Proceedings of IDETC'05
ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference
September 24-28, 2005, Long Beach, California, USA

DETC2005-85444

ON THE EFFECT OF FAULTS IN VIBRATION CONTROL OF FAIRING STRUCTURES [*]

Young Man Kim
Kookmin University,
Seoul, Korea
Email: ymkim@kookmin.ac.kr

Anish Arora, Vinodkrishnan Kulathumani
Dept. of Computer Science and Engineering
The Ohio State University
Columbus, Ohio, USA - 43210
Email: {vinodkri,anish}@cse.ohio-state.edu

Umamaheshwaran Arumugam, Sandeep Kulkarni
Dept. of Computer Science
The Michigan State University
East Lansing, MI, USA - 43210
Email: {arumugam, sandeep}@cse.msu.edu

ABSTRACT

Sensor-actuator networks are increasingly being used in distributed control applications. The cost of sensors and actuators is dropping substantially and hence control by a large number of these components is now feasible. One such application is the damping of acoustic and structural vibration associated with the launch of a rocket. Reliability in the presence of faults is critical for such mission systems. These faults could be broken components, insecure or compromised components offering erroneous data to the control. The network itself could add unpredictable delays and data drop outs that could affect the control in potentially unanticipated ways. In this paper, we consider the Boeing Open Experimental Platform fairing control application for acoustic and structural vibration damping and study the effect of component level and network level faults. We identify several scenarios under which control performance is intolerable. This leads us to design an alternative control scheme. We design the application using a purely local on-off control scheme and compare its performance with that of the original system.

Keywords: Sensor-actuator networks, Fault-effects, Vibration control, Local on-off control.

1 Introduction

A new class of distributed control applications is emerging in the field of sensor-actuator networks. As the cost of sensors and actuators is dropping substantially, it is feasible to achieve control by a network of these components deployed in large numbers. Although wireless networks are in vogue, a wired network is desirable for many of these control applications. Wired networks have better network bandwidth and reliability compared to wireless networks.

By way of an example of a control application, we consider the control of acoustic and structural vibrations associated with the launch of a rocket. Although satellite payloads are enclosed in a fairing made of composite material, they still have to be reinforced to withstand the shock of the launch. The acoustic exci-

[*] THIS WORK WAS PARTIALLY SPONSORED BY DARPA CONTRACT OSU-RF #F33615-01-C-1901, NSF GRANT NSF-CCR-9972368, AN AMERITECH FACULTY FELLOWSHIP, AND TWO GRANTS FROM MICROSOFT RESEARCH.

tation at the payload fairing during launch is modeled as a disturbance with sound pressure levels as large as 140-150dB. A standard approach to attenuate the disturbance is to attach acoustic blankets to the fairing. However, this passive approach is only effective for frequencies above approximately 250 Hz due to limitations in the size and weight of the acoustic blankets [1]. The application of active control to attenuate lower frequencies of vibration in structures has been studied from many years [2,3]. The earlier systems were mostly centralized in control. However, the extension of this technology to large scale systems motivates the design of control that is distributed [4]. These systems are easy to control if information about individual vibration modes can be obtained. Because these MEMS based devices are cheap, a large number of these devices can be embedded on the payload. Combinations of those sensors can be used to obtain the required mode vibration information and then the output from these combinations can be used to provide adequate control. The vehicle subsystem that we consider is characterized by 100s to 1000s of nodes driving sensors and actuators trying to achieve fine grain control.

Since sensors and actuators are low cost and deployed in large numbers, these control applications are subject to a rich class of faults. In distributed control applications involving sensor actuator networks, faults could be broken components, insecure or compromised components offering erroneous data to the control. The network itself introduces vulnerabilities such as unpredictable delays and data dropouts. Since it is a potentially large class of faults, in this paper we experimentally evaluate which of these have substantial or catastrophic effects on the performance of the vibration control system.

We had 2 choices to run our fault injection experiments on. One was a hardware testbed, which is a scaled down version (about 1/16th) of a typical fairing shaped space launch vehicle. Six speakers installed around the fairing approximate the field vibrations. A network of PVDF sensors is attached to the fairing to map the response of the fairing to acoustic inputs. Piezoelectric actuators control the structural response of the fairing. Since it was infeasible to inject potentially catastrophic faults into this hardware testbed, we considered the Boeing Open Experimental Platform, which is a simulation framework intended to capture the vibroacoustic damping problem on a satellite launch vehicle.

Contributions We identify potential component and network level faults for the fairing control application. Using a fault injection framework we evaluate the effect of these faults on the control performance of the Boeing OEP. Our study reveals that the performance of the Boeing OEP hierarchical control scheme degrades substantially in the presence of component crashes, random behavior of sensors and actuators and network delays. This leads us to design an alternative control scheme - one that is purely local. We find that the local on-off control scheme is tolerant to a rich class of faults but requires a higher sampling frequency and hence is more energy consuming. While many of the existing techniques for fault-tolerant control are based on fault-detection and isolation based approaches [5–7], our observations motivate the need for designing reliable control scheme that maintain stability and performance in the presence of arbitrary component faults.

Organization of the Paper In section 2, we describe the architecture of the Boeing Open Experimental Platform fairing control application. In section 3, we describe the experimental setup, define metrics that we use for performance evaluation and analyze the results of the fault-injection experiments. In section 4, we present an alternate control scheme and compare that with the Boeing control scheme. In section 5, we present conclusions and goals for future research.

2 Boeing Platform Simulator

This section describes the architecture of the Boeing Open Experimental Platform . The simulated environment of the Boeing OEP application includes a fairing plant model which is a simulation of the fairing structure, a hierarchical control application [8, 9] and a fault injection framework each of which are described in the following subsections.

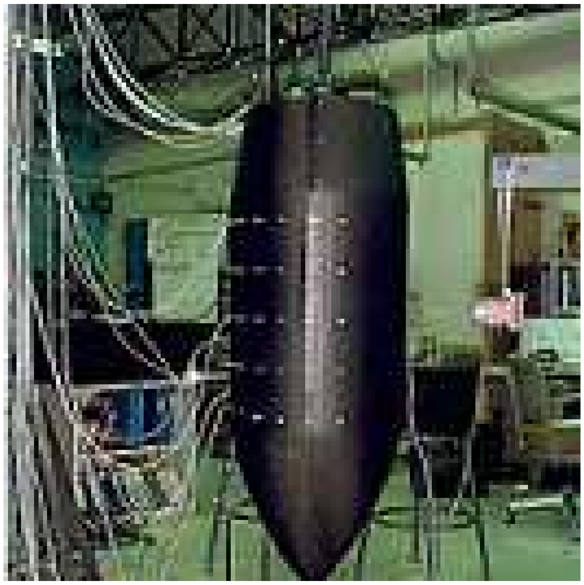


Figure 1. Fairing Payload in the Hardware Testbed

2.1 Vibro-Acoustic Model

The Boeing OEP contains a simulation of the fairing structure, using 100 computational nodes. Each software node comprises a sensor, an actuator and a processing element and simulates a distinct hardware node embedded in the physical structure. The dynamic model of the fairing used in the development of the Boeing OEP Simulator is a fully coupled structural acoustic model relating structural motion to the internal acoustic response.

The model of the vibro-acoustic dynamics of the faring can be represented by the following equation.

$$M\ddot{x} + D\dot{z} + Kx = Hw + F \tag{1}$$

where x is physical displacement, M, D and K are the modal mass, damping and stiffness, and F is applied (control) force. w is the disturbance. One can represent the physical displacement as the product of mode shape (ϕ) and modal coordinate (q). The mode shape describes how a mode varies spatially (over the fairing) while the modal coordinate characterizes how a mode varies temporally. Using this representation produces the following equation.

$$M\phi\ddot{q} + D\phi\dot{q} + K\phi q = Hw + F \tag{2}$$

Multiplying by the mode shape transpose gives the following equation.

$$\phi^T M\phi\ddot{q} + \phi^T D\phi\dot{q} + \phi^T K\phi q = \phi^T Hw + \phi^T F \tag{3}$$

This can now be written in state space form as follows

$$\begin{aligned} \dot{z} &= Az + Bu + \overline{H}w \\ y &= Cz \end{aligned}$$

The output y is the sensor output. The matrix A in Eq. (4) can be written as follows

$$A = \begin{pmatrix} 0 & I \\ -\Lambda & -2\zeta\omega_n \end{pmatrix} \tag{4}$$

Λ is the diagonal matrix of system eigenvalues. $2\zeta\omega_n$ is the matrix of modal damping ratios multiplied by the undamped natural frequencies.

2.2 The Hierarchical Control Application

The 100 node system is partitioned into several groups. Each group acts to damp a particular mode of vibration in the fairing. The number of nodes assigned to each mode depends on the frequency of that mode and the energy required to achieve the dampening. There can be as many as 20 modes to address. As shown in Fig. 2, the system functionality is divided into 4 main categories that are described below. Each node implements one component of the hierarchical control application.

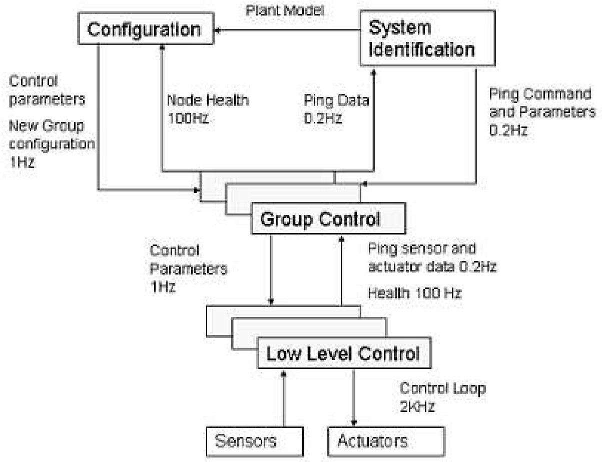


Figure 2. Functional Diagram Of Boeing OEP Platform

Low Level Control Low level control performs actions associated with an individual node. It detects the displacement of the fairing and damps vibration by providing a command output to the actuator. The controller actuation command is combined with the ping command generated by the system identification function to generate the total actuator command. The controller works at 2 KHz. Each node also provides health status information to the group control function at a frequency of 100Hz.

Group Control The group control function gathers health information from all individual nodes and provides it to the configuration function. Updated modal characteristic data is received from the system identification function. This data is refined as appropriate for use by low level control algorithms and the modified control parameters are sent to individual nodes. Individual group members are also activated or deactivated based on commands from the configuration function.

System Identification Function The system identification function is responsible for determining the current fairing vibroacoustic modal characteristics. This is achieved by a set of commands to ping selected actuators and then measuring the response from corresponding sensors. In the Boeing OEP, pinging is performed for a duration of 1 second at a frequency of 0.2 Hz. Since the sampling rate is 2 KHz, a total of 4000 values from sensors and actuators are stored over this interval and used to compute modal characteristics such as shapes, frequencies and damping parameters associated with each vibroacoustic mode of interest.

Configuration function Based on information obtained from the group control function regarding node health status and the vibroacoustic modal characteristic data obtained from the system identification function, the configuration module defines nodes that are more effective in controlling each mode of interest. Thus it re-optimizes group membership and also modifies group configuration and/or low level parameters. This information is then provided to the group control function.

2.3 Fault Injection Framework

The Boeing OEP is subject to potential platform level and network level faults. Platform level faults that capture hardware outages and software failures are injected into the Boeing OEP using script files. Separate script files are used for sensor and actuator faults. Platform level faults are specified by their type and their start and stop times. We model network faults using communication delays and we re-implement certain functions in the Boeing OEP to simulate these delays.

We now summarize the types of faults that we inject into the Boeing OEP:

1. Fail-stop faults Sensors, actuators or the processing element at a node simply stop functioning due power failure at the node, physical damage etc. Communication with such nodes is cut off. Fail-stop faults are detectable.

2. Crash faults Nodes or components within a node may fail in such a way that they are not detectable. For example, communication with nodes remains intact, however the actuators stop functioning. The faults cannot be detected by higher layers to take corrective action.

3. Random and stuck-at faults Hardware failures or hardware-software interface failures cause sensors and actuators to behave in an arbitrary manner. Random sensors return an arbitrary reading. Random actuators perform an arbitrary action. We specify the randomness of a node by providing a mean and standard deviation for the values. A special case of the random fault category is a stuck-at fault. Stuck-at sensors return a constant specified value. Stuck-at actuators always apply a constant specified force.

4. Byzantine faults A Byzantine actuator is one that applies an arbitrary control input to the plant at all times. Faults in the hardware and the underlying software services can manifest themselves in the form of arbitrary actuator behavior. In reality they could even be the worst possible at all times. Byzantine faults represent a nondeterministic and potentially malicious behavior of the components. They capture an arbitrary behavior of actuators that could potentially be the worst possible at all times. In order to study the effect of low level faults that can cause maximum harm to the system, we program certain nodes to behave to generate worst possible signal values. These Byzantine nodes generate values of maximum magnitude in a direction opposite to the correct one. Such Byzantine faults could also arise due to security loop holes in the system.

5. Debonding faults Under extensive vibrations, sensors and actuators can physically separate from their contacts on the fairing. So sensors and actuators are not fully effective. They perform only at a fraction of their capability. Thus a p% debonded actuator applies p% of the actual force that it is supposed to apply. Likewise, a debonded sensor only records a fraction of the actual measurement at its location. The debonding is specified in terms of a final effectiveness percentage and the rate of debonding.

6. Network Faults The network that handles communication between different nodes is itself subject to faults. We model these faults using delay in communication. We study the effect of delays and jitter on control performance. We add the ability to introduce message delays in the Boeing OEP by re-implementing certain functions in the OEP using threads to simulate delays.

3 Fault-Injection Experiments

In this section, we first describe the configuration of the Boeing OEP platform under which all our experiments were performed. We then explain the performance metrics that we use for our analysis and describe each fault-injection experiment.

3.1 Experimental Setup

In the fairing simulator there are 100 physical nodes that are capable of low level actuation. In the configuration that we use, there is 1 configurator node, 1 system id node, 11 group controller nodes, and 87 low level controller nodes. At the beginning of each experiment, an optimum node-mode assignment is assumed to be made. There are 11 active modes. Modes 2 to 9 have 10 nodes assigned to them. Mode 1 has 3 nodes, and modes 10 and 11 have 2 nodes each assigned to them. These assignments can change at every system-id cycle. Initially the low level controllers are not active. They get activated upon receiving control parameters from the group controller. The faults are specified to the system by means of an input file. This file contains the ids of the faulty nodes, the type of the fault and the duration of the fault. Note that each experiment was performed 10 times and the resulting metrics were averaged.

3.2 Performance Metrics

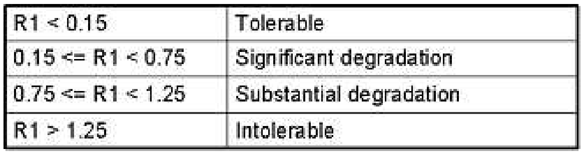
The sum of the squares of structural velocities at the 100 nodes is taken as a measure of the total energy in the fairing. Let E_u be the energy at a given instant in the fairing without any control being applied. Let E_c be the energy in the fairing at a given instant with control being applied. The energy reduction ratio, denoted as ERR for a given experiment is defined as follows:

$$ERR = \frac{Mean(E_c)}{Mean(E_u)} \tag{5}$$

In order to compare results from different fault experiments, we define a performance degradation ratio, denoted as R for each experiment. Let R1 be the energy reduction ratio for a given fault experiment X. Let R2 be the average energy reduction ratio over experiments without any fault for the corresponding duration.

$$R = \frac{R1}{R2} \tag{6}$$

Qualitative Performance Index In the configuration that we operate the Boeing OEP, R2 is found to be 0.07 on an average over 50 experiments without any fault. Based on this measure, we define the qualitative performance indices for any given fault experiment as given in the following table:



R1 < 0.15	Tolerable
0.15 <= R1 < 0.75	Significant degradation
0.75 <= R1 < 1.25	Substantial degradation
R1 > 1.25	Intolerable

Figure 3. Qualitative Performance Index

3.3 Crash Faults

Here we study the effect of actuators within nodes that simply stop functioning. However, communication of these nodes remains intact and hence these failures go undetected. The case of the low level nodes crash is distinguished from that of the group controllers crash since their effects could be different.

Low level controller crash First, 10 low level controller nodes chosen randomly from different modes were made to crash. Each mode had at least one node active. Then, the effect of 10 crashed controllers that belong to the same mode was tested. The latter experiment was repeated for each mode separately. The worst degradation results amongst those are presented. From the observations it is clear that having at least one node active per mode yields tolerable control quality.

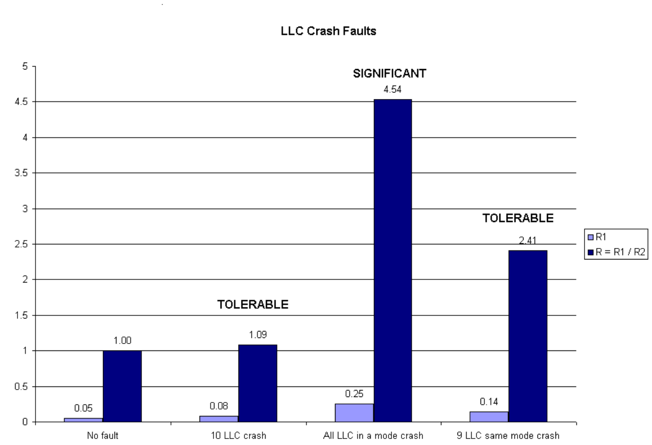


Figure 4. Effect of Low Level Controller Crash Faults on Boeing OEP

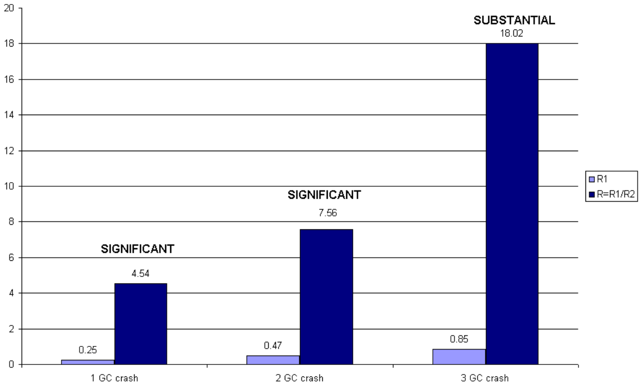


Figure 5. Effect of Group Controller Crash Faults on Boeing OEP

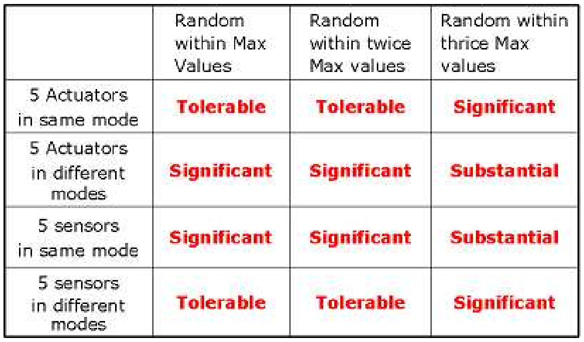
> Observation 1: When 10 low level controllers that are assigned to different modes crash, degradation in control performance is tolerable. When all the 10 controllers belong to the same mode, thus one mode is uncontrollable, control performance significantly degrades.

Group controller crash Group controller nodes are responsible supplying the configurator with node health values and also supplying low level nodes with modified control parameters in every system-id cycle. Thus when a group controller crashes, all the low level nodes continue with the old and now incorrect parameters. If the group controller crashes before the low level nodes are activated, its low level controllers offer no actuation. Experiments were repeated with different group controllers crashing and at different times in a system-id cycle.

> Observation 2: The crash of 1 or 2 group controllers results in significant degradation of control performance. Crashing of 3 group controllers results in substantial degradation.

3.4 Random and stuck-at faults

Based on our experiments without any fault injected, the operating range of sensors and actuators were noted in the form of maximum and average values. In this section we first describe the effect of sensors and actuators generating random values that are close to their operating range and the effect of stuck-at faults. We consider sensors and actuators belonging to same and different modes. We then describe the effect of sensors and actuators producing random values that can be anywhere in the range that they are capable of generating.



	Random within Max Values	Random within twice Max values	Random within thrice Max values
5 Actuators in same mode	Tolerable	Tolerable	Significant
5 Actuators in different modes	Significant	Significant	Substantial
5 sensors in same mode	Significant	Significant	Substantial
5 sensors in different modes	Tolerable	Tolerable	Significant

Figure 6. Minor Random Faults- Summary

Minor random and stuck-at faults 5% of sensors and actuators chosen randomly from different modes were programmed to generate random values entirely within the range of operation. In another set of experiments, they were all chosen from the same mode. The range of random values was then increased to twice and thrice the maximum normal operating value. For testing stuck-at faults, 5% of sensors and actuators were made to generate the maximum operating value. Fig. 6 summarizes our observations with 5 nodes behaving randomly. Fig. 7 shows the comparison between five sensors and actuators getting stuck at the maximum value and them generating constantly varying values within the maximum.

Observation 3: When sensors and actuators generate random values entirely within the range of operation, the degradation of control is tolerable. However when the range is within twice or more of the maximum normal operating value, it results in significant or substantial degradation. When sensors and actuators get stuck-at some arbitrary value, the effect is less severe than constantly varying values and degradation is tolerable.

Extreme Random Faults The system was tested with up to 5 sensors and actuators generating random values to their capacity. For the actuators this was 10 times their normal maximum operating range. For the sensors, this was about 25 times their normal operating range. The nodes were chosen arbitrarily from different modes. The results are summarized in Fig. 8 and Fig. 9.

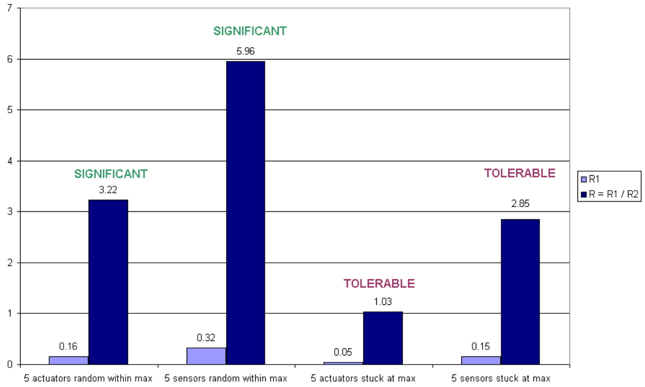


Figure 7. Random Vs Stuck-at Faults

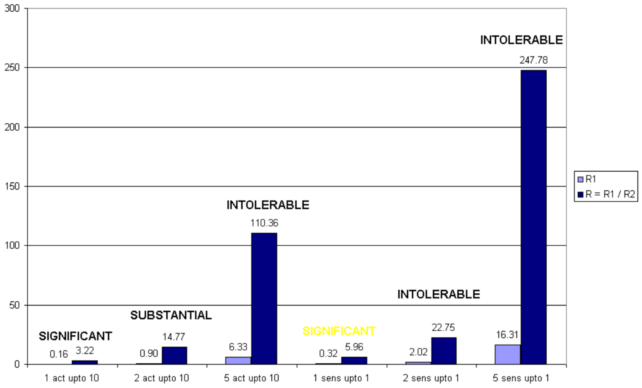
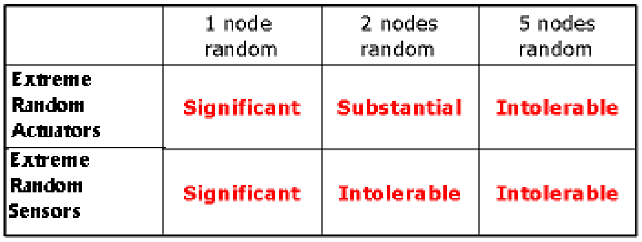


Figure 8. Effect of Extreme Random Faults on Boeing OEP



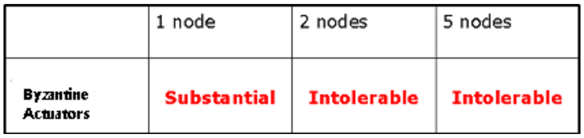
	1 node random	2 nodes random	5 nodes random
Extreme Random Actuators	Significant	Substantial	Intolerable
Extreme Random Sensors	Significant	Intolerable	Intolerable

Figure 9. Extreme Random Faults - Summary

Observation 4: Even a single sensor or actuator exhibiting extreme random behavior results in significant degradation in control performance. When 5 nodes exhibit extreme random behavior, it results in intolerable degradation.

3.5 Byzantine Faults

A Byzantine actuator is one that applies an arbitrary control input to the plant at all times. In reality they could even be the worst possible at all times. Byzantine faults represent a nondeterministic and potentially malicious behavior of the components. In the following set of experiments we introduce Byzantine faults by commanding the actuator to a maximum value in a direction opposite to that generated by the control law. These nodes were chosen arbitrarily for the experiments.



	1 node	2 nodes	5 nodes
Byzantine Actuators	Substantial	Intolerable	Intolerable

Figure 10. Effect of Byzantine Faults on Boeing OEP

Observation 5: Even a single Byzantine node causes substantial degradation in performance. More than one node exhibiting Byzantine behavior is intolerable to control performance.

3.6 Debonding Faults

To study the effect of debonding, the debonding was specified in terms of a final effectiveness percentage and the rate of debonding. We experimented with all the sensors and actuators in the system debonding to different levels from 10% to 90% and at different rates.

Observation 6: All the nodes debonding up to 50% is tolerable. The deterioration of performance is independent of the rate of debonding. There is however, a steep degradation at higher values of debonding. At 75%, the degradation in control performance is substantial.

3.7 Network Faults

In this section, we present the effect of delays introduced by the network, on the quality of vibroacoustic control. The Boeing OEP provides a hierarchical control to adapt the system to varying vibration modes (i.e., frequency modes) of the fairing that occurs in a launch. During such frequency mode changes, nodes are reassigned to a different group controller that control the specific frequency mode. These new group controllers report new parameters for control to their respective group members. In an ideal environment, the transition from an old group to a new group would be instantaneous. However, in the presence of network delays it is possible that a node has changed its group, however, the old group controller is not aware of this reconfiguration. It is also possible that a node has not yet changed to new group, however, the new group controller determines how the control should be modified to provide sufficient damping.

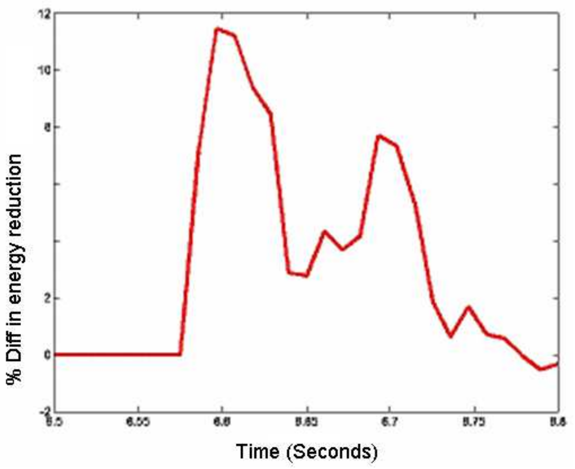


Figure 11. %Change in energy reduction during reconfiguration due to network delays

Implementation Boeing OEP does not support network delays in message communication. To introduce network delays in message communication, we reimplemented some of the functionalities in the Boeing OEP using threads[11]. We implemented the function that deletes (respectively, adds) a node from (respectively, to) a group controller using threads. The message delay is set to a maximum of 4 seconds. Fig. 11 shows the performance degradation of the control when delays are introduced. The metric used in this figure is the percent change in average energy reduction when delay is introduced.

Observation 7: 10-15% degradation in control performance is observed during reconfiguration. The effect lasts for about 0.2-0.3 seconds after reconfiguration is complete. Based on these observations, we anticipate that the transient degradation during the reconfiguration would be severe if mode changes occur too frequently.

3.8 Summary

The hierarchical control application in the Boeing OEP is found to be vulnerable to group controller crashes, sensors and actuators exhibiting random behavior and network delays. The crash of 1 or 2 group controllers results in significant degradation in control performance. Even a small fraction (5%) of sensors or actuators exhibiting random behavior within thrice their operating range results in substantial degradation in control performance. The presence of 2 or more Byzantine nodes is intolerable to control performance. In the presence of network delays 10-15% degradation is observed in control performance. This degradation is expected to worsen when system-id cycles occur more frequently.

4 Local On-Off Velocity Feedback Control Scheme

In the previous section we saw that nodes generating random forces and Byzantine nodes are substantially or intolerably detrimental to control performance. Moreover since control is hierarchical, group controller crashes and network delays degrade control performance. This leads us to design an alternative control scheme. In this section, we discuss a velocity feedback scheme that is purely local [3].

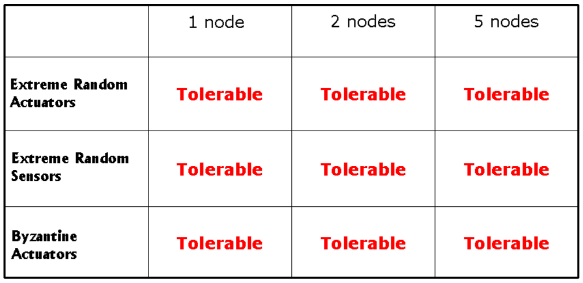
We assume individual sensor-actuator pairs to be colocated and apply the following local on-off output feedback control law to stabilize the system.

$$u_i = \alpha \times sign(y_i), \quad i = 1....m \tag{7}$$

where u_i is the control applied by the i^{th} actuator, y_i is the output measured by the i^{th} sensor and α is less than zero. Further u_i equals zero when y_i is zero. Thus a correct actuator can have 3 possible control values 0, $-\alpha$ and α. Control force is binary and is applied in the opposite direction of the sensor output which is typical of an *on-off* control scheme. The magnitude of this force can be chosen to vary. We choose $|\alpha|$, the magnitude of the actuator force, to be equal to the average actuator force in the Boeing control scheme. The smaller this value, the longer it takes to stabilize the system [10]. In this scheme there is no hierarchy of control and hence there is no system-id, group controllers or configurators. Control is purely local.

Experiment We first studied the performance of this scheme under no faults. In order to study the effect of uncertainties in the plant parameters, one of our experiments was to change the plant parameters by about 20% and observe control performance using the purely local scheme. We then studied the effect 1 extreme random node and 1 Byzantine node.

> Observation 8: The performance in terms of energy reduction ratio of the local output feedback control scheme is as good as the Boeing scheme. in the absence of faults. The performance remains tolerable even when system parameters are uncertain by a magnitude of +/- 20%. Since there is no hierarchy of control, the effect of node crashes remains local and network delays do not affect control performance. Control performance remains tolerable even in the presence of 1 sensor or actuator exhibiting extreme random behavior and in the presence of 1 Byzantine node.



	1 node	2 nodes	5 nodes
Extreme Random Actuators	Tolerable	Tolerable	Tolerable
Extreme Random Sensors	Tolerable	Tolerable	Tolerable
Byzantine Actuators	Tolerable	Tolerable	Tolerable

Figure 12. Effect of extreme random and Byzantine Faults on On-Off Control Scheme

Result The local on-off control scheme is found to be more fault-tolerant than the Boeing control scheme. This approach challenges the hierarchical control approach followed by the Boeing OEP. However, on-off control scheme is sensitive to sampling rate. At lower sampling frequencies like 2 KHz in the Boeing OEP, the sensors may be too slow in detecting the direction of the modal velocities and thereby resulting in incorrect direction of the actuator force. Hence sampling frequency has to be higher and so the energy spent in control is higher.

5 Conclusions

In this paper we evaluated the effect of potential faults on a hierarchical fairing control application designed by Boeing. Group controller crashes and extreme random faults are found to be catastrophic to control performance. Network delays also introduce degradation in control performance during system reconfiguration which becomes substantial in the presence of high frequency system-id cycles. We then designed an alternate control scheme and compared its performance with the Boeing OEP.

A purely local on-off control scheme based on velocity feedback is found to be tolerant to a rich class of faults. In this scheme there is no hierarchy of control and hence it does not suffer from group controller crashes and network faults. The scheme is also more tolerant to extreme random and Byzantine faults. The local on-off scheme however requires a higher sampling frequency and hence is more energy consuming.

A theoretical validation of the experimental results presented in this paper would be interesting. In this regard, design of reliable control schemes that maintain stabillity and performance in the presence of arbitrary component faults is a subject of ongoing work and a related paper [11] has been submitted to the same publication. Regarding further extensions to our work, we would like to consider other types of network faults including message loss, reordering and omissions.

REFERENCES

[1] Challenge problem description for network embedded software technology (nest). Boeing Tech Report, The Boeing Company, St. Louis, MO 63166, April 2002.

[2] L. Meirovitch. *Dynamics and Control of Structures*. John Wiley and Sons, New York, 1990.

[3] M. J. Balas. Direct velocity feedback control of large space structures. *Journal of Guidance and Control*, 2(3):252–253, 1979.

[4] D. Šiljak. *Decentralized Control of Complex Systems*. Academic Press, New York, 1991.

[5] M. Blanke, R. I. Zamanabadi, and Bogh. Fault tolerant control systems, a holistic view. *Control Engineering Practice*, 5(5):693–702, 1997.

[6] G. Karsai, G. Biswas, T. Pasternak, and S. Narasimhan. Fault-adaptive control: a CBS application. In *8th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems*, Washington DC, April 2001.

[7] C. Bonivento, A. Paoli, and L. Marconi. Fault-tolerant control for a ship propulsion system. In *European Control Conference*, Porto, Portugal, 2001.

[8] J. P. How. Local control design methodologies for a hierarchic control architecture. Master's thesis, Dept. of Aeronautics and Astronautics, MIT, Cambridge, 1990.

[9] J. P. How. Local control design methodologies for a hierarchic control architecture. *Journal of Guidance, Control and Dynamics*, 15(3):654–663, 1992.

[10] Y. M. Kim, A. Arora, and V. Kulathumani. Local distributed control of linear systems despite byzantine faults. OSU Tech Report OSU-CISRC-7/03-TR42, The Ohio State University, Columbus, Ohio 43210, July 2003.

[11] V. Kulathumani, Y. M. Kim, and A. Arora. Reliable control system design despite byzantine actuators. OSU Tech Report OSU-CISRC-3/05-TR13, The Ohio State University, Columbus, Ohio 43210, March 2005.