# Reconfigurable Direct Allocation for Multiple Actuator Failures

Asim Kumar Naskar, Sourav Patra, and Siddhartha Sen

***Abstract*— Flight control systems are often equipped with reconfigurable control allocation schemes to redistribute control commands in the event of actuator failures. In this brief, a reconfiguration scheme is proposed and is based on the direct allocation method. An iterative algorithm is developed and uses the same lookup tables prepared offline for normal operations, thereby saving a considerable amount of computation time and process memory. To elucidate the effectiveness of the method, the proposed allocation algorithm is applied to a satellite launch vehicle model. Simulation results show satisfactory performance of the method in the presence of multiple actuator failures.**

***Index Terms*— Actuator failures, direct allocation, fault tolerant control, overactuated systems, reconfiguration.**

## I. Introduction

STABILITY and reliability of air vehicles are vulnerable to actuator faults. Among different types of actuator faults one difficult situation arises when a faulty actuator fails to respond to control signals and additionally generates constant moments that acts as a disturbance. To overcome this situation, the control allocator should redistribute the command among the healthy actuators so as to cancel the effect of the disturbance and maintain the performance of the system as close as possible to the prefault situation [1]–[6]. Fixed point method and pseudoinverse method have been used for allocation and reconfiguration of actuators in [1], [3], and [4], while bisection method is adopted in [2] and [6] for the same. The reallocation technique proposed in [5] uses a set of stored matrix inverses to determine optimal actuator position. On the other hand, fault tolerance can also be achieved by reconfiguring controller [7]–[9]. Recently in [10], a sliding mode controller is proposed to reconfigure the controller command to nullify the disturbance moment generated by the faulty actuators. However, the usage of allocator instead of controller is advantageous to handle actuator faults [11]. For fault tolerant control, it is assumed that the fault is recoverable [12] and there exists a fault detection and isolation (FDI) unit. The FDI unit provides a technique for detection and isolation of failed components as early as possible. FDI unit makes a binary decision whether there is any fault or not and also determines the location and nature of the fault [13] based on which the allocation module reconfigures the actuator commands.

The problem of controller command distribution among actuators in an over-actuated system is known as control allocation. The first allocation method called direct allocation (DA) was proposed in [14] and later improved in [15]. In [16], the weighted pseudo inverse-based allocation method was proposed. Different optimization-based methods can be found in [17]–[20], but the majority of them are computationally intensive. The multiparametric quadratic programming based allocation uses off-line table and requires a small on-line computational effort in nonfaulty condition [19]. On-line computation is done through a binary search tree, where depth of the tree decides the number of computational steps. However, to accommodate actuator failure conditions, actuator constraints are treated as parameters that increases the dimension, and as a result the number of elements increases in the table that enhances the on-line computation effort. On the other hand, DA method provides a simple approach suitable for on-line implementation. In this method, all the moment vectors those are achievable within actuator position constraints are first generated off-line in the form of an attainable moment set (AMS) and stored in lookup tables [15]. When the desired moment vector is specified, the on-board computer finds, through simple calculations, the unique commands to be generated by each of the actuators using the lookup tables. However, in the event of a fault in one or more actuators, the shape of the AMS gets deformed and reconfiguration cannot be carried out without preparing fresh lookup tables for the postfault condition. Since obtaining a new AMS is computationally intensive, real-time fault reconfiguration is not possible using the existing technique. In this brief, an improvement to the DA method is proposed.

This brief proposes a reconfiguration technique to tackle faults in one or more actuators by modifying the lookup table based DA method. Here, it is assumed that the FDI unit works without data loss. With the details of the actuator faults obtained from the FDI unit, the proposed method tries to obtain iteratively an equivalent moment vector in the prefault AMS, which will correspond to the desired moment vector in the postfault AMS. In other words, it establishes a relationship between the prefault AMS and the postfault AMS. Therefore, reallocation of controls can be achieved without actually computing the postfault AMS. Using the original lookup tables, the reconfiguration scheme directly allocates the commands among the healthy actuators. Since it uses prefault lookup tables, the on-board computation is low; therefore, it is suitable for real-time applications. Exhaustive simulation studies on single and multiple actuator stuck-faults have been carried out and it has been seen that the proposed technique requires at most two/three iterations before converging to the equivalent moment vector.

## II. Statement of the Problem

To demonstrate the actuator command reallocation technique under different types of actuator failures, the basic feedback control structure of an over-actuated flight control

Manuscript received December 18, 2013; revised April 8, 2014; accepted April 13, 2014. Date of publication May 22, 2014; date of current version December 15, 2014. Manuscript received in final form April 18, 2014. Recommended by Associate Editor F. Caccavale.

The authors are with the Department of Electrical Engineering, IIT Kharagpur, Kharagpur 721302, India (e-mail: asim@ee.iitkgp.ernet.in; sourav@ee.iitkgp.ernet.in; ssen@ee.iitkgp.ernet.in).

Digital Object Identifier 10.1109/TCST.2014.2321665

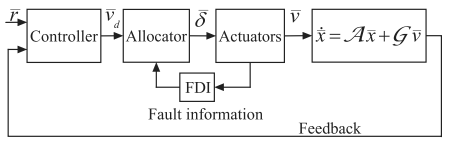


Fig. 1. Basic control block diagram of an air vehicle with allocator.

system is shown in Fig. 1. For fault tolerant control, the closed-loop control scheme contains a stabilizing controller, a reconfigurable allocation module and a FDI unit to provide fault information and isolation mechanism. In Fig. 1, let the time-invariant linearized state dynamics of an air vehicle be given as

$$\dot{\overline{x}} = \mathcal{A}\overline{x} + \mathcal{G}\overline{v} \tag{1}$$

$$\overline{v} = \mathcal{B}\overline{\delta} \tag{2}$$

where $\overline{x} \in \mathbb{R}^n$ is the state vector, $\overline{v} \in \mathbb{R}^3$ is the moment vector generated by $p$ number of actuators and the deflection of the actuators is governed by the actuator command $\overline{\delta} \in \mathbb{R}^p$ generated by the allocator. The command is constrained to limits $\delta_{i,\min} \le \delta_i \le \delta_{i,\max}$, for $i = 1, 2, \ldots, p$. It is worth mentioning that in this brief, the moment vector is assumed to be of dimension three (for aerospace applications they are pitch, yaw and roll) while $p > 3$, due to over-actuation in the control input. The system matrix, the equivalent input matrix and the control effectiveness matrix are denoted by $\mathcal{A} \in \mathbb{R}^{n\times n}$, $\mathcal{G} \in \mathbb{R}^{n\times 3}$ and $\mathcal{B} \in \mathbb{R}^{3\times p}$, respectively. It is also assumed that the actuator dynamics have infinite bandwidth. Referring to Fig. 1, $\overline{r}$ denotes a reference command generated by a pilot or an autopilot and $\overline{v}_d \in \mathbb{R}^3$ is the desired moment generated by the controller. When $\overline{v} = \overline{v}_d$, the actuators generate the desired moment perfectly.

Let $L$ denote a set of faulty actuators. These actuators generate constant moments, thereby termed as stuck-actuators. The set of stuck-actuators generate a disturbance vector, $\overline{d} = \sum_{i\in L} \overline{b}_i \delta_i^*$, where $\overline{b}_i$ is the $i$th column of $\mathcal{B}$ and $\delta_i^*$ is the stuck position of the $i$th actuator. Note that for a stuck-at-one fault in the $i$th actuator, $\delta_i^* = \delta_{i,\max}$ or $\delta_{i,\min}$; for a stuck-at-zero fault, $\delta_i^* = 0$. It is assumed that the rest of the actuators are nonfaulty and are operated by the actuator command $\overline{\delta}^L \in \mathbb{R}^p$, where $\delta_i^L = 0$, $i \in L$. In the case of multiple stuck-actuator failures, (2) can be expressed as

$$\overline{v_d} - \overline{d} = \mathcal{B}\overline{\delta}^L. \tag{3}$$

From (3), it is apparent that due to stuck-actuator faults, a disturbance vector appears in the closed-loop system. The objective of this brief is to develop a reconfiguration algorithm to reallocate the actuator command in the event of stuck-actuator faults to nullify the effect of this disturbance. This problem can be stated as follows.

In the presence of multiple stuck-actuator failures, find $\overline{\delta}^L$ such that $\mathcal{B}\overline{\delta}^L$ is as close as possible to $\overline{v}_d - \overline{d}$ in magnitude satisfying constraint limits $\delta_{i,\min} \le \delta_i^L \le \delta_{i,\max}$, where $i = 1, 2, \ldots, p$.

In this brief, reallocation of actuator commands in the event of stuck-actuator faults is performed using the prefault lookup tables instead of forming the new set of lookup tables.

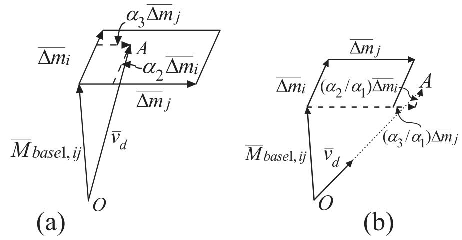


Fig. 2. Facet, facet$_{1,ij}$, in 3-D moment space where (a) $\overline{v}_d$ touches the facet and (b) extended $\overline{v}_d$ intersects the extended facet.

## III. Background: Lookup Table-Based DA Method

The direct allocation method using AMS for 3-D moment problems has been described in [15]. For a system with $p$ actuators, the actuator command is confined to the $p$-dimensional hyper rectangular volume, a convex set, generated by the command constraint limits. The AMS is obtained by a linear transformation of that volume through the control effectiveness matrix ($\mathcal{B} \in \mathbb{R}^{3\times p}$). Therefore, an AMS is a zonotope [21], i.e., the projection of a $p$-dimensional hyper rectangular volume on the 3-D space. A zonotope or an AMS can also be thought of as a Minkowski sum of line segments. In [22], an algorithm was developed to construct such sets using a sequence of Minkowski sum. In this brief, a relationship between the prefault and the postfault AMS is established for reconfiguration underpinning the concept of Minkowski sum. A prefault AMS is the Minkowski sum of postfault AMS and the maximum moment vectors that can be constructed with the help of faulty actuators information.

### A. AMS Formation

Depending upon the structure of the control effectiveness matrix, a system may be categorized into two types [15]. If any three columns of the control effectiveness matrix ($\mathcal{B} \in R^{3\times p}$, ($p > 3$)) are linearly dependent (in a plane), the system is called a coplanar system; otherwise, it is called a noncoplanar system. If a system is of latter type, its AMS has parallelogram plane facets on the boundary and every point on the boundary corresponds to a unique actuator command.

In a noncoplanar system, a pair of actuator commands decides the size of a pair of parallelogram facets, placed opposite to each other on the AMS, named as Max and Min facets in [15]. In this brief, for the $i$th and the $j$th actuator command pair, the Max and Min facets are denoted by facet$_{1,ij}$ and facet$_{2,ij}$, respectively.

Fig. 2(a) shows a facet, facet$_{1,ij}$, where two sides connected to a vertex are denoted by the vectors $\overline{\Delta m}_i \in \mathbb{R}^3$ and $\overline{\Delta m}_j \in \mathbb{R}^3$ and the vector from the origin $O$ to that vertex is denoted by $\overline{M}_{\text{base}1,ij} \in \mathbb{R}^3$. Any moment vector on the facet can be represented by these three independent vectors. For example, if $\overline{v}_d$ is a moment vector as shown in Fig. 2(a), then one can write $\overline{v}_d = \alpha_1 \overline{M}_{\text{base},1ij} + \alpha_2 \overline{\Delta m}_i + \alpha_3 \overline{\Delta m}_j$, or equivalently, it can be represented as

$$[\,\alpha_1 \quad \alpha_2 \quad \alpha_3\,]^T = \left[\,\overline{M}_{\text{base}1,ij} \quad \overline{\Delta m}_i \quad \overline{\Delta m}_j\,\right]^{-1} \overline{v}_d \tag{4}$$

where $\alpha_i \in \mathbb{R}$, $i = 1, 2, 3$. The above matrix is invertible as it comprises of three linearly independent column vectors. The values of $\alpha_i, i = 1, 2, 3$, obtained from (4), indicate whether the vector $\overline{v}_d$ intersects the facet, or it points to the facet or not. To identify the different cases, the following conditions are considered.

*Conditions:*

(*C1*) $\alpha_1 > 0$ and (*C2*) $0 < \alpha_2/\alpha_1,\ \alpha_3/\alpha_1 \leq 1$.

*Inferences:*

1) If (*C1*) and (*C2*) are satisfied: *(I1)* Extended $\overline{v}_d$ intersects the facet when $\alpha_1 < 1$. *(I2)* The tip of $\overline{v}_d$ touches the facet when $\alpha_1 = 1$. *(I3)* $\overline{v}_d$ intersects the facet when $\alpha_1 > 1$.
2) If (*C1*) is satisfied but not (*C2*): *(I4)* extended $\overline{v}_d$ intersects the extended facet when $\alpha_1 < 1$ [Fig. 2(b)]. *(I5)* The tip of $\overline{v}_d$ touches the extended facet when $\alpha_1 = 1$. *(I6)* $\overline{v}_d$ intersects the extended facet when $\alpha_1 > 1$.
3) If (*C1*) is not satisfied: *(I7)* $\overline{v}_d$ directs opposite to the facet in reference to the origin.

For a facet, $\text{facet}_{1,ij}$, the base vector $\overline{M}_{\text{base}1,ij}$ and the two corresponding side vectors, $\overline{\Delta m_i}$ and $\overline{\Delta m_j}$, are defined as follows:

$$\left.\begin{aligned} \overline{M}_{\text{base}1,ij} &= \overline{M}_{d1,ij} + \delta_{i,\min}\overline{b}_i + \delta_{j,\min}\overline{b}_j \\ \overline{\Delta m_i} = (\delta_{i,\max} - \delta_{i,\min})\overline{b}_i&,\ \overline{\Delta m_j} = (\delta_{j,\max} - \delta_{j,\min})\overline{b}_j \end{aligned}\right\} \quad (5)$$

where $\overline{M}_{d1,ij}$ is a positioning vector (defined later), $\overline{b}_i$ and $\overline{b}_j$ are, respectively, the $i$th and the $j$th columns of $\mathcal{B}$. For $\text{facet}_{2,ij}$, the side vectors $\overline{\Delta m_i}$ and $\overline{\Delta m_j}$ can similarly be defined as above, but the base vector becomes $\overline{M}_{\text{base}2,ij} = \overline{M}_{d2,ij} + \delta_{i,\min}\overline{b}_i + \delta_{j,\min}\overline{b}_j$. The base vector in (5) is defined following the same convention given in [15], however, one can also define it in a different way using $\delta_{i/j,\max}$ instead of $\delta_{i/j,\min}$; the definition for side vectors in (5) would have to be changed accordingly.

The positioning vectors $\overline{M}_{d1,ij}$ and $\overline{M}_{d2,ij}$ are defined as follows [15]:

$$\left.\begin{aligned} \overline{M}_{d1,ij} &= \sum_{k=1,k\neq i,j}^{p} \overline{\mu}_{k,\max} \\ \text{where} \quad \overline{\mu}_{k,\max} &= \begin{cases} \delta_{k,\max}\overline{b}_k & \text{if } \overline{b}_k^T\overline{\eta}_{ij} > 0 \\ \delta_{k,\min}\overline{b}_k & \text{if } \overline{b}_k^T\overline{\eta}_{ij} < 0 \end{cases} \end{aligned}\right\} \quad (6)$$

$$\left.\begin{aligned} \overline{M}_{d2,ij} &= \sum_{k=1,k\neq i,j}^{p} \overline{\mu}_{k,\min} \\ \text{where} \quad \overline{\mu}_{k,\min} &= \begin{cases} \delta_{k,\max}\overline{b}_k & \text{if } \overline{b}_k^T\overline{\eta}_{ij} < 0 \\ \delta_{k,\min}\overline{b}_k & \text{if } \overline{b}_k^T\overline{\eta}_{ij} > 0. \end{cases} \end{aligned}\right\} \quad (7)$$

In (6) and (7), $\overline{\eta}_{ij} = \overline{b}_i \times \overline{b}_j$ is the normal vector to $\text{facet}_{1,ij}$. From (6) and (7), it is apparent that $\overline{M}_{d1,ij}$ and $\overline{M}_{d2,ij}$ are directing opposite to each other and hence, the facets $\text{facet}_{1,ij}$ and $\text{facet}_{2,ij}$ are placed on the AMS opposite to each other. Equations (5)–(7) contain the complete set of information of the AMS. This set of data is stored in arrays, which are discussed next.

### *B. Offline Array Formation [15]*

Array-1 keeps the corresponding facet number (identifier) of points on the AMS where indices are the quantized values of azimuth and horizontal angles. Array-2, a 1-D array, stores the set of vectors given in (5) corresponding to each facet. Array-3 is also 1-D where each cell stores a $p \times 1$ column corresponding to each facet. The $p \times 1$ column is denoted by $\Delta_{1,ij}$ for a facet $\text{facet}_{1,ij}$. It stores the values $\delta_{k,\max}$ or $\delta_{k,\min}$ for $k = 1, 2, \ldots, p$, as indicated in (6). Similarly, the column $\Delta_{2,ij}$ is related to a facet, $\text{facet}_{2,ij}$. Among all the components of $\Delta_{1,ij}$ the $i$th and the $j$th components are unknown, which are called free (actuator) commands marked by suitable identifiers in the array. The known components in $\Delta_{1,ij}$ or in $\Delta_{2,ij}$ are the limiting values.

### *C. Moment Allocation*

An allocator generates an actuator command corresponding to a desired moment vector $\overline{v}_d$ demanded by a controller. Suppose $\overline{v}_d$ is the moment vector that points at $A$ on a facet on the AMS [Fig. 2(a)]. The point $A$ on the AMS boundary can be uniquely defined by the azimuth and the horizontal angles. Also, using these angles, the corresponding facet number can be picked up from Array 1. Suppose from the data stored in Array-1, the facet is $\text{facet}_{1,ij}$. Then, from Array-2, the base and two side vectors corresponding to $\text{facet}_{1,ij}$ are picked up. Using these vectors and (4) and (5), one can write: $\overline{\text{OA}} = \overline{v}_d/\alpha_1 = \overline{M}_{d1,ij} + (\delta_{i,\min} + \alpha_2/\alpha_1(\delta_{i,\max} - \delta_{i,\min}))\overline{b}_i + (\delta_{j,\min} + \alpha_3/\alpha_1(\delta_{j,\max} - \delta_{j,\min}))\overline{b}_j$. The $i$th and the $j$th free commands are given as follows:

$$\begin{aligned} i\text{th free command} &= \left(\delta_{i,\min} + \frac{\alpha_2}{\alpha_1}(\delta_{i,\max} - \delta_{i,\min})\right) \\ j\text{th free command} &= \left(\delta_{j,\min} + \frac{\alpha_3}{\alpha_1}(\delta_{j,\max} - \delta_{j,\min})\right). \end{aligned} \quad (8)$$

Therefore, the commands for the $i$th and $j$th actuators corresponding to $\overline{\text{OA}}$ are calculated using (8) and the rest of the commands are obtained from $\Delta_{1,ij}$ available in Array-3. Suppose the command vector corresponding to $\overline{\text{OA}}$ is $\overline{\delta}$. Then, the command vector corresponding to $\overline{v}_d$ is calculated as $|\overline{v}_d|/|\overline{\text{OA}}|\overline{\delta}$.

In case of faults in actuators, it is essential to follow a different algorithm to allocate actuator commands, which is the main focus of this brief.

## IV. Reconfiguration in the Event of Stuck-Actuator Failures

In the event of actuator faults, to avoid the construction of new arrays, a relationship between the pre-fault and post-fault AMSs is established using Eqs. (5)–(7). If a fault occurs at the $i$th actuator such that $\delta_{i,\max} = \delta_{i,\min} = 0$, $\overline{\Delta m_i} = 0$ and $\overline{M}_{d1,ij}$ remains unchanged. In this case, the $facet_{1,ij}$ is one dimensional, while for simultaneous faults at the $i$th and $j$th actuators $facet_{1,ij}$ becomes a point and $\overline{M}_{d1,ij}$ remains unchanged as it does not include the faulty actuators. But for $\{k, l\} \neq \{i, j\}$, all the lengths of $\overline{M}_{d1,kl}$ are reduced after the fault, and all the post fault facet, $facet_{1,kl}$ are placed in parallel with the pre-fault ones as they are formed by the same pair of $\overline{\Delta m_k}$, $\overline{\Delta m_l}$.

Let the set of faulty actuators be denoted by $L$. A facet, $\text{facet}_{1,lm}$, where $l$, $m$ indicate the free commands, is called

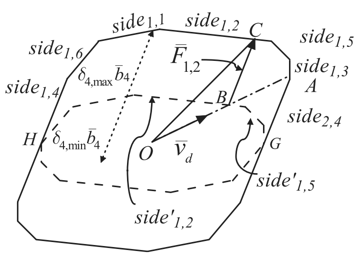


Fig. 3. Prefault (solid-line) and the postfault (dashed-line) AMS.

as faulty facet if either or both of the $l$ and $m$ are faulty. Thus, for a faulty facet, $\{L \cap \{l, m\}\} \neq \{0\}$. The prefault AMS will be called an outer AMS and the postfault AMS an inner AMS. After faults, corresponding to a nonfaulty facet $\text{facet}_{1,lm}$ on the outer AMS, a similar facet, denoted by $\text{facet}'_{1,lm}$, is obtained on the inner AMS. These two similar facets, $\text{facet}_{1,lm}$ and $\text{facet}'_{1,lm}$, are separated by a translational vector, which is defined as the fault vector, denoted by $\overline{F}_{1,lm}$. The translation direction from the inner AMS to the outer AMS is assumed to be positive.

### *A. 2-D Example*

To visualize the reconfiguration problem, a 2-D six actuator problem is considered, where the control effectiveness matrix is $\mathcal{B}$ and the corresponding actuator constraints are given in detail in Example 1 in Section V. It is assumed that the fault has occurred in the fourth actuator (i.e., $L = \{4\}$).

*1) Changes in AMS:* As shown in Fig. 3, due to fault in the fourth actuator, the prefault AMS polygon loses two $\text{side}_{2,4}$ and $\text{side}_{1,4}$ that collapse to the points $G$ and $H$, respectively; other nonfaulty sides form the inner AMS boundary via translation, shown by the dashed lines. In Fig. 3, $\text{side}'_{1,2}$ and $\text{side}_{1,2}$ are similar sides separated by the fault vector $\overline{F}_{1,2}$. The changes in AMS can also be understood in terms of the Minkowski sum. Suppose an inner AMS is given, as shown in Fig. 3. If the dotted vectors $\delta_{4,\max}\overline{b}_4$ and $\delta_{4,\min}\overline{b}_4$ are moved along the periphery of the dash-lined polygon, the outer AMS is obtained.

*2) Reconfiguration Task:* Let a moment vector $\overline{v}_d$ point at $A$ on $\text{side}_{1,3}$ of the outer AMS and $B$ on $\text{side}'_{1,2}$ of the inner AMS, as shown in Fig. 3. In a fault condition, the vector $\overline{v}_d$ can be reallocated if the information about point $B$ and the inner AMS is provided. From prefault arrays, the point $A$ can be obtained but there exists no direct relationship between the points $A$ and $B$. In this case, as shown in Fig. 3, it is worth noting that $\text{side}'_{1,2}$ is similar to $\text{side}_{1,2}$ and the corresponding translation vector is $\overline{F}_{1,2}$. Hence, there must exist a point $C$ on $\text{side}_{1,2}$ such that $\overline{\text{BC}} = \overline{F}_{1,2}$. The reconfiguration task here is to find the point $C$ on $\text{side}_{1,2}$, which is referred to as target facet. Note that due to translation along $-\overline{F}_{1,2}$, $\text{side}_{1,2}$ intersects $\overline{\text{OA}}$ at $B$ [Inference (*I3*)]. Therefore, the knowledge of the fault vector, the point $A$ and the prefault arrays indirectly give the information about the point $B$.

### *B. Reconfiguration for Multiple Stuck-at-Zero Faults*

The proposed reconfiguration method described below captures multiple stuck-at-zero actuator fault scenarios for noncoplanar systems. The discussion on the other types of stuck faults is deferred until Section IV-C. Let $L$ be the set of faulty actuators, and $l$, $m$ indicate the free commands associated with $\text{facet}_{1,lm}$. The facet is faulty if $\{L \cap \{l, m\}\} \neq \{0\}$.

Before describing the main algorithm, some definitions, involved in the reconfiguration method, are presented.

*Definition 1 (Normal Vector):* A vector, denoted by $\overline{\eta}_{1,lm}$, is said to be the normal vector corresponding to a facet, $\text{facet}_{1,lm}$ if the vector originates at $O$ and directs along the normal to the facet, $\text{facet}_{1,lm}$. ◇

*Definition 2 (Fault Vector):* For a facet, $\text{facet}_{1,lm}$, the fault vector denoted by $\overline{F}_{1,lm}$ is equal to $\mathcal{B}\overline{\delta}^{L_F}$, where $\overline{\delta}^{L_F} \in \mathbb{R}^p$ and

$$\overline{\delta}^{L_F} = \begin{cases} \delta_i{}^{L_F} = \left(\Delta_{1,lm}\right)_i & \text{for} \quad i \in L \setminus \{l,\ m\} \\ \delta_i{}^{L_F} = 0 & i \in \{1, 2, \ldots, p\} \setminus \{L \setminus \{l,\ m\}\} \end{cases}$$

where $\Delta_{1,lm}$ is a $p \times 1$ column containing the limiting values of $\delta_i$ for $i = 1, 2, \ldots, p$, used in (6). ◇

*Definition 3 (Similar Facet):* A facet, $\text{facet}'_{1,lm}$ represented by the following set of vectors:

$$\left.\begin{array}{l} \overline{M}'_{\text{base}1,lm} = \overline{M}_{\text{base}1,lm} - \overline{F}_{1,lm} \\ \overline{\Delta m}'_l = \overline{\Delta m}_l,\ \overline{\Delta m}'_m = \overline{\Delta m}_m \end{array}\right\} \tag{9}$$

is termed as the similar facet corresponding to a facet, $\text{facet}_{1,lm}$. ◇

As presented in (4), a vector $\overline{v}$ may be represented by the above set of vectors as shown below

$$[\,\alpha_1\ \alpha_2\ \alpha_3\,]^T = [\,\left(\overline{M}_{\text{base}1,lm} - \overline{F}_{1,lm}\right)\ \overline{\Delta m}_l\ \overline{\Delta m}_m\,]^{-1}\overline{v}. \tag{10}$$

For a 2-D case, (10) becomes

$$[\,\alpha_1\quad \alpha_2\,]^T = \left[\,\left(\overline{M}_{\text{base}1,l} - \overline{F}_{1,l}\right)\quad \overline{\Delta m}_l\,\right]^{-1}\overline{v}. \tag{11}$$

Note that the conditions and inferences presented in Section III are also valid for a similar facet for different values of $\alpha_i$, $i = 1, 2, 3$ given in (10). Moreover, the free actuator commands calculation using (8) is also used for a similar facet. The sign of these free actuator commands play an important role in determining the reconfiguration vector.

*Definition 4 (Reconfiguration Vector):* Let a similar facet, $\text{facet}'_{1,lm}$, intersect the desired vector or its extension at a point $B'$. The reconfiguration vector corresponding to $\text{facet}_{1,lm}$, denoted by $\overline{R}_{1,lm}$, is equal to $\mathcal{B}\overline{\delta}^{L_R}$. $\overline{\delta}^{L_R} \in \mathbb{R}^p$ is defined as

$$\overline{\delta}^{L_R} = \begin{cases} \delta_i^{L_R} = \left(\Delta_{1,lm}\right)_i & \text{for} \quad i \in L \setminus \{l,\ m\} \\ \delta_i^{L_R} = 0 & i \in \{1, 2, \ldots, p\} \setminus L \\ \delta_i^{L_R} = \delta_{i,\max} \text{ or } \delta_{i,\min} & \text{respectively, depending upon} \\ & \text{the positive or negative sign of the free} \\ & \text{actuator command, required to generate } \overline{\text{OB}'}, \\ & \text{(using (10) and (8)),} \quad i \in L \cap \{l,\ m\} \end{cases}$$

where $\Delta_{1,lm}$ is a $p \times 1$ column containing limiting values of $\delta_i$ for $i = 1, 2, \ldots, p$, used in (6). ◇

Note that when a facet is identified, the fault vector and the reconfiguration vector are obtained from the prefault arrays. If the facet is a nonfaulty type, i.e., $L \cap \{l,\ m\} = \{0\}$ and

$L \backslash \{l,\ m\} = L$, then $\overline{R}_{1,lm} = \overline{F}_{1,lm}$. Moreover, comparing the above definitions and (6), the nonzero components of $\overline{F}_{1,lm}$ are $\overline{\mu}_{k,max}$, where $k = i$ in Definitions 2 and 4, and thus from (6), $\overline{F}^T_{1,lm}\overline{\eta}_{1,lm} = \overline{R}^T_{1,lm}\overline{\eta}_{1,lm} > 0$. However, for a faulty facet, the nonzero components $\delta_i^{L_R}$, $i \in L \cap \{l,\ m\}$ make the fault vector different from the reconfiguration vector. In this case, $\overline{F}^T_{1,lm}\overline{\eta}_{1,lm} > 0$ and $\overline{R}^T_{1,lm}\overline{\eta}_{1,lm} > 0$ as $\overline{b}_i^T\overline{\eta}_{1,lm} = 0$ where $\overline{b}_i$ is the $i$th column of $\mathcal{B}$.

The main focus of this brief is to develop a reconfiguration method based on an iterative algorithm whose convergence is established in the following lemma.

*Lemma 1:* Let the desired moment vector $\overline{v}_d$ or its extension intersect the outer AMS and the inner AMS at points $A$ and $B$, respectively. Suppose the searching vector is initialized as $\overline{v}_m = \overline{\text{OA}}$, where $O$ is the origin and the following three steps are carried out.

1) Find the facet, where $\overline{v}_m$ points at/intersects the outer AMS.
2) Calculate the fault vector $\overline{F}$ and find the point $B'$ where the translated facet intersects $\overline{\text{OA}}$. The intersected vector is denoted by $\overline{\text{OB}'}$.
3) Calculate the reconfiguration vector $\overline{R}$ and $\overline{v}_m = \overline{\text{OB}'} + \overline{R}$.

If these three steps are repeated, a scalar defined as $\beta = |\overline{\text{OB}'}|/|\overline{\text{OA}}|$ will decrease monotonically to the value $\beta = |\overline{\text{OB}}|/|\overline{\text{OA}}|$.

*Proof:* To prove the Lemma, Definitions 1–4 and some geometrical facts, stated and proved in Appendix B are required. First applying facts 1 and 2 (or 3), $|\overline{\text{OB}'}| < |\overline{\text{OA}}|$. Next, applying 2 (or 3) and 4, it can be proved that the length of $|\overline{\text{OB}'}|$ decreases in each of the next iterations. Now, as $|\overline{\text{OA}}|$ is fixed, $\beta$ will decrease monotonically. The iteration ends when $\overline{\text{OB}'} = \overline{\text{OB}}$, because when $B$ is on a similar facet on the inner AMS, the resultant vector $\overline{\text{OB}} + \overline{R}$ points to a nonfaulty facet (on the outer AMS), which is similar to the facet containing $B$. This ends the proof. □

*Proposed Algorithm:* This algorithm is given for 3-D systems, and the type of actuator fault is considered as stuck-at-zero. Let $L$ be the set of faulty actuators and the desired moment $\overline{v}_d$ points at $A$ on the outer AMS boundary.

*Inputs to Algorithm:*

1. lookup tables (for prefault AMS): Arrays-1–3;
2. information of faulty actuators, i.e., the set $L$;
3. the vectors $\overline{\text{OA}}$ and $\overline{v}_d$.

*Solution Algorithm:*

*Step 1:* Initialize: $\overline{v}_m = \overline{\text{OA}}$; $k = 1$.

*Step 2:* From Array-1, find the facet where $\overline{v}_m$ is pointing at/to the outer AMS.

*Step 3:* Pick up the free commands from Array-3. Calculate the fault vector $\overline{F}$ as in Definition 2. From Array-2, get the base and side vectors and calculate $\alpha_1$, $\alpha_2$ and $\alpha_3$ using (10). Calculate the reconfiguration vector $\overline{R}$ as in Definition 4. Set $\overline{\text{OB}}_k = (1/\alpha_1)\overline{\text{OA}}$ and $\overline{v}_m = \overline{\text{OB}}_k + \overline{R}$.

*Step 4:* If the facet is faulty, or Condition ($C2$) is violated, set $k = k + 1$ and go to Step 2; otherwise, the obtained facet is the target facet.

*Step 5:* Calculate the free actuator commands using $\alpha_2/\alpha_1$, $\alpha_3/\alpha_1$ and (8); the rest of the commands are picked up from Array-3. Store the set of commands in a temporary vector $\overline{\delta}^T$ and insert zero to the positions corresponding to the faulty actuators. The modified vector is denoted by $\overline{\delta}^{Lo}$.

*Step 6:* If $|\overline{v}_d|/|\overline{\text{OA}}/\alpha_1| \le 1$, $\overline{\delta}^L = (|\overline{v}_d|/|\overline{\text{OA}}/\alpha_1|)\overline{\delta}^{Lo}$ or $\overline{\delta}^L = \delta^{L_0}$.

*Output From Algorithm:* Actuator command $\overline{\delta}^L$. ◇

*Remark 1:* Note that $B_k = B'$ and $1/\alpha_1 = \beta$, where $B'$ and $\beta$ are the intersection point and the scalar variable, respectively, as mentioned in Lemma 1.

*Remark 2:* The ratio $|\overline{v}_d|/|\overline{\text{OA}}/\alpha_1| \le 1$ ensures that the desired vector $\overline{v}_d$ is attainable. Note that $\overline{\text{OA}}/\alpha_1$ is the largest vector along the direction of $\overline{v}_d$ that may be allocated after the fault and the convergence of the algorithm is independent of the length of the desired vector.

*Remark 3:* In Step 5, $\overline{\delta}^T$ corresponds to a moment vector on the outer AMS. If this vector is denoted by $\overline{\text{OC}}$, from Step 4 it is apparent that the point $C$ must be lying on a nonfaulty facet. Translating point $C$ along $-\overline{F}$ direction, point $B$ is obtained on the inner AMS, i.e., $\overline{\text{BC}} = \overline{F} = \mathcal{B}\overline{\delta}^{L_F}$ (Definition 2). Since $\overline{\text{OB}} = \overline{\text{OC}} - \overline{F} = \mathcal{B}(\overline{\delta}^T - \overline{\delta}^{L_F}) = \mathcal{B}\overline{\delta}^{L_0}$, inserting zeros in $\overline{\delta}^T$ (in Step 5) provides the command vector corresponding to $\overline{\text{OB}}$. Note that $\mathcal{B}\overline{\delta}^{L_0} = \overline{\text{OB}} = \overline{\text{OB}_k} = \overline{\text{OA}}/\alpha_1$.

## C. Reconfiguration for Multiple Nonzero Stuck Faults

Due to nonzero stuck faults, disturbance moments generated by the faulty actuators can be calculated as $\overline{d} = \sum_{i\in L}\overline{b}_i\delta_i^*$, where $\overline{b}_i$ is the $i$th column of the control effectiveness matrix $\mathcal{B}$, $\delta_i^*$ is the nonzero stuck position of the $i$th actuator and $L$ is the set of faulty actuators. In this case, the modified desired moment vector $\overline{v}_T = \overline{v}_d - \overline{d}$. The reconfiguration task here is to allocate $\overline{v}_T$ in place of $\overline{v}_d$ among the healthy actuators considering the type of actuator fault as stuck-at-zero.

## D. Reconfiguration for Coplanar Systems

Coplanar systems may contain overlapping parallelogram facets [15], so Array-1 becomes 3-D. In Step 2 of the algorithm, an additional search in the third dimension of Array-1 is required to find a nonfaulty facet.

# V. Simulation Results

## A. Example 1

Consider the 2-D problem discussed in Section III-A. The control effectiveness matrix is

$$\mathcal{B} = \begin{bmatrix} 10 & 18 & 0 & 8 & 3 & 5 \\ 2 & -2 & -3 & 20 & -3 & 6 \end{bmatrix}.$$

This is a six-actuator noncolinear system (analogous to noncoplanar systems), i.e., no two columns are linearly dependent. For all actuators, $\delta_{\max} = 1$ and $\delta_{\min} = -1$. The AMS consists of 12 sides, as shown in Fig. 3. In 2-D case, one of the controls

TABLE I

ARRAY-1 FOR EXAMPLE 1

| | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Index | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| Side | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 9 | 11 | 11 | 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 |
| Index | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 |
| Side | 8 | 8 | 8 | 8 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 12 | 12 | 12 | 12 | 12 | 12 |
| Index | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 |
| Side | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 10 | 3 | 3 | 5 |
| Index | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 |
| Side | 5 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 |
| Index | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 100 |
| Side | 1 | 1 | 1 | 1 | 6 | 6 | 6 | 6 | 6 | 6 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 |

TABLE II

ARRAY-2 FOR EXAMPLE 1. $\overline{M}$ STANDS FOR $\overline{M}_{\text{base}}$

| Side | 1 | | 2 | | 3 | | 4 | | 5 | | 6 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Vectors | $\overline{M}$ | $\overline{\Delta m}$ | $\overline{M}$ | $\overline{\Delta m}$ | $\overline{M}$ | $\overline{\Delta m}$ | $\overline{M}$ | $\overline{\Delta m}$ | $\overline{M}$ | $\overline{\Delta m}$ | $\overline{M}$ | $\overline{\Delta m}$ |
| | -18 | 20 | 2 | 36 | 44 | 0 | -44 | 16 | 38 | 6 | -28 | 10 |
| | 32 | 4 | 36 | -4 | 26 | -6 | -20 | 40 | 32 | -6 | 20 | 12 |
| Side | 7 | | 8 | | 9 | | 10 | | 11 | | 12 | |
| Vectors | $\overline{M}$ | $\overline{\Delta m}$ | $\overline{M}$ | $\overline{\Delta m}$ | $\overline{M}$ | $\overline{\Delta m}$ | $\overline{M}$ | $\overline{\Delta m}$ | $\overline{M}$ | $\overline{\Delta m}$ | $\overline{M}$ | $\overline{\Delta m}$ |
| | -2 | 20 | -38 | 36 | -44 | 0 | 28 | 16 | -44 | 6 | 18 | 10 |
| | -36 | 4 | -32 | -4 | -20 | -6 | -20 | 40 | -26 | -6 | -32 | 12 |

TABLE III

ARRAY-3 FOR EXAMPLE 1. F CORRESPONDS TO FREE COMMAND

| Side | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $\Delta$ | F | 1 | 1 | -1 | 1 | -1 | F | -1 | -1 | 1 | -1 | 1 | Act-1 |
| | -1 | F | 1 | -1 | 1 | -1 | 1 | F | -1 | 1 | -1 | 1 | Act-2 |
| | -1 | -1 | F | -1 | -1 | -1 | 1 | 1 | F | 1 | 1 | 1 | Act-3 |
| | 1 | 1 | 1 | F | 1 | 1 | -1 | -1 | -1 | F | -1 | -1 | Act-4 |
| | -1 | -1 | 1 | -1 | F | -1 | 1 | 1 | -1 | 1 | F | 1 | Act-5 |
| | 1 | 1 | 1 | -1 | 1 | F | -1 | -1 | -1 | 1 | -1 | F | Act-6 |

within its limit forms a pair of sides on the boundary of an AMS (like Max and Min facet in 3-D case), and the rest of the controls decide the distance between them. A Max side on the outer AMS constructed by $i$th control is denoted by side$_{1,i}$, whereas the base and side vectors are denoted by $\overline{M}_{\text{base}1,i}$ and $\overline{\Delta m}_i$, respectively. Three arrays are given in Tables I–III. Here, stuck-at-zero type double actuator faults are considered. For this problem, Table I is a 1-D array that contains side numbers. The index in Table I is the quantized azimuth angle spanning from $-180°$ to $180°$. The span is divided into 100 equal segments and index 1 represents $-180°$. In the quantization process, fractions are rounded to the nearest integer value. Tables II and III are also 1-D arrays with 12 cells. Each cell in Table II contains a $2 \times 2$ matrix whose first column represents the base vector and the second column represents the side vector corresponding to a side of the outer AMS. On the other hand, a cell in Table III is a $6 \times 1$ column that contains the limiting values to generate positioning vectors corresponding to each side.

*1) Double-Actuator Fault:* Assume that stuck-at-zero type faults have occurred in Actuators-2 and 4, i.e., $L = \{2,\ 4\}$, and $\overline{v}_d = [2.5\ 1]^T = 2.69\angle 21.8°$ is the desired vector. The changes in AMS due to the faults and the steps followed for the reconfiguration of actuator commands are shown in Fig. 4. The ray along $\overline{v}_d$ meets the outer AMS at $A$ and $O$ is the origin. The vector $\overline{\text{OA}}$ is calculated using the method as explained in Section III-C.

*Iteration 1 (k = 1):*

*Step 1:* $\overline{\text{OA}} = [42.8595\ 17.1438]^T = 46.16\angle 21.8°$.

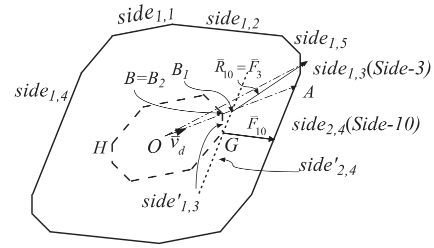


Fig. 4. AMS before (solid) and after (dashed) faults in Actuators-2 and 4.

*Step 2:* For $\angle 21.8°$, index $= 57$ and Side-10 is selected (Table I).

*Step 3:* From Table III, the fourth command is the free command (for Side-10). So $L \setminus \{4\} = \{2\}$, $\overline{\delta}^{L_F} = [0\ 1\ 0\ 0\ 0\ 0]^T$ and $F_{10} = \mathcal{B}\overline{\delta}^{L_F} = [18\ -2]^T$. From Table II, the base and side vectors (for Side-10) are $\overline{M}_{\text{base}} = [28\ -20]^T$ and $\overline{\Delta m} = [16\ 40]^T$, respectively. The vector $\overline{\text{OA}}$ is intersected by translated Side-10 at $B_1$ and the factors are $\alpha_1 = 2.09314$ and $\alpha_2 = 1.3705$ [using (11)]. The vector $\overline{\text{OB}}_1 = \overline{\text{OA}}/\alpha_1 = [20.476\ 8.1904]^T$. Now, Side-10 is faulty and the free command required to generate $\overline{\text{OB}}_1$ by translated Side-10 is 0.3094 [using (8)]. So $\overline{\delta}^{L_R} = [0\ 1\ 0\ 1\ 0\ 0]^T$, $\overline{R}_{10} = \mathcal{B}\overline{\delta}^{L_R} = [26\ 18]^T$ and $\overline{v}_m = \overline{\text{OB}}_1 + \overline{R}_{10} = [46.4761\ 26.1904]^T$.

*Step 4:* As $\{L \cap \{4\}\} = \{4\}$ (Side-10 faulty), $k = k + 1$ and the iteration is repeated.

*Iteration 2 (k = 2):*

*Step 2:* $\overline{v}_m = [46.4761\ 26.1904]^T = 53.34\angle 29.40°$, index $= 59$ (nearest integer) and Side-3 is selected (Table I).

*Step 3:* The third command is the free command (Table III). So $L \setminus \{3\} = \{2,\ 4\}$, $\overline{\delta}^{L_F} = [0\ 1\ 0\ 1\ 0\ 0]^T$, $\overline{F}_3 = \mathcal{B}\overline{\delta}^{L_F} = [26\ 18]^T$, $\overline{M}_{\text{base}} = [44\ 26]^T$, $\overline{\Delta m} = [0\ -6]^T$ (Table II), $\alpha_1 = 2.381$ and $\alpha_2 = 0.3174$ [using (11)].

*Step 4:* As $\{L \cap \{3\}\} = \{0\}$ and $\frac{\alpha_2}{\alpha_1} = 0.133$ satisfy Condition ($C2$), the iteration stops.

*Step 5:* The actuator commands are picked up from the column corresponding to Side-3 in Table III and using (8), the free actuator command is $-0.733$. All the commands are stored in $\overline{\delta}^T$ and zero is inserted in the second and fourth positions (Actuators-2 and 4 are faulty) of $\overline{\delta}^T$. The reconfigured command vector corresponding to $\overline{\text{OB}}$ is $\overline{\delta}^{L_0} = [1\ 0\ -0.733\ 0\ 1\ 1]^T$.

*Step 6:* The reallocated command corresponding to $\overline{v}_d$ is $\overline{\delta}^L = \alpha_1(|v_d|/|\overline{\text{OA}}|)\overline{\delta}^{L_0} = [0.139\ 0\ -0.102\ 0\ 0.139\ 0.139]$. The generated moment vector is $\overline{v} = \mathcal{B}\overline{\delta}^L = [2.502\ 1.001]^T \approx \overline{v}_d$, which signifies success of the reallocation algorithm.

### B. Example 2

A 3-D allocation problem of a satellite launch vehicle (SLV) [23] is considered. The launch vehicle has eight actuators to control pitch, roll and yaw moments. The control effectiveness matrix ($\mathcal{B}$) of the system and the actuator constraints ($\bar{\delta}_{\max/\min}$) are given in Appendix A. The launch vehicle is a coplanar system.

*1) Computational Efficiency of the Proposed Reconfiguration Method:* Under faulty condition, computation time of

TABLE IV

CONTROL REALLOCATION RESULTS FOR SLV SYSTEM: FEASIBLE SET

| Method | Avg. conv. Time (s) | Max. conv. Time (s) | Avg. error norm (deg/$s^2$) | Max. error norm (deg/$s^2$) |
|---|---|---|---|---|
| Matlab linear program | 0.0255 | 0.033 | 0 | 0 |
| Simplex | 0.00433 | 0.0054 | 0 | 0 |
| Proposed method | 0.00094 | 0.00102 | 0 | 0 |

TABLE V

CONTROL REALLOCATION RESULTS FOR SLV SYSTEM: INFEASIBLE SET

| Method | Avg. conv. Time (s) | Max. conv. Time (s) | Avg. error norm (deg/$s^2$) | Max. error norm (deg/$s^2$) |
|---|---|---|---|---|
| Matlab linear program | 0.0283 | 0.031 | 1.02248 | 1.0266 |
| Simplex | 0.0046 | 0.0079 | 1.02248 | 1.0266 |
| Proposed method | 0.00087 | 0.00104 | 1.02248 | 1.0266 |

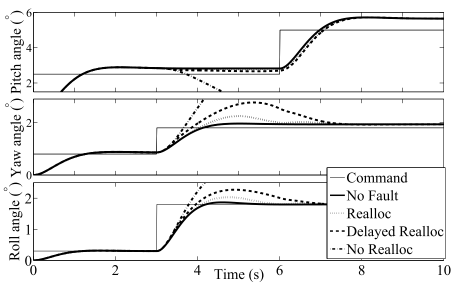


Fig. 5. System responses to step command under four different conditions: no fault, fault without reallocation, and fault reallocation with and without delay after stuck type faults.

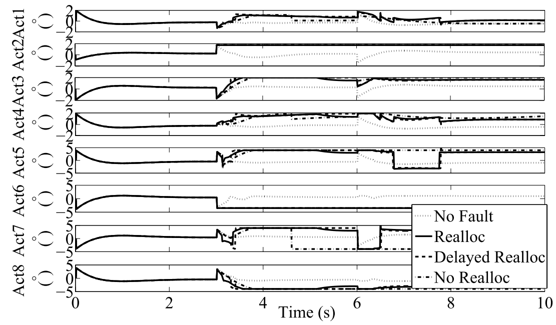


Fig. 6. Deflection of Actuators-1–8 under four different conditions: no fault, fault without reallocation, and fault reallocation with and without delay after stuck type faults.

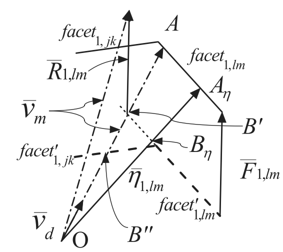


Fig. 7. Similar facets separated by the fault vector $\overline{F}_{1,lm}$.

the proposed approach has been compared with two different approaches of linear programming method—active set and simplex. MATLAB inbuilt function linprog is used for the active set method. The Blond rule for pivot selection is adopted in the Simplex method to prevent the cycling [17]. We assume Actuators-2 and 7 are faulty. The algorithms are implemented in MATLAB 12a and run on Pentium 4 platform.

First, we consider attainable moment vectors with values in the range $[\pm 4\ \pm 3\ \pm 2]^T \times 10^{-3}$°/s$^2$ and the results are summarized in Table IV. Then, we consider unattainable moment vectors with values in the range $[\pm 0.8\ \pm 0.3\ \pm 0.6]^{T^\circ}$/s$^2$, and the results are summarized in Table V. The comparison shows that the on-line computation time of the proposed method is less than the linear programming methods. Error norms remain same in each individual table since all the methods solve DA formulation.

*2) Closed-Loop Performance After Reconfiguration:* The postfault transient performance of the SLV system in closed loop is simulated with and without reconfiguration. The system model ($\mathcal{A}$, $\mathcal{G}$), and state feedback controller ($\mathcal{F}$) are given in Appendix A. At $t = 0$ s, a step command of $[2.5\ 0.8\ 0.3]^{T^\circ}$ is applied along the pitch, yaw and roll axes. Fig. 5 shows that the vehicle is steered close to the desired position.

Now, a step command of $[2.5\ 1.8\ 1.8]^{T^\circ}$ is applied at $t = 3$ s. Suppose, while actuating the command at $t = 3$ s, Actuators-2 and 6 are stuck at positions 1.74° and −3.45°, respectively. Fig. 5 shows that the system becomes unstable without reconfiguration. With the reallocation, the stability is restored and the healthy actuators can satisfy the prefault objectives. Also Fig. 5 shows that if the detection of fault is delayed by 0.3 s, the system can retain its stability after reallocation. However, due to the delay larger overshoots appear in the output response. In general, a modified controller design must be adopted to tackle the fault detection delay in a system. Furthermore, if another step command of $[2.5\ 0\ 0]^{T^\circ}$ is applied at $t = 6$ s, the system can track the command effectively. The actuator responses are shown in Fig. 6.

## VI. CONCLUSION

DA is a powerful tool for distributing control commands in an over-actuated system. The AMS for a system is prepared a priori and stored in lookup tables through which the allocation is carried out directly. In the event of actuator failures, the moment set gets distorted and typically new lookup tables are prepared for reallocation. In this brief, a new iterative algorithm has been proposed where the reallocation in the postfault scenario is performed using the original (prefault) lookup tables. The algorithm is suitable for multiple stuck-actuator failures. It assumes that the fault information is already available through a fault detection scheme. Moreover, the loss of data in an FDI unit is not taken into consideration. System stability and performance under such situation could be worth considering in future scope to exploit this brief for further extension. The efficacy of the method has been explained with reference to a simplified six-actuator 2-D case and also to an eight-actuator SLV control problem. The computational overhead is minimal and it has been observed that the algorithm normally requires two/three iterations before convergence and hence can effectively be used for online applications. The proposed method may also find ready application to other types of control allocation problems.

$$\mathcal{A} = \begin{bmatrix} 0 & 1000 & 0 & 0 & 0 & 0 \\ 706.6 & 0 & 0.0187 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1000 & 0 & 0 \\ 0.0271 & 0 & 437.9 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1000 \\ -0.5713 & 0 & 0.5468 & 0 & 0 & 0 \end{bmatrix} \times 10^{-3}$$

$$\mathcal{B} = \begin{bmatrix} 0.2985 & -0.2985 & -0.2985 & 0.2985 & 0.0779 & 0 & -0.0779 & 0 \\ -0.2985 & -0.2985 & 0.2985 & 0.2985 & 0 & 0.0779 & 0 & -0.0779 \\ -0.1618 & 0.1618 & -0.1618 & 0.1618 & 0 & 0.0211 & 0 & 0.0211 \end{bmatrix} \times \frac{\pi}{180}$$

$$\overline{\delta}_{\max} = \begin{bmatrix} 2 & 2 & 2 & 2 & 4 & 4 & 4 & 4 \end{bmatrix}^{T\circ} \text{ and } \overline{\delta}_{\min} = -\begin{bmatrix} 2 & 2 & 2 & 2 & 4 & 4 & 4 & 4 \end{bmatrix}^{T\circ}$$

$$\mathcal{G} = \begin{bmatrix} 0 & 0 & 0 \\ 5750.8 & -0.2223 & -1.4948 \\ 0 & 0 & 0 \\ 0.2207 & -5173 & 2.0768 \\ 0 & 0 & 0 \\ -4.6494 & 8.784 & 14153 \end{bmatrix} \times 10^{-3}$$

$$\mathcal{F} = \begin{bmatrix} 1069.3 & 620.84 & -0.0407 & -0.0272 & 0.0353 & 0.0560 \\ 0.0378 & 0.0257 & -1207.7 & -709.47 & 0.2344 & 0.1163 \\ 0.2591 & 0.1957 & 0.8037 & 0.4419 & 549.49 & 278.77 \end{bmatrix}$$

## APPENDIX A
## CLOSED-LOOP SYSTEM MATRICES

See equations at the top of the page.

## APPENDIX B
## FACTS

*Fact 1:* Suppose a ray along a moment vector $\overline{v}_d$ intersects a facet, facet$_{1,lm}$ at $A$, and the translated facet, facet$'_{1,lm}$ or its extension at $B'$. If $\overline{F}_{1,lm}^T\overline{\eta}_{1,lm} > 0$, where $-\overline{F}_{1,lm}$ is a translation vector by which facet$_{1,lm}$ is translated and $\overline{\eta}_{1,lm}$ is normal to facet$_{1,lm}$, then $|\overline{\mathrm{OB}'}| < |\overline{\mathrm{OA}}|$ where $O$ is the origin.

*Proof:* For nomenclatures, Fig. 7 is referred. Let $\overline{\eta}_{1,lm}$ intersects facet$_{1,lm}$ (or its extension) and facet$'_{1,lm}$ (or its extension) at $A_\eta$ and $B_\eta$, respectively. From the fact, $\overline{\mathrm{OA}}$ intersects those facets at $A$ and $B'$, respectively. Since $\overline{F}_{1,lm}^T\overline{\eta}_{1,lm} > 0$, the projection of $\overline{F}_{1,lm}$ on $\overline{\eta}_{1,lm}$, i.e., $\overline{B_\eta A_\eta}$ and $\overline{\mathrm{OA}_\eta}$ are pointing along the same direction. Therefore, $|\overline{\mathrm{OA}_\eta}| > |\overline{\mathrm{OB}_\eta}|$. Considering the line $AA_\eta$ parallel to the line $B'B_\eta$ in $\triangle AOA_\eta$, it is concluded that $|\overline{\mathrm{OA}}| > |\overline{\mathrm{OB}'}|$. □

*Fact 2:* Suppose a searching vector $\overline{v}_m$ points to a nonfaulty facet, facet$_{1,lm}$ and the extension of facet$'_{1,lm}$ intersects the ray along the desired moment vector $\overline{v}_d$ at $B'$. If the searching vector is reconfigured as $\overline{v}_m = \overline{\mathrm{OB}'} + \overline{R}_{1,lm}$, it must intersect a facet other than facet$_{1,lm}$ on the outer AMS.

*Proof:* As the vector $\overline{\mathrm{OB}'}$ touches the extension of facet$'_{1,lm}$, using (10) and from Inference (*I5*), we can write $\overline{\mathrm{OB}'} = (\overline{M}_{\mathrm{base}1,lm} - \overline{F}_{1,lm}) + \alpha_2\overline{\Delta m_l} + \alpha_3\overline{\Delta m_m}$. Since facet$_{1,lm}$ is a nonfaulty facet and $\overline{R}_{1,lm} = \overline{F}_{1,lm}$, we have $\overline{\mathrm{OB}'} + \overline{R}_{1,lm} = \overline{M}_{\mathrm{base}1,lm} + \alpha_2\overline{\Delta m_l} + \alpha_3\overline{\Delta m_m}$. Now from Inference (*I5*) $\alpha_2$ and/or $\alpha_3$ (here $\alpha_1 = 1$) do(es) not satisfy (*C2*). Therefore, the vector $\overline{\mathrm{OB}'} + \overline{R}_{1,lm}$ intersects the extension of facet$_{1,lm}$ and since the body is convex the vector must intersect another facet. □

*Fact 3:* Suppose a searching vector $\overline{v}_m$ points to a faulty facet, facet$_{1,lm}$, and facet$'_{1,lm}$ or its extension intersects the ray along the desired moment vector $\overline{v}_d$ at $B'$. If the searching vector is reconfigured as $\overline{v}_m = \overline{\mathrm{OB}'} + \overline{R}_{1,lm}$, it must intersect a facet other than the facet, facet$_{1,lm}$ on the outer AMS.

*Proof:* Along the same line of proof of Fact 2, we have $\overline{\mathrm{OB}'} = (\overline{M}_{\mathrm{base}1,lm} - \overline{F}_{1,lm}) + \alpha_2\overline{\Delta m_l} + \alpha_3\overline{\Delta m_m}$. Without loss of generality, let the $m$th actuator be faulty and a positive $m$th command be required to generate $\overline{\mathrm{OB}'}$. Adding and subtracting $\delta_{m,\min}\overline{b}_m$ and using (5) and (8), we have

$$\begin{aligned}\overline{\mathrm{OB}'} &= \overline{M}_{d1,lm} - \overline{F}_{1,lm} - \delta_{m,\min}\overline{b}_m \\ &\quad + \alpha_2\overline{\Delta m_l} + \underbrace{(\delta_{m,\min} + \alpha_3(\delta_{m,\max} - \delta_{m,\min}))}_{+\text{ve } m\text{-th command}}\overline{b}_m.\end{aligned}$$

Following the definitions of $\overline{R}_{1,lm}$ and $\overline{F}_{1,lm}$, their difference is caused by the set of commands belong to $\{L \cap \{l,\ m\}\}$, and $\overline{R}_{1,lm} - \overline{F}_{1,lm} = \delta_{m,\max}\overline{b}_m$ due to $+ve$ $m$th command. Now, we can write

$$\begin{aligned}&\overline{\mathrm{OB}'} + \overline{R}_{1,lm} = \overline{M}_{d1,lm} + \alpha_2\overline{\Delta m_l} \\ &+ \underbrace{\delta_{m,\max}\overline{b}_m}_{\overline{R}-\overline{F}} - \delta_{m,\min}\overline{b}_m + \underbrace{(\delta_{m,\min} + \alpha_3(\delta_{m,\max} - \delta_{m,\min}))}_{+\text{ve } m\text{-th command}}\overline{b}_m.\end{aligned}$$

Using $\overline{\Delta m_m} = (\delta_{m,\max} - \delta_{m,\min})\overline{b}_m$ and by rearranging, we have

$$\begin{aligned}\overline{\mathrm{OB}'} + \overline{R}_{1,lm} &= \overline{M}_{\mathrm{base}1,lm} + \overline{\Delta m_m} + \alpha_2\overline{\Delta m_l} \\ &\quad + \underbrace{(\delta_{m,\min} + \alpha_3(\delta_{m,\max} - \delta_{m,\min}))}_{+\text{ve } m\text{-th command}}\overline{b}_m.\end{aligned}$$

So $\overline{\mathrm{OB}'} + \overline{R}_{1,lm} = \overline{M}_{\mathrm{base}1,lm} + \gamma\,\overline{\Delta m_m} + \overline{\Delta m_m} + \alpha_2\overline{\Delta m_l}$ where the scalar $\gamma > 0$. The multiplication factor of $\overline{\Delta m_m}$, $(1+\gamma) > 1$.

So from (*I5*) the vector $\overline{\mathrm{OB}'} + \overline{R}_{1,lm}$ points at the extension of facet$_{1,lm}$, and since an AMS is convex the vector must intersect another facet. The proof for $-$ve $m$th command is similar. $\square$

*Fact 4:* Let the ray along the desired moment vector $\overline{v}_d$ be intersected by facet$'_{1,lm}$ or its extension at $B'$ and the vector $\overline{\mathrm{OB}'} + \overline{R}_{1,lm}$ intersects facet$_{1,jk}$ where $\{l, m\} \neq \{j, k\}$. If facet$'_{1,jk}$ or its extension intersects the ray along the desired moment vector at $B''$, then $|\overline{\mathrm{OB}}| \leq |\overline{\mathrm{OB}''}| < |\overline{\mathrm{OB}'}|$, where the point $B$ is the intersection point between the ray along the desired moment vector and the inner AMS.

*Proof:* Let a set of faulty actuators $L = \{1, 2, 3\}$. Commands $\{1, 2, 3\}$ forming the reconfiguration vector $\overline{R}_{1,lm}$ have all nonzero limiting values irrespective of whether facet$_{1,lm}$ is faulty or nonfaulty. Let $\overline{R}_{1,lm} = \overline{R}'_{1,lm} + \overline{R}''_{1,lm}$ and $\overline{F}_{1,jk} = \overline{F}'_{1,jk} + \overline{F}''_{1,jk}$. Without loss of generality, it is also assumed that $\overline{R}'_{1,lm}$ and $\overline{F}'_{1,jk}$ involve the 1st and 2nd actuator commands with same limiting values, i.e., $\overline{R}'_{1,lm} = \overline{F}'_{1,jk}$. Also, assuming the third actuator command involves in forming $\overline{R}''_{1,lm}$ and $\overline{F}''_{1,jk}$, which are different. If facet$_{1,jk}$ is nonfaulty, then $\overline{R}''_{1,lm} = -\overline{F}''_{1,jk}$. If facet$_{1,jk}$ is faulty, $\overline{F}''_{1,jk} = 0$, because only three actuators are considered faulty and out of them, two having nonzero limiting value, i.e., 1 and 2 are not free commands (corresponding to facet$_{1,jk}$). Therefore, the command 3 must be a free command which is involved in $\overline{F}''_{1,jk}$. As $\overline{\mathrm{OB}'} + \overline{R}_{1,lm}$ intersects facet$_{1,jk}$, we can write $\overline{\mathrm{OB}'} + \overline{R}_{1,lm} = \alpha_1 \overline{M}_{\mathrm{base}1,jk} + \alpha_2 \overline{\Delta m}_j + \alpha_3 \overline{\Delta m}_k$, where $\alpha_1 > 1$ and $0 \leq \alpha_2/\alpha_1, \alpha_3/\alpha_1 \leq 1$ [see (*I3*)]. Rearranging we have, $1/\alpha_1 \overline{\mathrm{OB}'} = \overline{v}_\alpha = (\overline{M}_{\mathrm{base}1,jk} - 1/\alpha_1 \overline{R}_{1,lm}) + \alpha_2/\alpha_1 \overline{\Delta m}_j + \alpha_3/\alpha_1 \overline{\Delta m}_k$. So $|\overline{\mathrm{OB}'}| > |\overline{v}_\alpha|$. One may consider $\overline{v}_\alpha$ intersects a fictitious facet facet$''_{1,jk}$ with the base vector $(\overline{M}_{\mathrm{base}1,jk} - 1/\alpha_1 \overline{R}_{1,lm})$ and the side vectors $\overline{\Delta m}_j$, $\overline{\Delta m}_k$. Now, it will be proved $|\overline{\mathrm{OB}''}| < \overline{v}_\alpha$ in cases of whether facet$_{1,jk}$ is faulty or nonfaulty. If facet$_{1,jk}$ is nonfaulty, facet$'_{1,jk}$ is obtained translating facet$_{1,jk}$ by $-\overline{F}_{1,jk}$. However, the similar-facet facet$'_{1,jk}$ can also be formed translating facet$''_{1,jk}$ by $(-(\alpha_1 - 1)/\alpha_1 \overline{F}'_{1,jk} - (1 + \alpha_1)/\alpha_1 F''_{1,jk})$. As $\alpha_1 > 1$, $\overline{F'}^T_{1,jk} \overline{\eta}_{1,jk} > 0$ and $\overline{F''}^T_{1,jk} \overline{\eta}_{1,jk} > 0$, applying the Fact 1, we have $|\overline{\mathrm{OB}''}| < |\overline{v}_\alpha|$. Therefore, $|\overline{\mathrm{OB}''}| < |\overline{\mathrm{OB}'}|$. Now if facet$_{1,jk}$ is *faulty*, facet$''_{1,jk}$ is translated by $(-(\alpha_1 - 1)/\alpha_1 \overline{F}'_{1,jk} + 1/\alpha_1 R''_{1,lm})$ to form facet$'_{1,lm}$. Via assumption, $R''_{1,lm}$ is involved to a free command corresponding to facet$_{1,jk}$, so $\overline{R''}^T_{1,lm} \overline{\eta}_{1,jk} = 0$ and $\overline{F'}^T_{1,jk} \overline{\eta}_{1,jk} > 0$. Applying Fact 1 we have $|\overline{\mathrm{OB}''}| < |\overline{\mathrm{OB}'}|$. Now $|\overline{\mathrm{OB}}| \leq |\overline{\mathrm{OB}''}|$ can be shown as follow. Since an AMS is a convex body, from the geometry of AMS it can be shown that the extension of inner facets (except those containing $B$ ) would intersect the ray along the desired vector above the point $B$ or would never intersect the ray. $\square$

## REFERENCES


[1] J. J. Burken, P. Lu, Z. Wu, and C. Bahm, "Two reconfigurable flight-control design methods: Robust servomechanism and control allocation," *J. Guid., Control, Dyn.*, vol. 24, no. 3, pp. 482–493, 2001.

[2] W. C. Durham, "Computationally efficient control allocation," *J. Guid., Control, Dyn.*, vol. 24, no. 3, pp. 519–524, 2001.

[3] Y. Zhang, V. S. Suresh, B. Jiang, and D. Theilliol, "Reconfigurable control allocation against aircraft control effector failures," in *Proc. Int. Conf. Control Appl.*, Singapore, Oct. 2007, pp. 1197–1202.

[4] S. Ye, Y. Zhang, X. Wang, and C. A. Rabbath, "Robust fault-tolerant control using on-line control re-allocation with application to aircraft," in *Proc. Amer. Control Conf.*, St. Louis, MO, USA, Jun. 2009, pp. 5534–5539.

[5] Y. Zhong, L. Yang, and G. Shen, "Control allocation based reconfigurable flight control for aircraft with multiple control effectors," in *Proc. 47th AIAA Aerosp. Sci. Meeting*, Jan. 2009, no. 2009-58.

[6] M. Demenkov, "Reconfigurable direct control allocation for overactuated systems," in *Proc. IFAC*, vol. 18. 2011, pp. 4696–4700.

[7] W. D. Morse and K. A. Ossman, "Model following reconfigurable flight control system for the AFTI/F-16," *J. Guid., Control, Dyn.*, vol. 13, no. 6, pp. 969–976, 1990.

[8] W. Chen and J. Jiang, "Fault-tolerant control against stuck actuator faults," *IEE Proc. Control Theory Appl.*, vol. 152, no. 2, pp. 138–146, Mar. 2005.

[9] R. K. Das, S. Sen, and S. Dasgupta, "Robust and fault tolerant controller for attitude control of a satellite launch vehicle," *IET Control Theory Appl.*, vol. 1, no. 1, pp. 304–312, Jan. 2007.

[10] 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.

[11] O. Harkegard and S. T. Glad, "Resolving actuator redundancy—Optimal control vs. control allocation," *Automatica*, vol. 41, no. 1, pp. 137–144, 2005.

[12] M. Staroswiecki, "On fault handling in control systems," *Int. J. Control, Autom., Syst.*, vol. 6, no. 3, pp. 296–305, 2008.

[13] I. Hwang, S. Kim, Y. Kim, and C. E. Seah, "A survey of fault detection, isolation, and reconfiguration methods," *IEEE Trans. Control Syst. Technol.*, vol. 18, no. 3, pp. 636–653, May 2010.

[14] W. C. Durham, "Constrained control allocation," *J. Guid., Control, Dyn.*, vol. 16, no. 4, pp. 717–725, 1993.

[15] J. A. Petersen and M. Bodson, "Fast implementation of direct allocation with extension to coplanar controls," *J. Guid., Control, Dyn.*, vol. 25, no. 3, pp. 464–473, 2002.

[16] J. D. Boskovic and R. K. Mehra, "Control allocation in overactuated aircraft under position and rate limiting," in *Proc. Amer. Control Conf.*, vol. 1. 2002, pp. 791–796.

[17] M. Bodson, "Evaluation of optimization methods for control allocation," *J. Guid., Control, Dyn.*, vol. 25, no. 4, pp. 703–711, 2002.

[18] O. Harkegard, "Efficient active set algorithms for solving constrained least squares problems in aircraft control allocation," in *Proc. 41st IEEE Conf. Decision Control*, Las Vegas, NV, USA, Dec. 2002, pp. 1295–1300.

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

[20] L. Cui and Y. Yang, "Disturbance rejection and robust least-squares control allocation in flight control system," *J. Guid., Control, Dyn.*, vol. 34, no. 6, pp. 1632–1643, 2011.

[21] G. Ziegler, *Lectures on Polytopes* (Graduate Texts in Mathematics), vol. 152. New York, NY, USA: Springer-Verlag, 1995.

[22] J. B. Lasserre, "Reachable, controllable sets and stabilizing control of constrained linear systems," *Automatica*, vol. 29, no. 2, pp. 531–536, 1993.

[23] W. C. A. Kishore, S. Dasgupta, G. Ray, and S. Sen, "Control allocation for an over-actuated satellite launch vehicle," *Aerosp. Sci. Technol.*, vol. 28, no. 1, pp. 56–71, 2013.