_6 are shown to compensate for the loss of the control effort.

VIII. Conclusion

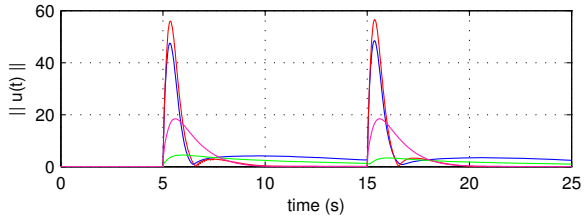
This paper proposed a method of designing a robust fault tolerant sliding mode controller through the use of a multiple objective particle swarm optimisation. The optimisation was shown to provide several different optimal controllers from which a designer could make an informed decision on. From the set of controllers an optimal trade-off was selected, this controller was shown to perform well in the presence of actuator faults and failures during simulation of a non-linear blended wing body aircraft.

References

[1] C. Edwards and S. Spurgeon, *Sliding mode control: theory and applications*. Taylor & Francis, 1998.

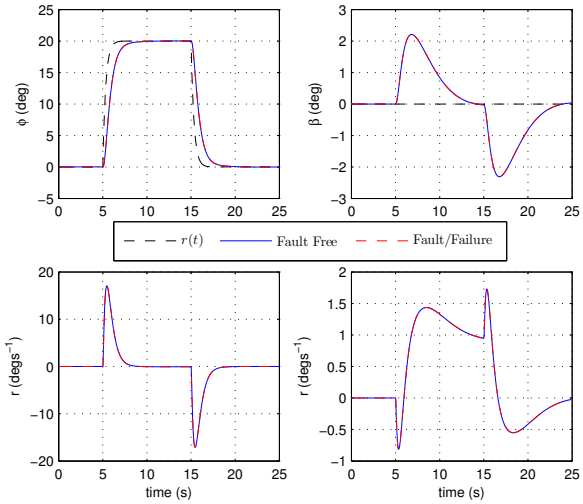


(a) States

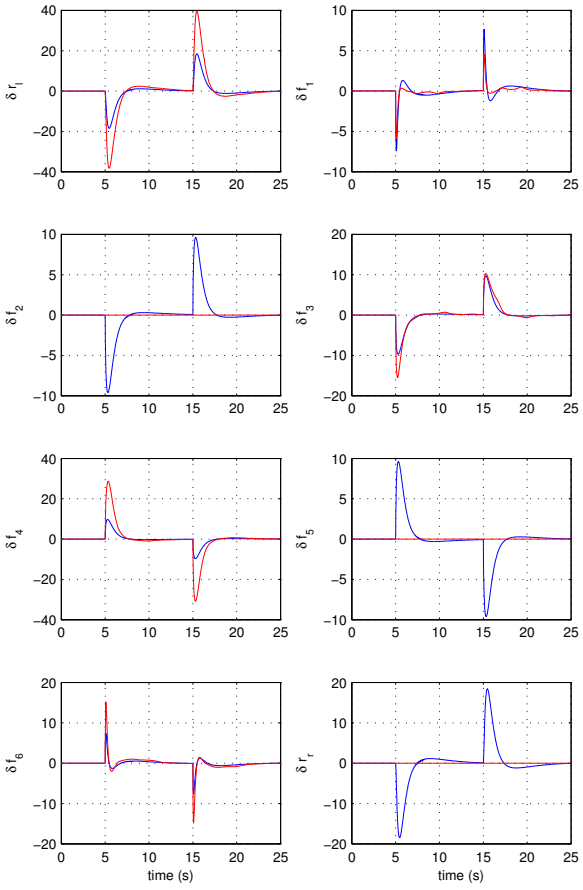


(b) Norm of Control Signal (deg)

Fig. 3. Comparison of Pareto Control Designs



(a) States



(b) Control Signals (deg)

Fig. 4. Optimal Control Design

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

[3] H. Alwi and C. Edwards, "Fault tolerant control using sliding modes with on-line control allocation," *Automatica*, vol. 44, no. 7, pp. 1859–1866, 2008.

[4] L. Vile, H. Alwi, and C. Edwards, "Fault Tolerant Control of a Blended Wing Body Aircraft using Priority Weighted Control Allocation and Sliding Modes," in *American Control Conference*, 2019.

[5] B. Wang and Y. Zhang, "An Adaptive Fault-Tolerant Sliding Mode Control Allocation Scheme for Multirotor Helicopter Subject to Simultaneous Actuator Faults," *IEEE Transactions on Industrial Electronics*, 2018.

[6] H. Alwi, M. T. Hamayun, and C. Edwards, "An integral sliding mode fault tolerant control scheme for an octorotor using fixed control allocation," in *13th International Workshop on Variable Structure Systems*, 2014.

[7] J. Wang and R. G. Longoria, "Coordinated and reconfigurable vehicle dynamics control," *IEEE Transactions on Control Systems Technology*, 2009.

[8] Y. Li, K. C. Ng, D. J. Murray-Smith, G. J. Gray, and K. C. Sharman, "Genetic algorithm automated approach to the design of sliding mode control systems," *International Journal of Control*, 1996.

[9] M. J. Mahmoodabadi, M. Taherkhorsandi, M. Talebipour, and K. Castillo-Villar, "Adaptive robust PID control subject to supervisory decoupled sliding mode control based upon genetic algorithm optimization," *Transactions of the Institute of Measurement and Control*, 2015.

[10] A. Hazzab, I. K. Bousserhane, and M. Kamli, "Design of a fuzzy sliding mode controller by genetic algorithms for induction machine speed control," *International Journal of Emerging Electric Power Systems*, 2004.

[11] B. Taran and A. Pirmohammadi, "Designing an optimal fuzzy sliding mode control for a two-link robot," *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, 2020.

[12] K. Gao, Z. Guo, Z. Qin, R. Li, and Y. Zhang, "Multi-objective optimal sliding mode control design of active suspension system with MOPSO algorithm," in *IOP Conference Series: Materials Science and Engineering*, 2019.

[13] M. J. Mahmoodabadi, M. Taherkhorsandi, and A. Bagheri, "Optimal robust sliding mode tracking control of a biped robot based on ingenious multi-objective PSO," *Neurocomputing*, 2014.

[14] H. Alwi and C. Edwards, "Sliding mode FTC with on-line control

allocation," in *Proceedings of the IEEE Conference on Decision and Control*, 2006.

[15] A. Cristofaro and T. A. Johansen, "Fault tolerant control allocation using unknown input observers," *Automatica*, vol. 50, no. 7, pp. 1891–1897, 2014.

[16] G. W. Stewart, "On scaled projections and pseudoinverses," *Linear Algebra and Its Applications*, vol. 112, no. C, pp. 189–193, 1989.

[17] U. Baumgartner, C. Magele, and W. Renhart, "Pareto optimality and particle swarm optimization," in *IEEE Transactions on Magnetics*, 2004.

[18] C. A. Coello, G. T. Pulido, and M. S. Lechuga, "Handling multiple objectives with particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, 2004.

[19] N. U. Rahman, "Propulsion and Flight Controls Integration for a Blended Wing Body Transport Aircraft," PhD, Cranfield University, 2010.

[20] T. Melin, "User's guide and reference manual for Tornado." Royal Institute of Technology (KTH), Department of aeronautics., Tech. Rep., 2012. [Online]. Available: http://tornado.redhammer.se/images/manual.pdf

[21] R. Martínez-Val, C. Cuerno, E. Pérez, and H. H. Ghigliazza, "Potential Effects of Blended Wing Bodies on the Air Transportation System," *Journal of Aircraft*, vol. 47, no. 5, pp. 1599–1604, 2010.