4

Application of Evolutionary Computing in Control Allocation

Hammad Ahmad, Trevor Young, Daniel Toal and Edin Omerdic
University of Limerick,
Republic of Ireland

1. Introduction

This issue of interaction of control allocation and actuator dynamics and has been dealt with by very few researchers. What was not considered in most control allocation algorithms is the fact that the control surfaces are manipulated by either hydraulics or electric actuators, and constitute a dynamic system which cannot produce infinite accelerations. In other words, if a control was initially at rest, and later commanded to move at its maximum rate in some direction for a specified amount of time, it would gradually build up speed until it reached the commanded rate. The final position of the control would therefore not be the same as that calculated using the commanded rate and the time during which it was instructed to move (Bolling 1997). In this chapter, a method, which post-processes the output of a control allocation algorithm, is developed to compensate for actuator dynamics. The method developed is solved for a diagonal matrix of gain corresponding to individual actuators. This matrix is then multiplied with the commanded change in control effector settings as computed by the control allocator and actuators dynamics interactions. The basic premise of this method is to post process the output of the control allocation algorithm to overdrive the actuators so that at the end of a sampling interval the actual actuator positions are equivalent to the desired actuator positions (Oppenheimer and Doman 2004). The overdriving of the actuators is done by multiplying the change in commanded signal with the identified gain matrix which is called the compensator. This identification is done by using a soft computing technique (i.e. genetic algorithms). The simulation setup including control allocator block, compensator and actuator rig makes a non-linear set up. During the identification of the compensator using this setup by soft computing technique such as genetic algorithms, the likelihood of the solution being a global minimum is high as compared to other optimisation techniques. This is why genetic algorithms have been used in this analysis rather than other techniques such as linear programming. The main contribution is to design a compensator using an evolutionary computing technique (i.e. genetic algorithms) to compensate the interaction between control allocation and actuator dynamics. It should be mentioned that in this method the model of the actuator does not need to be known. The simulation setup consists of excitation signals, the control allocation block, the compensator and the actuators rig.

When designing control allocation typically the actuator dynamics are ignored because the bandwidth of the actuators is larger than the frequencies of the rigid body modes of the aircraft. Fig. 1 shows a control allocator with actuator dynamics neglected. If there is a case in which actuator frequencies are comparable with the bandwidth of the rigid body modes then the actuator dynamics cannot be neglected, as shown in Fig. 2.

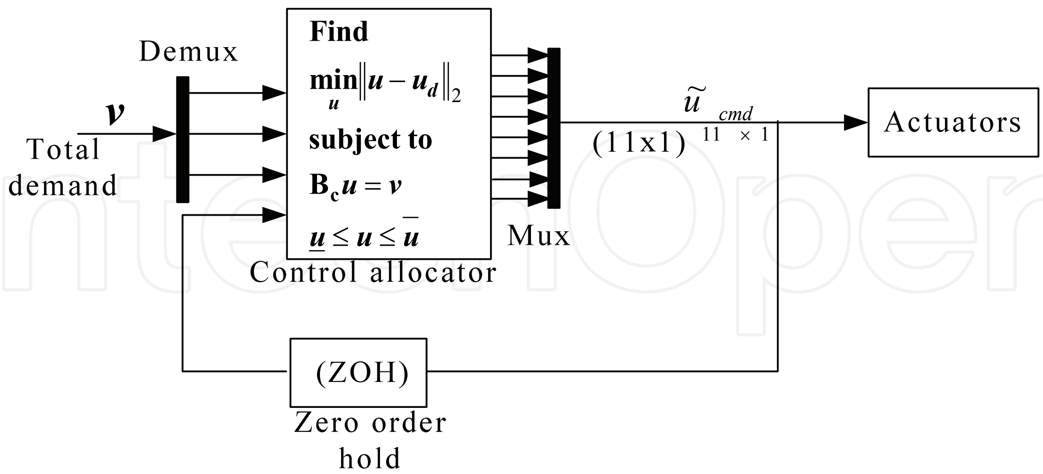

Fig. 1. Control allocation with actuator dynamics neglected

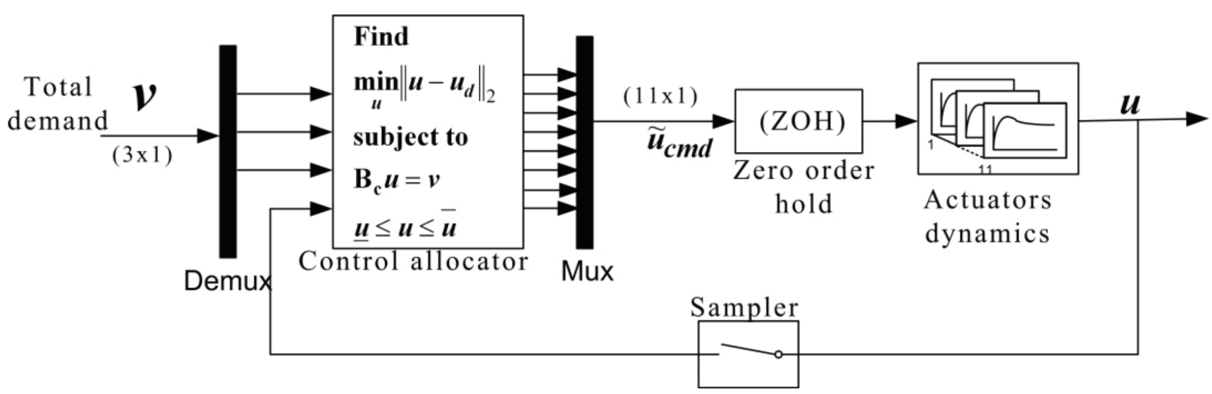

Fig. 2. Control allocation with actuator dynamics

In this case the output of the control allocator should match the output of the actuator dynamics. In reality the output of the control allocation is attenuated due to the presence of non-negligible actuator dynamics. The loss of the gain from the CA output signal is compensated by the scheme shown in Fig. 3. In the second order dynamics of the actuator the rate could be estimated using a Kalman filter if the rate sensing is not available. The Kalman filter is an efficient recursive filter that estimates the state of a dynamic system from a series of noisy measurements. The matrix of gains as shown in Fig. 3 is tuned offline using GA. The structure of the compensator is taken from (Oppenheimer and Doman 2004).

1.1 Control allocation for aircraft: graphical illustration

Control allocation is merely a mapping (i.e. linear or non-linear) from total virtual demands in terms of body angular accelerations to the control position setting subject to rate and position constraints. An illustration of control allocation is given in Fig. 4.
Section 2 describes the interaction of first order actuator dynamics and control allocation and the structure of the compensator is established in this section for first order actuator dynamics. Similarly, in section 2.2 the structure of the compensator is established for second

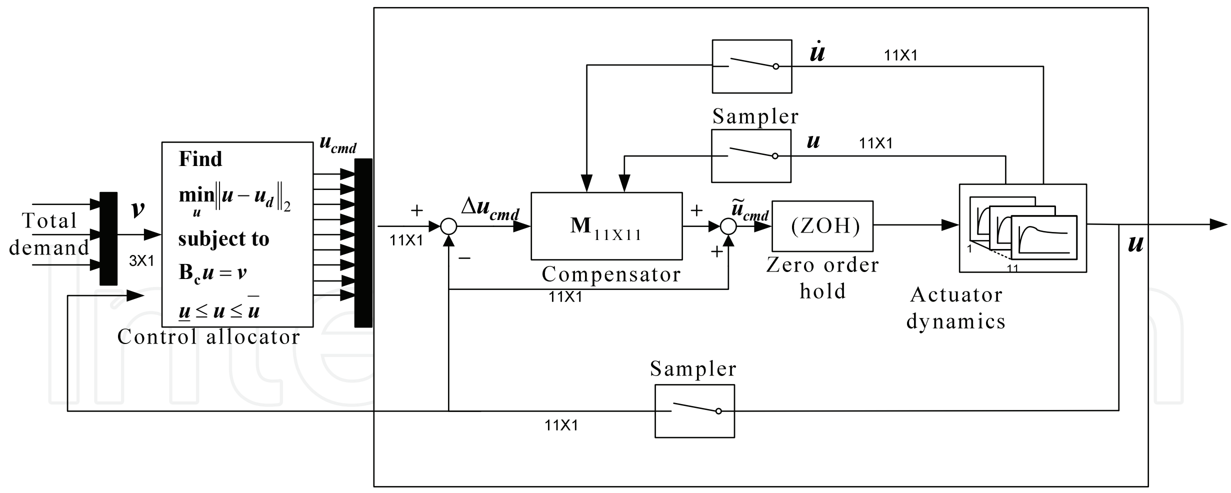

Fig. 3. Structure of compensator with actuator dynamics with diagonal gain matrix M of dimension (11X11)

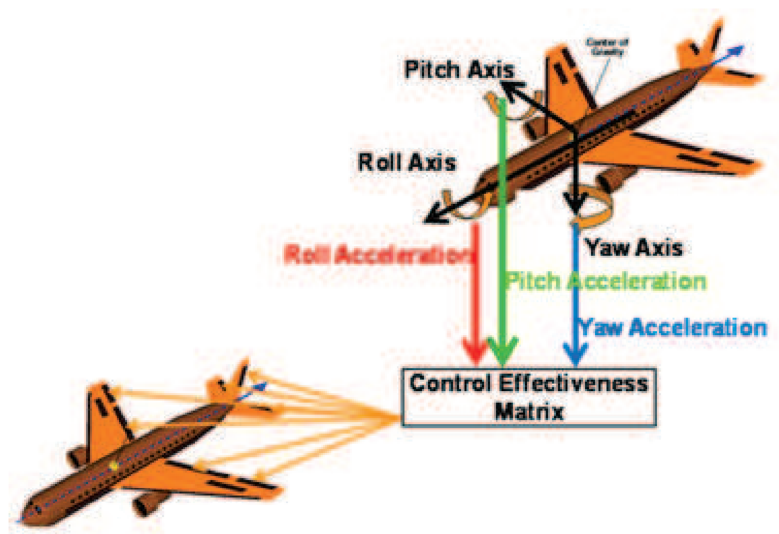

Fig. 4. Control allocation for aircraft: graphical illustration

order actuator dynamics. In section 3 tuning of the compensator parameters using genetic algorithm is described. In section 4 simulation and results for a tuned compensator are shown for a range of first and second order actuator dynamics. Finally, in section 5 some conclusions are established.

2. First-order actuator dynamics interaction

In this section, the effects of first-order actuator on the system are shown in Fig. 2 (Oppenheimer and Doman 2004). Let the dynamics of a single actuator be represented by a continuous time first order transfer function of the form

$$\frac{u(s)}{\tilde{u}_{cmd}(s)} = \frac{a}{s+a} \tag{1}$$

The discrete time solution to the first-order actuator dynamic equation for one sample period is given by

$$u(kT+T) = \mathrm{e}^{-a\mathrm{T}}u(kT) + \int_{kT}^{kT+T} e^{-a(kT+T-\tau)}\,\tilde{u}_{cmd}(\tau)d\tau \tag{2}$$

where T is the sampling time. This result does not depend on the type of hold because u is specified in terms of its continuous time history, $\tilde{u}_{cmd}(t)$ over a sample interval (Franklin *et al.* 1998). The most common hold element is zero-order hold (ZOH) with no delay, i.e.

$$\tilde{u}_{cmd}(\tau) = \tilde{u}_{cmd}(kT),\;\; kT \le \tau \le kT+T \tag{3}$$

Performing substitution

$$\gamma = kT + T - \tau \tag{4}$$

in Eq. (2) yields

$$u(kT+T) = \mathrm{e}^{-a\mathrm{T}}u(kT) + \int_{0}^{T} e^{-a\gamma}\,\tilde{u}_{cmd}(kT)d\gamma \tag{5}$$

Defining

$$\begin{gathered}\Phi = \mathrm{e}^{-a\mathrm{T}} \\ \Gamma = \int_0^T e^{-a\gamma}\,d\gamma\end{gathered} \tag{6}$$

Eq. (5) can be written as a difference equation of standard form

$$u(k+1) = \Phi u(k) + \Gamma\tilde{\mathrm{u}}_{\mathrm{cmd}}(\mathrm{k}) \tag{7}$$

The signal $\tilde{u}_{cmd}(k)$, is held constant over each sampling period. The command to actuator is given by

$$\tilde{u}_{cmd}(k) = \Delta u_{cmd}(k) + u(k) \tag{8}$$

The command increment change in actuator position over one sample as shown in Fig. 5 is defined by

$$\Delta u_{cmd}(k) \triangleq u_{cmd}(k) - u(k) \tag{9}$$

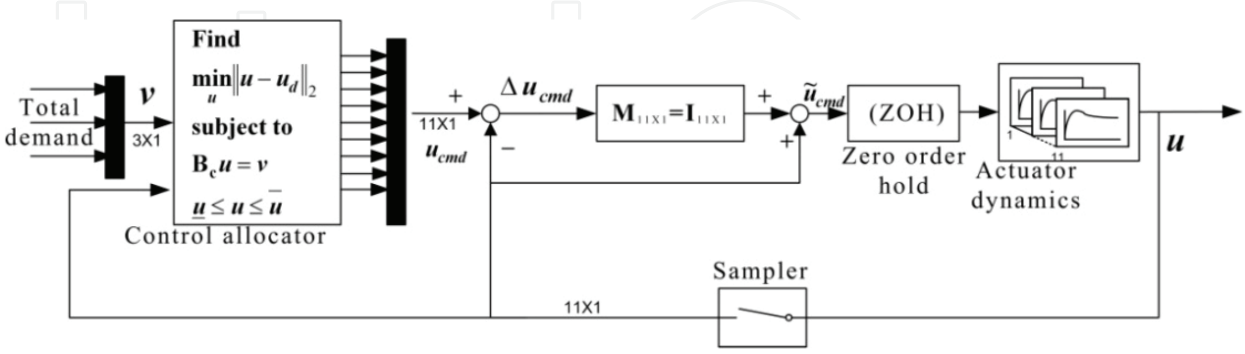

Fig. 5. Command increment change in actuator position with gain matrix M equal to Identity matrix I of dimension (11X11)

where u_{cmd} is the actuator command coming from the control allocator. Since the effector commands are held constant for one sample period then $\Delta u_{cmd}(k)$ appear to be a step command from the measured position $u(k)$. Substituting Eq. (8) in Eq. (7) gives

$$u(k+1) = \Phi u(k) + \Gamma(\Delta u_{cmd}(k) + u(k)) \tag{10}$$

If $\Gamma < 1$, the incremental commanded signal from the control allocation algorithm, $\Delta u_{cmd}(k)$ is attenuated by the actuator dynamics, thus $u(k+1) \neq u_{cmd}$. The objective is to find the gain M that changes the output of the control allocation algorithm such that $u(k+1) = u_{cmd} = \Delta u_{cmd}(k) + u(k)$ (Oppenheimer and Doman 2004). Hence

$$u(k+1) = \Phi u(k) + \Gamma(\mathrm{M}\Delta u_{cmd}(k) + u(k)) \tag{11}$$

The gain M is tuned by using the genetic algorithm in section 3.2. If there is a bank of first order actuator dynamics, then the gain M is chosen to be a diagonal matrix **M** of dimensions (11x11), as shown in Fig. 3 and Fig. 5.

2.1 Example showing effect of first-order actuator dynamics

Let us consider an example with

$$\mathbf{B_c} = 10^{-3}\begin{bmatrix} -4.2 & 4.2 & -5.0 & 5.0 & 0 & 0 & 0 & 0 & 0 & 2.6 & 0.7 \\ -0.9 & -0.9 & -2.9 & -2.9 & -9.4 & -9.4 & -6.9 & -6.9 & -80.5 & -7.7 & 5.8 \\ -0.2 & 0.2 & -0.1 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Position in (deg) and rate in (deg/s) constraints are defined as follows:

$$\begin{aligned} u_{min} &= [-20 \quad -20 \quad -12 \quad -12 \quad -23 \quad -23 \quad -23 \quad -23 \quad -12 \quad -25 \quad -25]^T \\ u_{max} &= [20 \quad 20 \quad 15 \quad 15 \quad 17 \quad 17 \quad 17 \quad 17 \quad 3 \quad 25 \quad 25]^T \end{aligned} \tag{12}$$

$$\begin{aligned} \dot{\rho}_{min} &= [-45 \quad -45 \quad -45 \quad -45 \quad -37 \quad -37 \quad -37 \quad -37 \quad -0.5 \quad -50 \quad -50]^T \\ \dot{\rho}_{max} &= [45 \quad 45 \quad 45 \quad 45 \quad 37 \quad 37 \quad 37 \quad 37 \quad 0.5 \quad 50 \quad 50]^T \end{aligned} \tag{13}$$

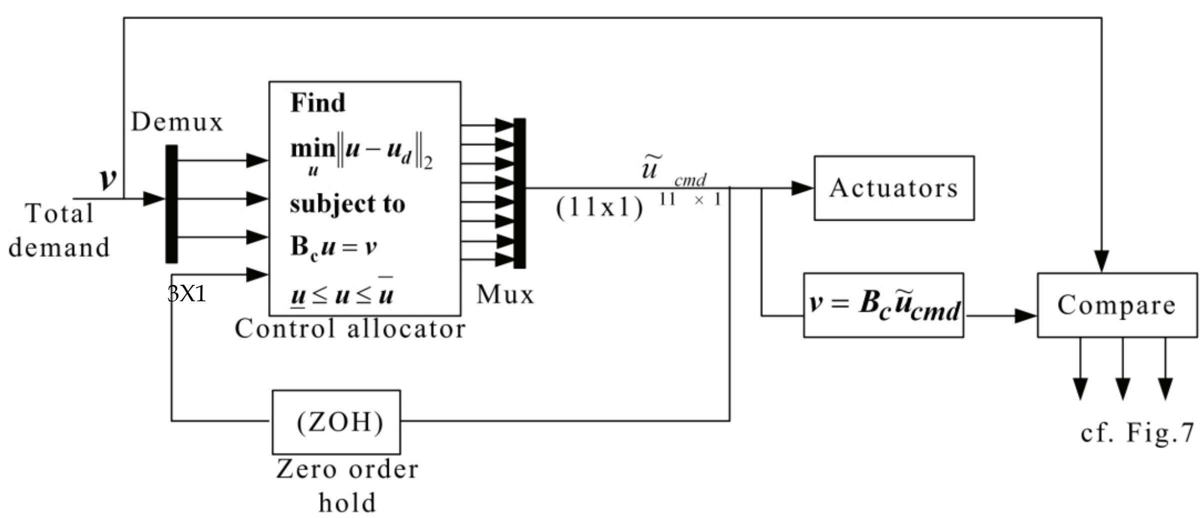

Fig. 6. Block diagram with desired demand produced by the control allocator and compared with the actual demand when there is no actuator dynamics included[1]

First the time response of control allocation without actuator dynamics is shown in Fig. 6 and Fig 7. It can be seen that if the actuators are fast enough to cater for the rigid body

[1] cf. (from Latin *confer*) means compare

modes, there is no need to consider the actuator dynamics and hence one to one mapping between the control allocator and control surfaces is sufficient. This would not be the case with the non aerodynamic actuators, so actuator dynamics cannot be ignored. It can be seen from the results shown in Fig. 8 and Fig. 9 that how the actuator dynamics affects the outcome of the control allocator. It can also be seen that how the control allocator command is attenuated. The first-order actuator dynamics used for this example are given as

$$\frac{u(s)}{\tilde{u}_{cmd}(s)} = \frac{0.6128}{s + 0.6128} \tag{14}$$

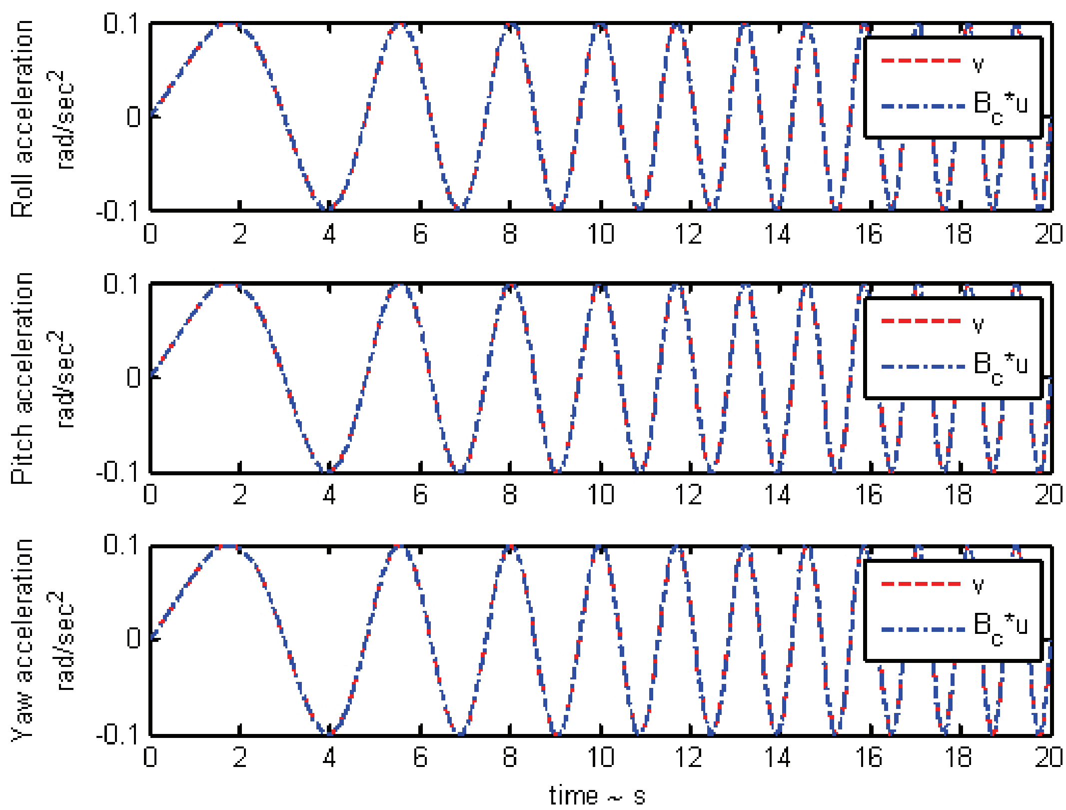

Fig. 7. Comparison of results with no dynamics of actuator involved

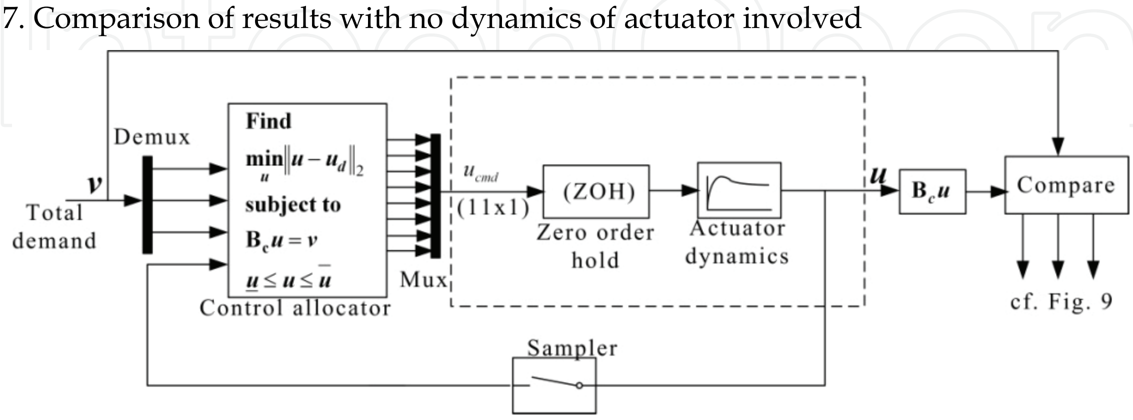

Fig. 8. Block diagram with desired demand produced by the control allocator and compared with the actual demand when there is actuator dynamics included

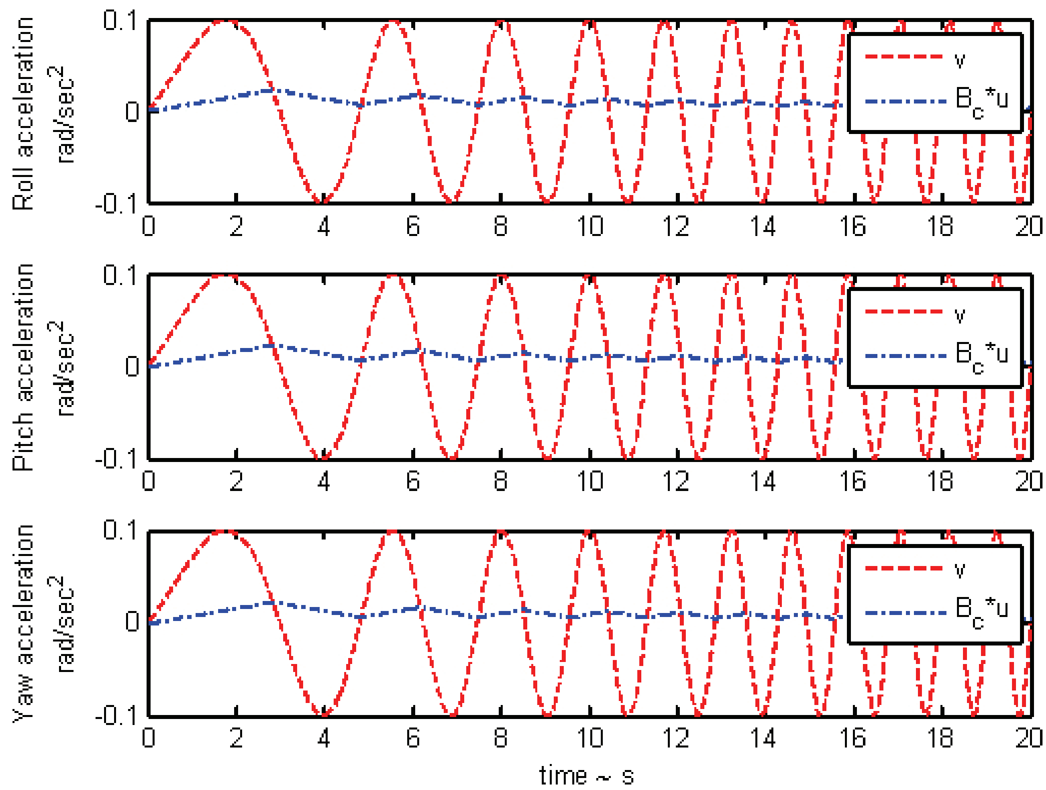

Fig. 9. Time responses of desired and actual responses of virtual demand with actuator dynamics included

In the following section the second order actuator dynamics are parameterised for the design of the compensator.

2.2 Second-order model dynamics interaction

In this section, the effects of second – order actuator on the system is shown in Fig. 2 (Oppenheimer and Doman 2004). Let the dynamics of a second actuator be represented by a continuous time second order function of the form

$$\frac{u(s)}{\tilde{u}_{cmd}(s)} = \frac{k}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{15}$$

The state space representation of this transfer function is given

$$\begin{aligned} \begin{bmatrix} \dot{u}(t) \\ \ddot{u}(t) \end{bmatrix} &= \begin{bmatrix} 0 & 1 \\ -\omega_n^2 & -2\zeta\omega_n \end{bmatrix} \begin{bmatrix} u(t) \\ \dot{u}(t) \end{bmatrix} + \begin{bmatrix} 0 \\ k \end{bmatrix} \tilde{u}_{cmd}(t) \\ \begin{bmatrix} \dot{u}(t) \\ \ddot{u}(t) \end{bmatrix} &= \mathbf{A} \begin{bmatrix} u(t) \\ \dot{u}(t) \end{bmatrix} + \mathbf{B}\tilde{u}_{cmd}(t) \end{aligned} \tag{16}$$

$$\begin{bmatrix} \dot{u}(t) \\ \ddot{u}(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u(t) \\ \dot{u}(t) \end{bmatrix} = \mathbf{C} \begin{bmatrix} u(t) \\ \dot{u}(t) \end{bmatrix} \tag{17}$$

The discrete time solution to the second-order actuator dynamic Eq. (16) to Eq. (17) for one sample period is given by

$$\begin{bmatrix} u(kT+T) \\ \dot{u}(kT+T) \end{bmatrix} = \mathrm{e}^{\mathbf{A}\mathrm{T}} \begin{bmatrix} u(kT) \\ \dot{u}(kT) \end{bmatrix} + \int_{kT}^{kT+T} e^{\mathbf{A}(kT+T-\tau)} \mathbf{B}\, \tilde{u}_{cmd}(\tau) d\tau \tag{18}$$

where T is the sampling time. This result does not depend on the type of hold because $\tilde{u}_{cmd}$ is specified in terms of its continuous time history, $\tilde{u}_{cmd}(t)$ over a sample interval (Franklin *et al.* 1998). A zero-order hold (ZOH) with no delay is given by

$$\tilde{u}_{cmd}(\tau) = \tilde{u}_{cmd}(kT), \;\; kT \le \tau \le kT + T \tag{19}$$

Performing substitution

$$\gamma = kT + T - \tau \tag{20}$$

In Eq. (18) yields

$$\begin{bmatrix} u(kT+T) \\ \dot{u}(kT+T) \end{bmatrix} = \mathrm{e}^{\mathbf{A}\mathrm{T}} \begin{bmatrix} u(kT) \\ \dot{u}(kT) \end{bmatrix} + \int_{0}^{T} e^{\mathbf{A}\gamma}\, d\gamma \mathbf{B} \tilde{u}_{cmd}(kT) \tag{21}$$

Defining,

$$\begin{gathered} \mathbf{\Phi} = \mathrm{e}^{\mathbf{A}\mathrm{T}} = \begin{bmatrix} \Phi_{1,1} & \Phi_{1,2} \\ \Phi_{2,1} & \Phi_{2,2} \end{bmatrix} \\ \mathbf{\Gamma} = \int_{0}^{T} e^{\mathbf{A}\gamma}\, d\gamma \mathbf{B} \end{gathered} \tag{22}$$

$$\begin{bmatrix} u(kT+T) \\ \dot{u}(kT+T) \end{bmatrix} = \mathbf{\Phi} \begin{bmatrix} u(kT) \\ \dot{u}(kT) \end{bmatrix} + \mathbf{\Gamma} \tilde{u}_{cmd}(kT) \tag{23}$$

The first state variable $u(kT+T)$ equation can be written as

$$u(k+1) = [\Phi_{1,1} \quad \Phi_{1,2}] \begin{bmatrix} u(kT) \\ \dot{u}(kT) \end{bmatrix} + \tilde{u}_{cmd}(kT) \int_{0}^{T} k\Phi_{1,2}\,(\gamma) d\gamma \tag{24}$$

Parameterizing Eq. (24) will give

$$u(k+1) = C_1 u(k) + C_2 \dot{u}(k) + C_3 \tilde{u}_{cmd}(k) \tag{25}$$

where $C_1 = \Phi_{1,1}$, $C_2 = \Phi_{1,2}$ and $C_3 = \int_0^T k\Phi_{1,2}\,(\gamma) d\gamma$.
The objective is to find M to modify the $\Delta u_{cmd}(k)$, as shown in Fig. 3 such that $u(k+1) = u_{cmd}(k)$.

$$\Delta u_{cmd}(k) + u(k) = C_1 u(k) + C_2 \dot{u}(k) + C_3(\mathrm{M}\Delta u_{cmd}(k) + u(k)) \tag{26}$$

Solving for M gives (Oppenheimer and Doman 2004)

$$\mathrm{M} = \frac{\Delta u_{cmd}(k) + (1 - C_3 - C_1) u(k) - C_2 \dot{u}(k)}{C_3 \Delta u_{cmd}(k)} \tag{27}$$

These parameters C_1, C_2 and C_3 are tuned using genetic algorithm optimisation. Here it is assumed that the positions and rate of change of actuators are available. If there is a bank of second order actuator dynamics then M is chosen to be a diagonal matrix $\mathbf{M}$ of dimension (11X11).

In the following a stochastic evolutionary algorithm technique was discussed and applied to tune the parameters for the compensator design in section 3.

3. Tuning of compensator to mitigate interaction using GAs

The idea is to combine the design objective in the form of a cost function that is to be optimised using an optimizer such as a Genetic Algorithm. Where the cost function includes the time domain objective; the tracking error is transformed into the integrated square of error between the commanded signal and actual output u of the actuators. In addition there is another design objective, exception handling (e.g. division by zero) and this is also included in the cost function. The schematic is shown in Fig. 10.

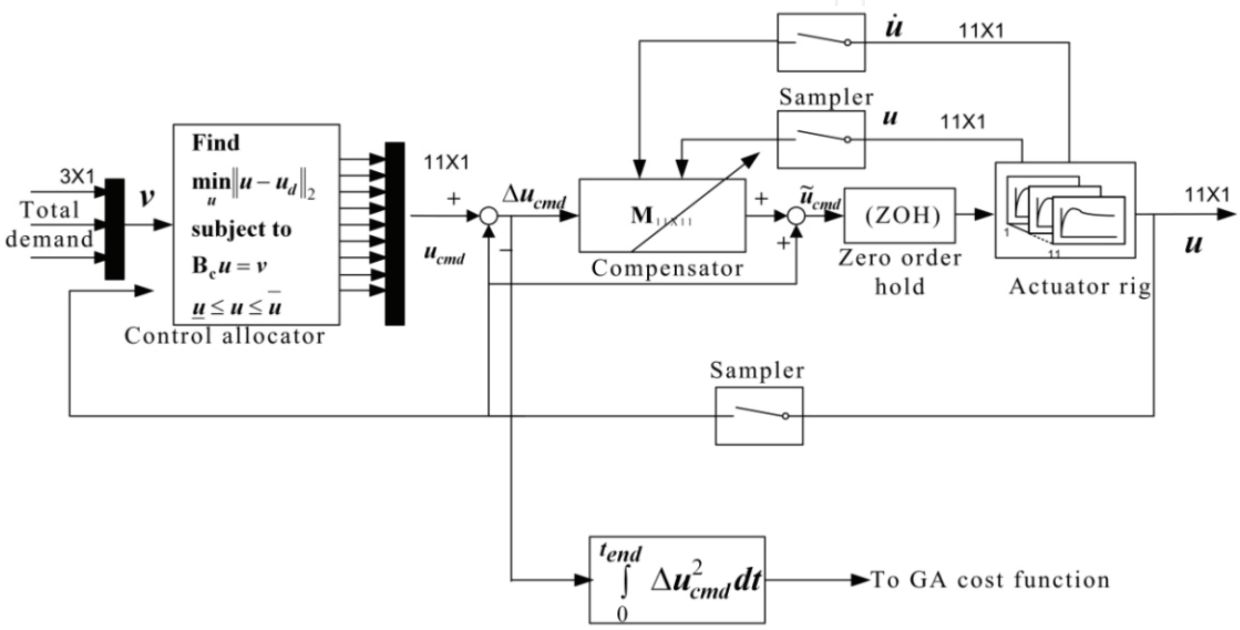

Fig. 10. Cost function error generated by the simulation

Numerically the cost function for tracking error is given by

$$E_{track} = \int_0^{t_{end}} \Delta u_{cmd}^2 dt \tag{28}$$

where the t_{end} is the simulation run time.
Numerically the cost function for exception handling is given by

$$E_{excep} = \begin{cases} \text{penalty } if \text{ exception generated} \\ 0 \; if \text{ no exception} \end{cases} \tag{29}$$

In Eq. (29) the penalty is assigned as a large number like 10^{10} so that the individual generating this exception would most likely not be selected in the next generation because of having very low fitness value. Numerically the combined cost function is given as

$$J = E_{track} + E_{excep} \tag{30}$$

This cost function is then minimised to tune the parameters for the compensator. In the next section GA based optimisation details are given.

3.1 GA based optimisation

Genetic Algorithms are a part of Evolutionary Computing which is a rapidly growing area of Artificial Intelligence. *"GA take up the process of evaluating the relative fitness of the individuals of a large population called genes, to select for a new generation, and mimic mutations and crossing over (mixing genes from two parent genes to form offspring genes), the so called evolution phenomenon"* (Lindenberg 2002). Unlike biological evolution, in GA the gene controls some other processes like compensator parameters in this work, and is evaluated by comparing properties of the process instead of simply computing some function on the gene space (Lindenberg 2002). Genes used in GA are encoded as bit strings, and their fitness (Lindenberg 2002) is a relation described by a real valued fitness function f on the set of bit strings $H = \{b_0, \dots\dots, b_{n-1}\}$ such that gene a is fitter than gene b if $f(b) < f(a)$.
Optimisation using GA begins with a set of solutions (represented by chromosomes) called the population. Solutions from one population (based on some selection criteria) are taken and used to form a new population. The flow chart of illustration of GA is shown in Fig. 11. After selection of an encoding method (binary encoding for this case) and fitness function (cost function value for an underlying gene), the algorithm proceeds in the following steps (Lindenberg 2002).

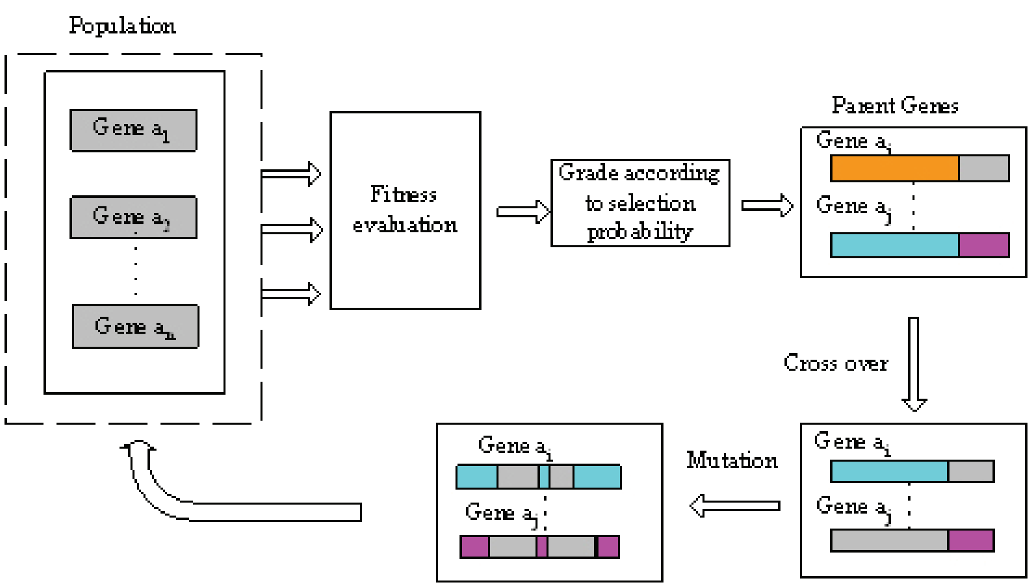

Fig. 11. Flow chart explanation of Genetic Algorithms

- Select an initial population indexed by P at random from some subset of *H*
- Repeat the following
    - evaluate the fitness of all the b_P and use them to assign selection probabilities r_P to b_P
    - select a new population, dropping genes of low fitness and duplicating fit ones, keeping index set *P* (the size of population)
    - apply the genetic operators of mutation and crossing over (after forming couples)

This is repeated until some condition (for instance maximum number of generations or improvement of the best solution) is satisfied. The main advantage (Leigh 2004) of GA over other optimizers is their parallelism, GA is travelling in a search space using more individuals so they are less likely to get stuck in a local minima. The most important attributes of GA are mutation and cross over. A good cross over rate is expected to take better parts of parent genes to the next generation. Mutation on the other hand changes the individuals and if it is kept to a safe low level it helps the population to avoid falling in local minima. This makes GA different from other optimisers, and particularly suitable for non-convex optimisation problems like the compensator parameter optimisation in this research. The main disadvantage linked with GA is the higher computation time and required resources, but this can be avoided if there is a possibility to stop the GA anytime in the routine. Also with the ever increasing processing power of computers over time this constraint diminishes.

3.2 Optimizing routine using GA

Numerically the optimizing problem is given as "Find **M** by minimizing J".

$$\min_{\mathbf{M}} J \tag{31}$$

where **M** is a diagonal gain matrix of dimension (11X11). The GA optimising routine is formulated by using the MATLAB Genetic Algorithm Direct Search Toolbox. A flow chart representation of the optimisation routine is shown in Fig. 12.

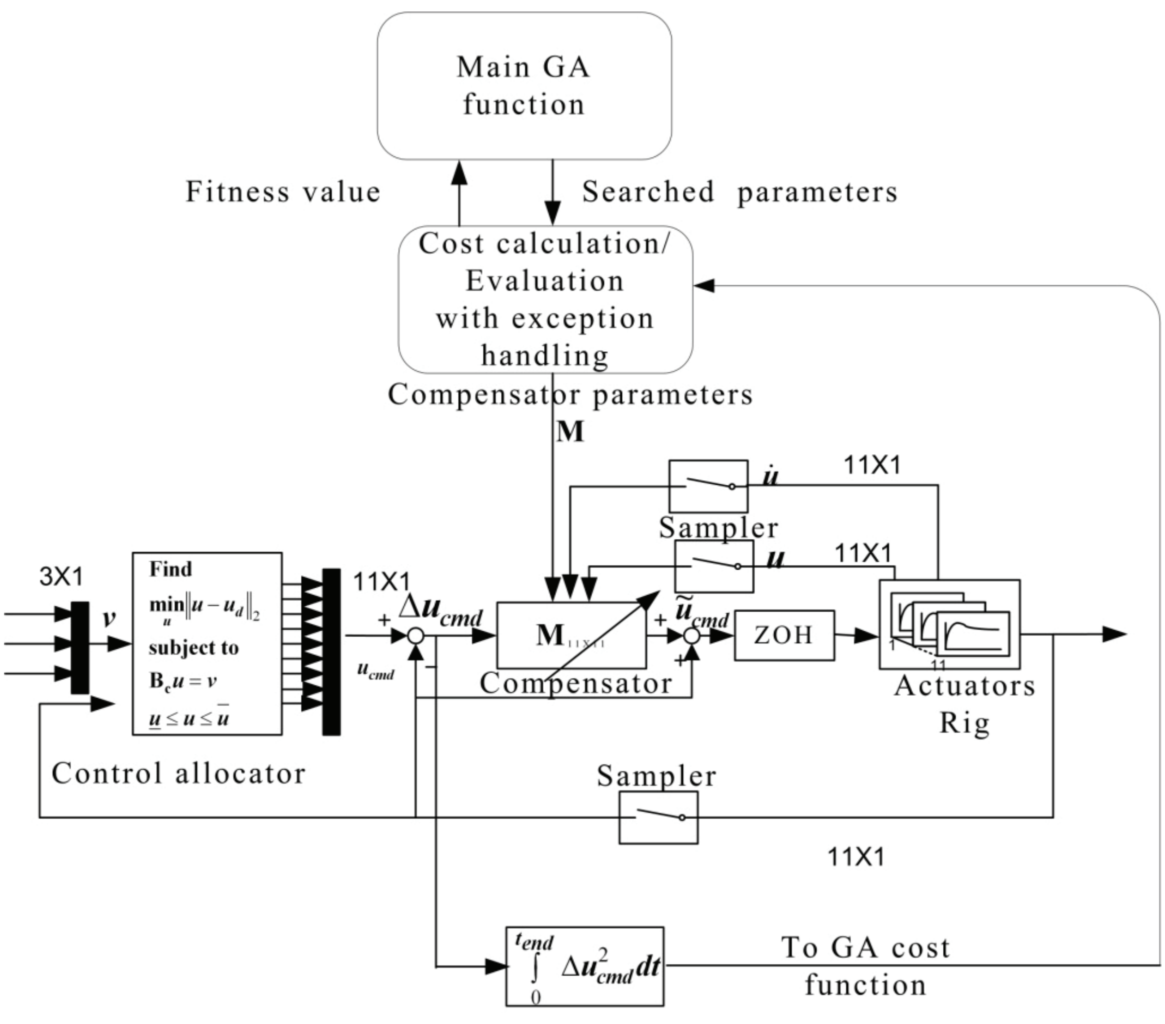

Fig. 12. Flow chart for tuning compensator parameters using GA

The complete process shown in Fig. 11 can be summarised as:

- The GA main function calls the evaluation function, giving searched parameters to calculate compensator parameters
- The evaluation function calculates compensator parameters and calls the simulation model giving the parameters for the compensator
- The simulation model runs the simulation for the given compensator parameter (i.e. individual of population) and returns the value of error between u_{cmd} and actual u
- The evaluation function calculates the cost function value for given errors and returns to the main GA function
- This is repeated for the total number of genes in one generation (population), and then one generation completes, and so the remaining generations are iteratively completed
- The above process is repeated until the cost function attains convergence, or the maximum number of generations is reached.

In the next section the simulation results are given to show how the compensator mitigates the interaction between the control allocation and actuator dynamics.

4. Simulation results

During simulation, a mixture of actuator dynamics was used. In the case of redundant control surfaces diagonal gain matrices were tuned by the GA. The control surfaces were approximated by the transfer functions as shown in Table 1.

Control surfaces	Number of control surfaces	Transfer functions
Ailerons	4	$\frac{270}{s^2 + 22.19s + 270}$
Elevators	4	$\frac{0.6128}{s + 0.6128}$
Stabilizer	1	$\frac{0.0087}{s + 0.0087}$
Rudders	2	$\frac{270}{s^2 + 22.19s + 270}$

Table 1. Aerosurfaces actuator dynamics (Esteban and Balas 2003)

The virtual control signal, $\boldsymbol{v}$, consists of chirps of amplitude 0.1, 0.15, 0.1 (rad/s^2) in roll, pitch and yaw angular accelerations respectively. The frequencies of chirps ranged from 0.1– 1 Hz in 20 seconds. In the processing of the GA routine exception handling is carried out to avoid breaking the GA optimisation process. For example if there is an individual (i.e. gains in diagonal matrix) in the population that gives division by zero that would break the simulation. This is dealt with in an exception handling block, which will give a penalty to that individual without breaking the simulation. In the next generation that individual would not be selected.

Simulations are done with compensation (Fig. 13 and Fig. 14) and without compensator (Fig. 15 and Fig 16). As can be seen clearly from the results with no compensation there is serious attenuation and mismatch, but as soon as the compensation is turned on, $\mathbf{B_c}\boldsymbol{u} = \boldsymbol{v}$ is achieved because sufficient control authority exists.

Deviations in the case of no compensation case means that the desired control surface positions coming out of the control allocator are different from the actual position of control surfaces. This interaction between the control allocator and the actuator dynamics results in serious consequence if the bandwidths of the actuators are not high or, in other words, the actuators are slow.

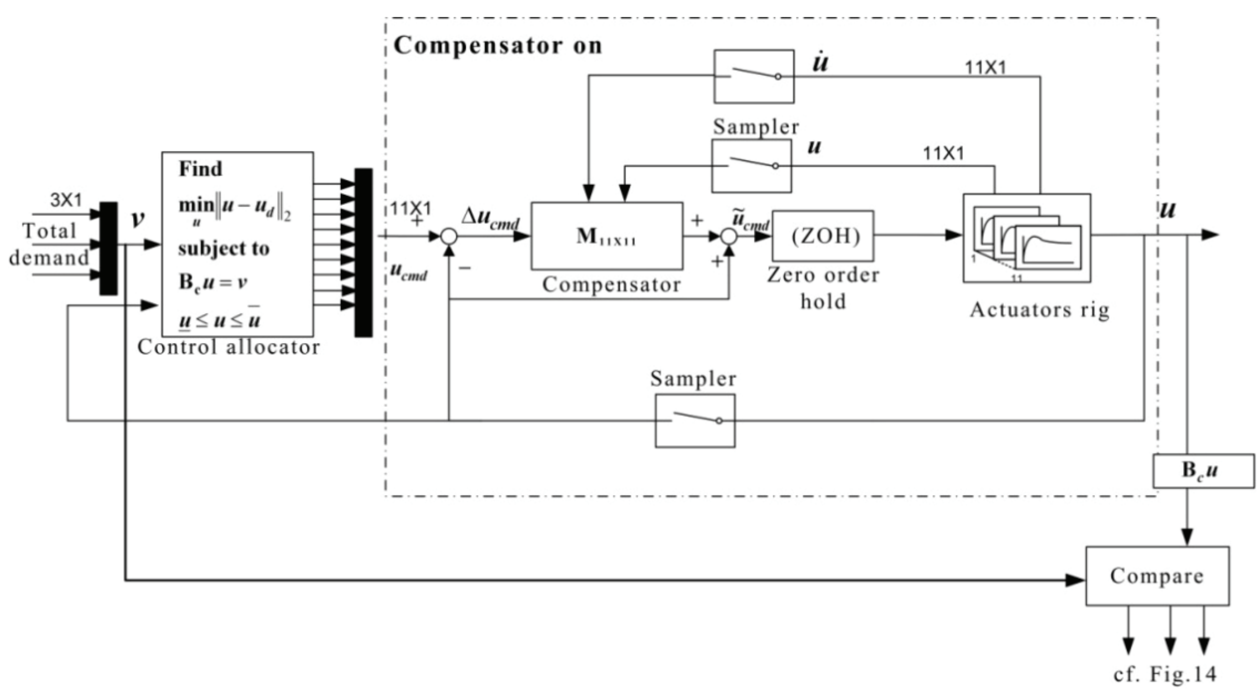

Fig. 13. Implementation scheme for compensator when the compensator is switched on

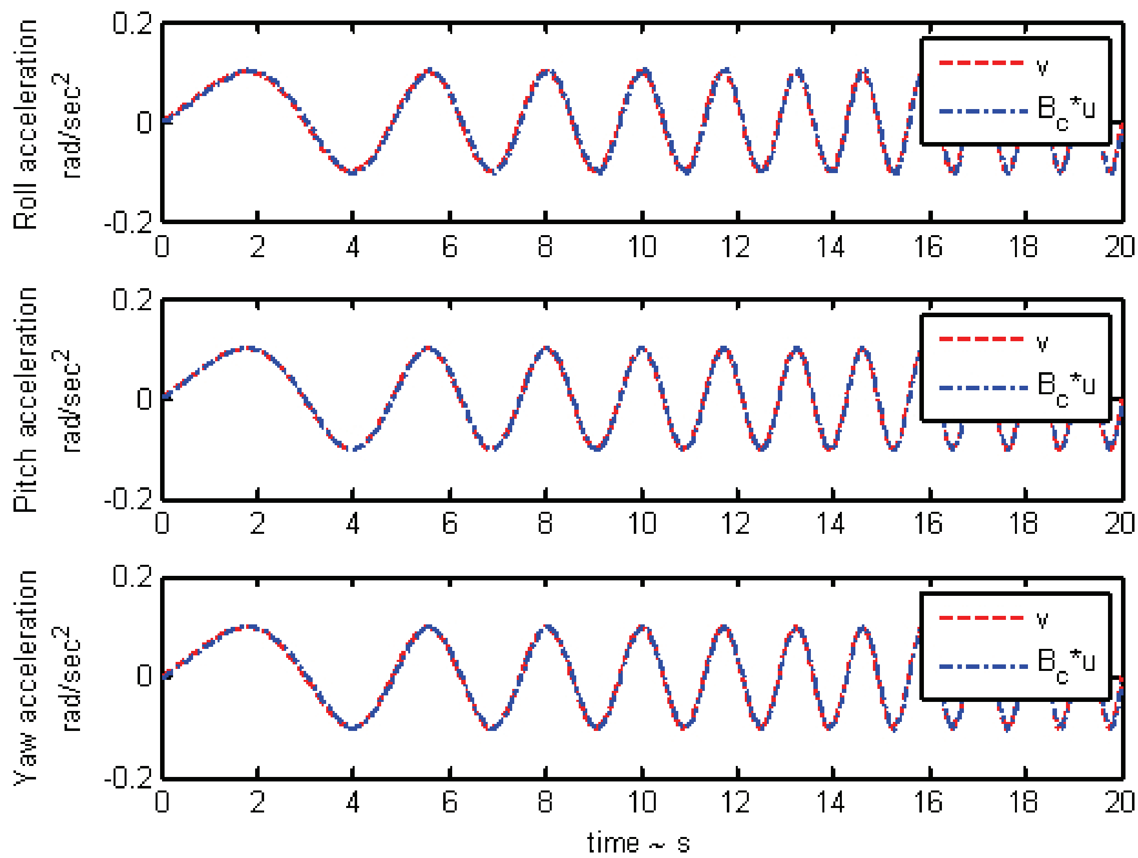

Fig. 14. Desired angular accelerations ($\boldsymbol{v}$) and actual angular acceleration ($\mathbf{B_c}\boldsymbol{u}$) in rad/s2 when compensation is on

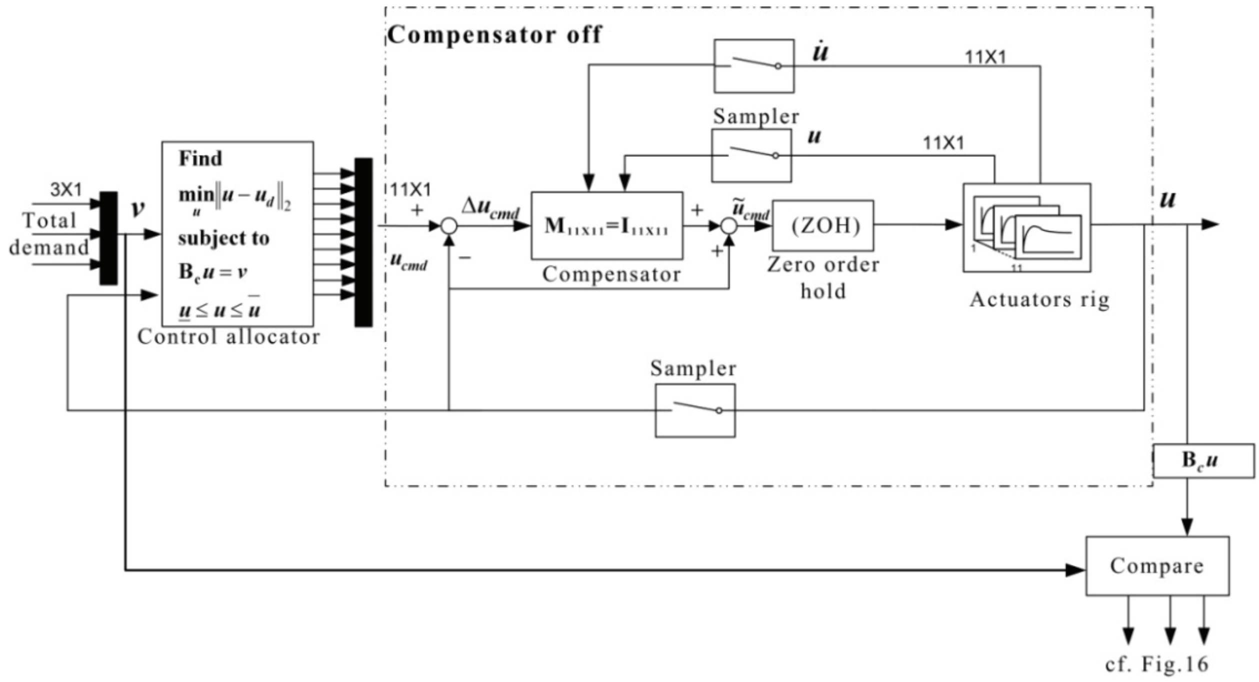

Fig. 15. Implementation scheme for compensator when the compensator is switched off

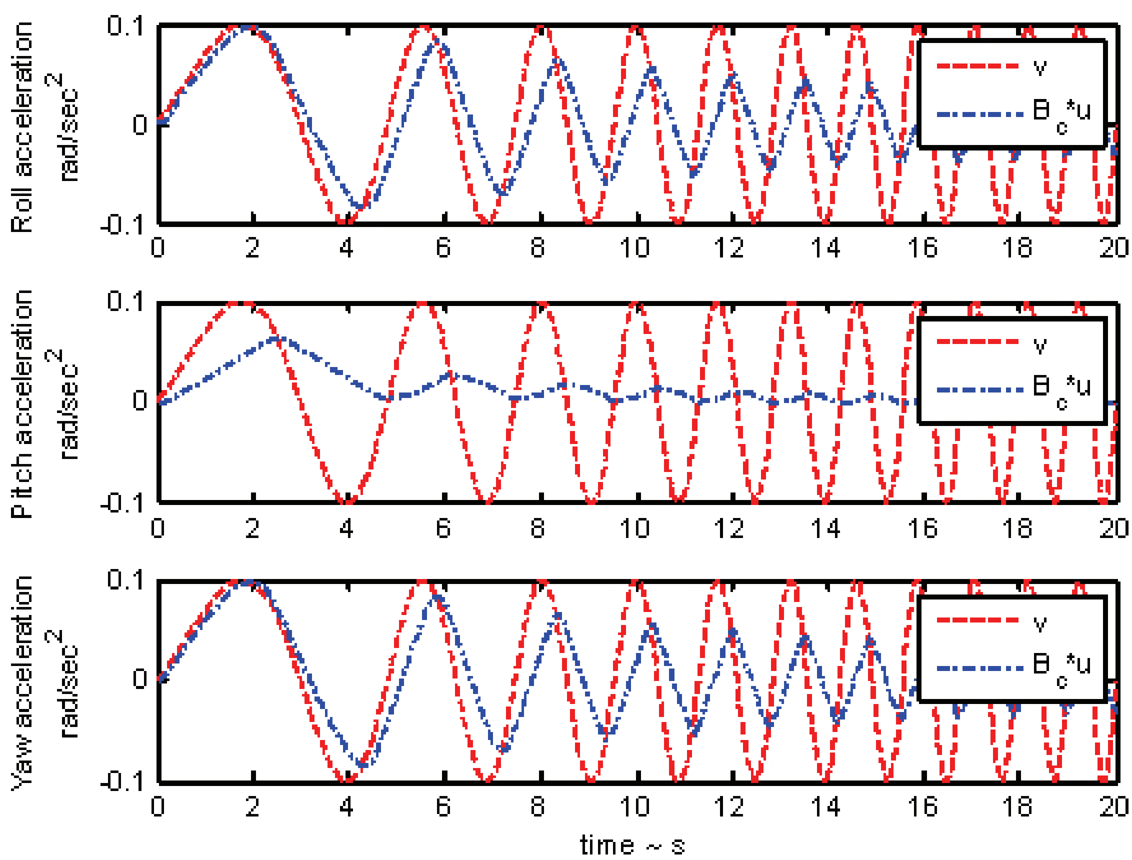

Fig. 16. Desired angular accelerations (v) and actual angular acceleration ($B_c u$) in rad/s2 when compensation is off

5. Conclusions

This chapter details the application of genetic algorithms for the design and tuning of a compensator to alleviate the effects of control allocation and actuator dynamics interaction. The effects of non-negligible actuator dynamics have been investigated first. It was observed that, for the Boeing 747-200, the actuator dynamics cannot be ignored if the excitations are in the range of 0.1 to 1 Hz, which normally depends on the pilot dynamics. Another observation suggests that the bandwidths of the actuators are smaller than the rigid body modes of the aircraft and should not be neglected. The benefit of using a soft-computing methodology for tuning the compensator gains is to avoid the optimisation converging to a local minima and it is seen that the likelihood of the genetic algorithms converging to local minima solution is less as compared to other techniques. In this methodology the model of the actuator is not needed to be known because this methodology was designed to be used on the actuator rig. In the case of the second order actuator, the rates should be either measured or observed. GAs are used offline and the band limited chirps signal is used as the excitation signal in the simulation. However, in the real system a band limited pseudo-random binary signal (PRBS) for this type of identification process could be used as an excitation signal rather than chirp because the later gives cyclic loading on the actuator, which could be problematic.

6. References

Bolling, J.G., (1997) Implementation of Constrained Control Allocation Techniques Using an Aerodynamic Model of an F-15 Aircraft, MSc. thesis, Virginia Polytechnic Institute and State University, Virginia, USA.

Esteban, A. M., Balas, G.J. (2003) A B747-100/200 aircraft fault tolerant and fault diagnostic benchmark, AEM-UoM 2003-1, Aerospace Engineering and Mechanics Department, University of Minnesota, USA.

Franklin, G.F., Powell, J.D. and Workman, M. (1998) Digital control of dynamic systems, 3rd ed., Addison-Wesley Longman, Inc., California

Lindenberg, F.M. (2002) Adaptive Compute Systems lecture notes, Technical University Hamburg-Harburg Germany.

Oppenheimer, M.W. and Doman, D.B. (2004) 'Methods for compensating for control allocator and actuator interactions', Journal of Guidance, Control, and Dynamics, 27(5), pp. 922-927.

Advances in Flight Control Systems

Edited by Dr. Agneta Balint

ISBN 978-953-307-218-0

Hard cover, 296 pages

Publisher InTech

Published online 11, April, 2011

Published in print edition April, 2011

Nonlinear problems in flight control have stimulated cooperation among engineers and scientists from a range of disciplines. Developments in computer technology allowed for numerical solutions of nonlinear control problems, while industrial recognition and applications of nonlinear mathematical models in solving technological problems is increasing. The aim of the book Advances in Flight Control Systems is to bring together reputable researchers from different countries in order to provide a comprehensive coverage of advanced and modern topics in flight control not yet reflected by other books. This product comprises 14 contributions submitted by 38 authors from 11 different countries and areas. It covers most of the currents main streams of flight control researches, ranging from adaptive flight control mechanism, fault tolerant flight control, acceleration based flight control, helicopter flight control, comparison of flight control systems and fundamentals. According to these themes the contributions are grouped in six categories, corresponding to six parts of the book.