JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 22, No. 1, January–February 1999

Practical View of Redundancy Management Application and Theory

Stephen Osder*
Osder Associates, Scottsdale, Arizona 85252

A perspective is developed on how redundancy management techniques for flight-critical systems have matured since the earliest applications in the 1960s, driven largely by the introduction of more powerful computer resources. As this evolution occurred, the basic issues involving system architecture tradeoffs changed very little, although the hardware mechanizations of the earlier analog systems have been replaced largely with the software of the newer digital systems. These basic issues are reviewed and how they tend to be resolved in practical mechanizations is shown. The large body of literature on analytic redundancy theory developed since the 1970s is discussed in the context of its applicability to practical systems. It is shown that analytic redundancy has an important role in real-world systems but that it is not a replacement for physical redundancy, and its proper implementation requires that it be embedded within the physical redundancy structure of the system.

I. Introduction

REDUNDANCY management in flight-critical systems is concerned with fault detection, isolation, and reconfiguration (FDIR). This paper emphasizes equipment redundancy as the basis for all practical detection and reconfiguration, despite the often-repeated suggestions in many published papers that physical redundancy is either archaic, impractical, or unnecessary. The perspective offered herein is based on the author's personal experience in the design of such systems, from the primitive analog systems in the late 1950s to the contemporary digital systems, and the intense scrutiny of every conceivable failure effect that is required to achieve their flight qualification. In the paper's title, the sequence of the words "application" and "theory" is intended to convey the fact that practice has preceded theory in this technology. An apparent disconnect between these two aspects of FDIR is discussed. Flight-critical systems with varying degrees of FDIR sophistication have been around for more than three decades, and their mechanizations always have been driven by safety and reliability requirements, which in turn are solved primarily with physical redundancy, augmented with built-in test (BIT) techniques of varying sophistication. It is shown that the prevalent theory associated with the nonphysical, or analytic, redundancy encounters fundamental safety barriers when one attempts practical applications of that theory in the real world. [There are many types of processes that can be described as analytic redundancy (AR) but, when this paper raises critical questions regarding the practicality of analytic redundancy theory, it specifically refers to the concept of estimating aircraft states from measurements of the aircraft's control vector.]

Many of the published papers on analytic redundancy contain obligatory, but misleading, statements such as "Hardware redundancy results in more costly, heavier, less practical and less potentially reliable systems than do various AR strategies."[1–4] Although there are successful applications of AR (such as that used in many tie-breaking algorithms), which achieve some reduction in hardware redundancy, the resulting hardware weight reduction is usually not very dramatic. As opposed to physical redundancy, which uses measurements from redundant elements of the system for detecting faults, AR is based on signals generated from a mathematical model of the system. In this paper, geometric redundancy, or measurement information derived from kinematic relationships, such as skewed sensors, is not viewed as AR, although the classification of AR in the literature often includes that category of applications. Fault detection in AR systems considered herein is accomplished by comparing actual measurements of system states with a priori information inherent in the mathematical model. These difference signals, or residuals, are analyzed by a variety of methods to produce failure decision actions. A common fallacy in demonstrations of AR techniques is that measurements made on the mathematical model are assumed to be immune to faults. In discussing this issue, it is demonstrated that faults in measurements made on the math model are as probable as the other system faults that one attempts to detect and diagnose, and indeed, these two classes of faults are coupled in a manner that often is neglected in much of the theoretical literature on this subject. Another weakness in AR theory relates to the validity of the analytical models used. These typically assume linear, time-invariant aircraft plants, where measurements of surface positions are used to define the control vector. Such perturbation models have difficulty in coping with the large trim positions of those surfaces. The subject of model validity is beyond the scope of this paper, although it is noted that many of the published papers on AR theory neglect to define how a perturbation model must transition through an aircraft's full flight regime.

This paper provides a perspective of how redundancy management techniques have matured with the introduction of more powerful computer resources. That maturization involves an evolution from analog mechanizations of the 1960s and early 1970s to the digital systems, starting in the mid-1970s and continuing through the 1990s. Early redundancy management applications appeared in mechanizations of civil autoland systems,[5–9] and military aircraft command augmentation systems. In the 1970s, these applications transitioned to digital versions,[10,11] and fly-by-wire systems became operational. Currently, most new civil transport and military aircraft have adopted fly-by-wire technology, for example, Boeing 777, Airbus A-320 and variations, C-17, F-117, F-22, B-2, V-22, and the Comanche Helicopter. As this evolution occurred, basic issues involved in architecture tradeoffs remained essentially the same as they were in the 1960s. These issues are reviewed. It is also shown that AR has an important role but its proper implementation requires that it be embedded in the system's physical redundancy structure.

II. Redundancy Management Evolution from Analog to Digital Systems

A. Origins

System safety always has been the primary motivation for redundant systems in flight-critical applications. Early autopilot designs encountered the tradeoff between autopilot control force authority needed to give adequate performance and the pilot's ability to overcome the autopilot's servo actuators in the event of a failure. In the early 1950s, the autopilot for the B-52 bomber resolved this tradeoff by giving the elevator servo higher force capability than

Received July 13, 1998; revision received Oct. 8, 1998; accepted for publication Oct. 9, 1998.

*Consultant, Guidance and Control, Box 9825. Associate Fellow AIAA.

the pilot could overpower, but also added an independent safety monitor that would disengage the autopilot if the aircraft's normal acceleration exceeded a threshold consistent with approximate values that the pilot could expect. It was soon apparent that the most accurate safety monitors were complete and independent duplications of all sensing, computing, and actuation elements. That complete duplication became known as the dual, fail-passive configuration, and became popular in many military and civil transport flight-control applications. It also illustrated a new dilemma for flight-control designers: System safety is not the same as system reliability! Increasing safety deteriorates reliability! In the dual, fail-passive architecture, we increase the number of components that can fail by more than a factor of two and, hence, cause the probability of system failure (shutdown) to more than double, but the added hardware prevented an undesirable, and possibly hazardous, failure transient.

When reliability demands prevailed with a need to continue operation of the automatic system, even in the event of a failure, the requirement for the fail-operative system appeared. Initially, that requirement arose during early development of all-weather landing systems, and of large authority command augmentation systems for reduced-stability airframes.[12] By the late 1960s and early 1970s, triplex fail-operative (747 autoland, F-15 and F-111 command augmentation, etc.), dual-dual fail-operative (DC-10 and L-1011 autoland), and quadruplex double fail-operative (experimental fly-by-wire demonstrator) architectures had been introduced. These early systems were analog, but digital versions were also under development at that time, the U.S. SST program, for example. To achieve comprehensive fault detection and isolation in the analog mechanizations, complexity grew enormously.[13] That complexity was associated with monitor circuitry, voter circuits, and BIT hardware. For example, if an internal element of a voter circuit experienced a hardover failure, the remainder of that circuit suppressed that fault and the voter continued to operate normally. However, if a faulty input signal entered that voter, the voter would select the faulty signal and reject the good signals. Additional self-test circuitry therefore had to be added to exercise the voter's internal circuits and thereby expose latent hardware faults. It is apparent that when digital computers implement redundancy management in software, the complexity growth stops because we no longer need additional hardware to perform the monitoring functions. This great advantage of the digital computer (in addition to ability to accommodate more sophisticated control algorithms), led to the replacement of analog flight-control systems by digital systems on all new aircraft of the 1980s era. However, as is well known, the complexity problem migrated from hardware to software as the digital systems with their dramatically improved BIT capability became operational.

B. Architecture Issues

Selection of a redundancy architecture for a given application is dependent upon the aircraft configuration, especially the control-surface distribution. Also, quantitative specifications for survivability in a given mission time will drive the requirements for redundancy of the electrical and hydraulic power systems, which, in turn, exercise key influences on actuator architectures and their interfaces with flight-control computers. Despite strong advocacies for different architectural approaches over the years, a generic optimum does not exist. We illustrate key issues with reference to a generic triplex configuration illustrated in Fig. 1, noting that these issues are inherent in all competing architectures.

Figure 1 can represent either an analog or a digital mechanization, because digital successors to analog systems of the 1960s and 1970s are architecturally equivalent despite profound differences in mechanization. Also, it must be understood that this is an abstract diagram that does not represent any real system that typically would be far more complex in regard to interfacing computers with the multiplicity of control surfaces and their associated actuators. This will become very apparent subsequently when we illustrate the Boeing 777 aircraft's computer–actuator interfaces. However, the important issues can be illustrated with reference to Fig. 1 in the following subsections.

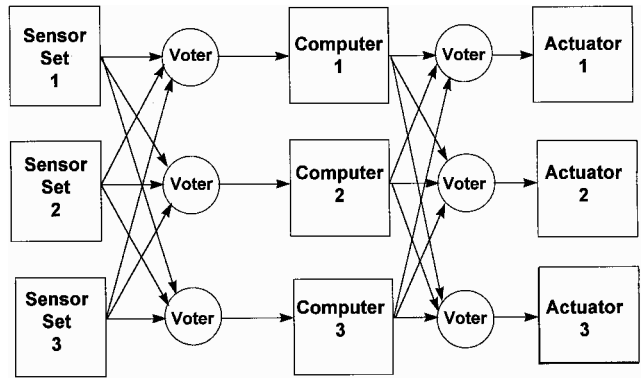

Fig. 1 Generic triplex architecture for flight-critical systems.

1. Group Redundancy

This refers to the grouping of all redundant sensors so that they are inputted to each computer and the grouping of all computer outputs so that all computers drive all actuators. Motivations for this grouping are the improvement in monitoring and fault detection with regard to false alarms and to survivability in the presence of failures. If we did not group, and sensor set 1 was only connected to computer 1, sensor set 2 to computer 2, and so on, we would be faced with a large tolerance buildup in the computer output commands to the actuators. If the actuator inputs also were isolated to commands computed by their respective channels, then the tolerance buildup would grow further and, indeed, for typical sensor and actuator tolerances, triple redundant actuators would be in a continuous hardover force fight. That type of force fight would be indistinguishable from the effects of true failure conditions, so that fault detection and isolation may not be possible. During the analog era, techniques such as channel equalization, also referred to as branch balancing, were developed to accommodate this tolerance phenomenon. The most important contribution to solving this tolerance problem was the voter circuitry introduced during the 1960s. They accepted multiple inputs and, in the case of the triplex voter, outputted the midvalue of the three inputs. Quadruplex voters usually were set to output the lower magnitude of the two middle values. As noted earlier, these circuits had nasty failure modes that required considerable complexity to overcome. With digital systems, voting algorithms in the software eliminated all of the voter circuit complexity but the issue of how to interconnect the redundant channels remained.

The other innovation of grouping involves the potential for increasing survivability, as illustrated by the equation for the probability of total system failure P_F, given the number of channels and the number of allowable failures:

$$P_F\left(\frac{r}{n}\right) = \sum_{x=r+1}^{n} \frac{n!}{x!(n-x)!} Q^x P^{(n-x)} \tag{1}$$

where $r =$ number of allowable failures for success ($r = 2$ for triplex channels, $r = 3$ for quad channels), $n =$ number of channels, $P =$ probability of individual channel success, and $Q =$ probability of individual channel failure.

Consider a quad channel case, with and without group redundancy. The probability of failure for a sensor set Q_S, for a computer channel Q_C, and for an actuator channel Q_A are related to their respective failure rates λ, where $Q_S = \lambda_S t$, $Q_C = \lambda_C t$, and $Q_A = \lambda_A t$ and $t =$ time:

For nongroup redundancy,

$$P_F = (Q_S + Q_C + Q_A)^4 \tag{2}$$

and for group redundancy,

$$P_F = (Q_S)^4 + (Q_C)^4 + (Q_A)^4 \tag{3}$$

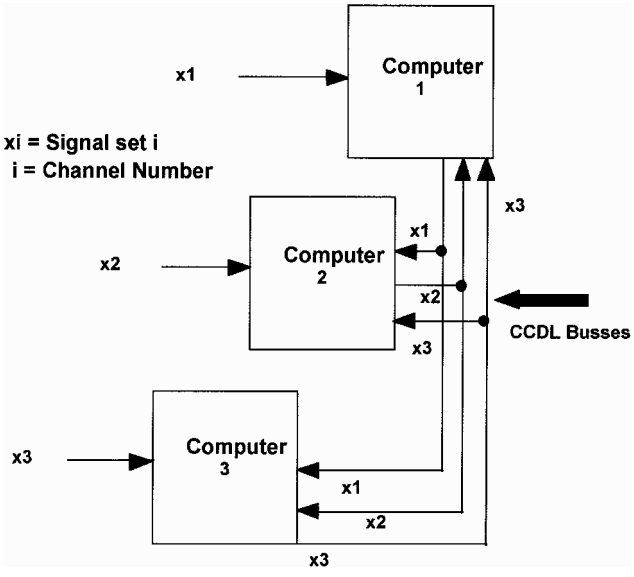

Fig. 2 Grouping sensors via CCDLs.

As a numerical example, if $Q_S = Q_C = Q_A = 10^{-2}$, then P_F for nongroup redundancy $= 81 \times 10^{-8}$, whereas, for group redundancy, P_F is reduced to 4×10^{-8} or a factor-of-20 improvement.

2. Mechanization of Voters, from Analog Circuits to Cross-Channel Data Links (CCDLs)

If we work out the details of a sensor-set interface to a computer, we find that the large number of sensor measurements used in a contemporary flight-control system begins to threaten the physical pin limitations associated with the box connectors. This type of problem usually occurred in the analog systems of the 1970s, but the arrival of digital systems provided several methods for alleviating the problem. If all of the sensor data would arrive at the computer on a serial digital bus, this problem would be solved, but that type of bus solution has been elusive. For example, if a simple pressure transducer, or a simple switch closure status, had to communicate with three computers via command–response data buses, it would require three independent remote terminals embedded into the transducer or switch, and this would increase size, weight, and cost unacceptably. In more recent systems, area multiplexing devices gather many of these miscellaneous data sources into a common terminal and interface with the computers via serial data buses. Regardless of whether such multiplexing is used, or individual measurements are communicated to the computers, the interface with the computers tends to be channelized, and the desired group redundancy is accomplished via high-speed data exchanges between the computers. That exchange is via serial data paths that connect all of the computers, but the integrity of those data paths is extremely critical and they must have appropriate fault detection and reconfiguration. This is shown in Fig. 2.

The communicating device is referred to as a CCDL. As shown, sensor set 1 is only connected to computer 1, but that computer transmits the values it found in set 1 to computers 2 and 3. Likewise, computers 2 and 3 transmit their sensor values to the other two computers. It is important that each computer use identical information from all three sensor sets at the same time. This requires that the CCDLs use a protocol that ensures some means of time synchronization for the computational frames in all three computers. Frame synchronization is the most common approach to implementing redundant computational channels that vote input and output information, although it is possible to design a system with 100% isolation of each computer's computational-frame timing. However, if this is not done with appropriate mode sequencing logic and channel equalization, it can lead to unexpected shutdowns (as in the case of early F-16 AFTI flight tests).[14] Solutions to such problems have been described in the literature.[15] In general, contemporary flight-control systems that use dissimilar hardware and software (because of a presumed vulnerability to generic faults) require asynchronous operation of redundant channels and therefore need to use these additional logical devices in their software.

3. Channel Symmetry

If sensors, computers, and actuators were perfectly symmetrical, system design would be greatly simplified. Figure 1 shows an ideal configuration in that the number of sensor sets, computers, and actuators are equal. Such symmetry rarely occurs in practice and certainly never occurs in large aircraft with many primary and secondary control surfaces. More common situations involve triplex computers with dual, triplex, or quadruplex sensors, and possibly dual actuators. In the analog era, one of the advantages of voter circuits was to resolve such asymmetry problems. For example, consider two cases. First is the variation of Fig. 1 where there are only two sensor sets or, more likely, one or more sensor types that are implemented with dual rather than triplex redundancy. In the second case, the triplex actuators are replaced with dual units. For the case of dual sensors, consider Fig. 1 without the third sensor set. All three computers see the two sensors, and the voting circuits of the analog era reverted to averaging rather than midvalue selection when one of the three inputs was removed. In contemporary digital mechanizations, the input processing software changes selection algorithms in an analogous manner. Thus, the voting function inherently solves the asymmetry of going from a lower level of redundancy to a higher level. Going from triplex to dual, however, as in the output technique for case 2, leads to some complication.

From Fig. 1, if we delete actuator 3, we see that the voter inherently connects those two actuators to all three computers but we are now confronted with a partitioning issue. Is the output voter resident at the computer or at the actuator? If it remains in the computer, the traditional location of early systems, we will require that computer 3 no longer connect to an actuator. This looks like a simple solution because it appears that we must only delete the wires between computer box 3 and the actuators in the aircraft wiring. This simplistic view forgets that each computer typically contains software for monitoring the response of its associated actuator channel, and for comparing its actuator response information with data about the other channels, as obtained via the CCDLs. We could accommodate this situation by activating a different output processing module for channel 3 but this type of software asymmetry is considered to be precarious, and to be avoided if possible. Asymmetric software between redundant computers opens the door to many vulnerabilities associated with maintaining frame or subframe synchronization, especially during failure detection and recovery from transient events. Such asymmetry in the software can be avoided if the output voting occurs at the actuator. Because the related servo electronics and servo monitoring functions are considerably more complex than the voter functions, and because the voters and monitors are now part of the software, this alternative requires that significant processing resources be resident at the actuators. Contemporary flight-control systems refer to this configuration as the smart actuator,[16] and by the 1990s, it has become a common approach to system partitioning in order to reduce cabling weight. Early attempts to locate the required electronic modules on actuator units led to concerns about reliability deterioration due to the extreme environment. A more common solution is to locate the required electronic-computer units in areas where they can service many actuators. Thus, we could expect to find such actuator-control-electronics (ACE) units in the left wing, right wing, and tail actuator areas.

Figure 3 shows a conceptual layout of a system that uses such ACE boxes. Each of the triplex computers drives each actuator electronics unit via its own serial bus. The actuator electronics unit votes the commands from each computer. There are many other types of arrangements where such voting is not needed, and where fault tolerance can be obtained by having multiple control surfaces. When there are many control surfaces, control to each surface can be implemented with dual rather than triplex computer interfaces. Figure 4 is an architecture diagram for the control distribution in the Boeing 777 fly-by-wire flight controls. It uses quad ACE units to drive the multitude of actuators shown. Critical factors in the design of such architectures involve the distribution of redundant hydraulic and electric

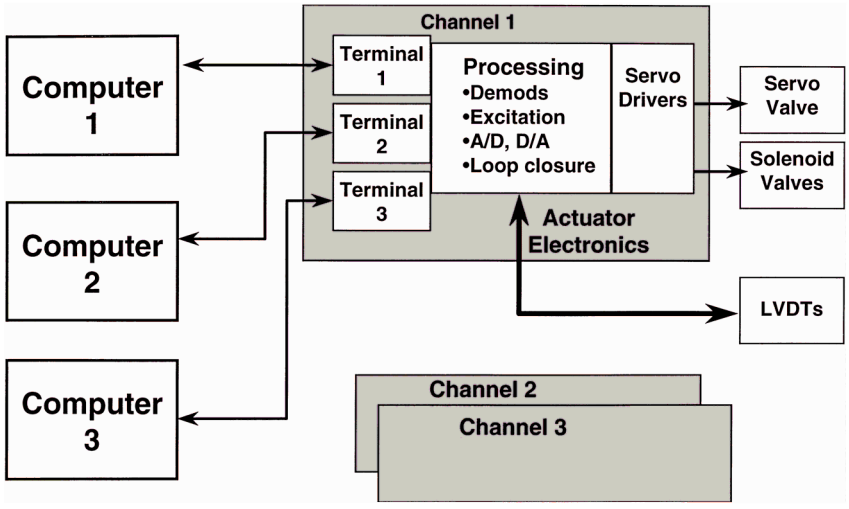

Fig. 3 Generic functional architecture for using remote actuator electronic units.

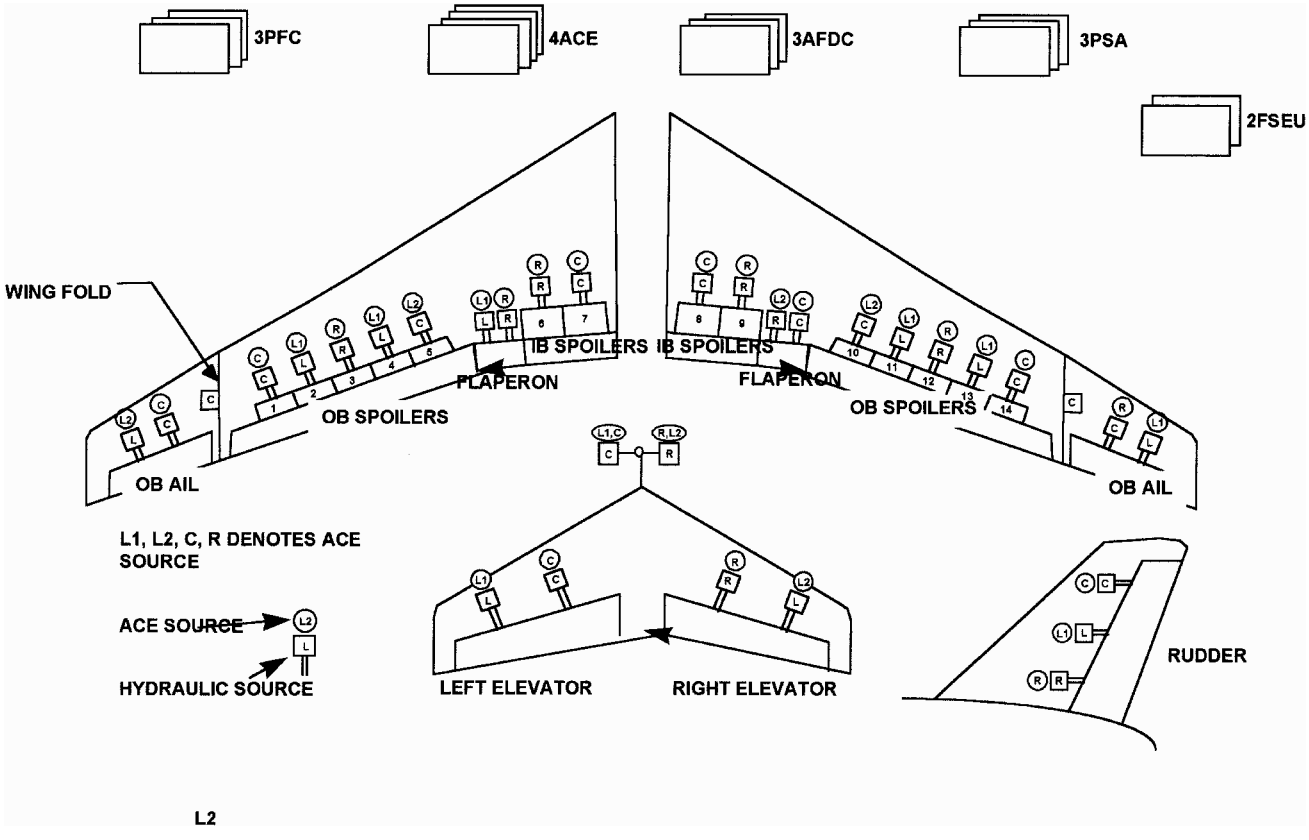

Fig. 4 Boeing 777 flight-control system architecture: control distribution.

supplies. In the Boeing 777, triplex electrical conditioning units provide fail operational power, triplex primary flight-control computers (PFC) provide the basic handling qualities (the ACEs also include augmentation capability), the autopilot flight director units (AFDC) provide the traditional autoland and auto-guidance functions, the flap-slat electronics units (FSEU are for secondary controls, and the PSAs provide fail-operative conditioned power. Note however, that multiple actuators sum at the surfaces so that only one ACE drives one actuator, rather than triplex channel drivers into a single actuator as shown in Fig. 5, an expansion of Fig. 4. In an aircraft that can distribute actuators to multiple redundant surfaces, it is possible to simplify the actuator drive interface. However, in aircraft such as helicopters with a single swashplate mechanism used to obtain pitch, roll, and vertical control, this type of actuator interface simplification is not applicable.

III. Sensor, Computer, and Actuator Fault Detection and Isolation

A. Actuator Monitoring

It is important that researchers in FDIR theory appreciate how actuators can fail. A multiplicity of failures within an actuator system can occur, and the aircraft will continue to operate normally. A primary requirement for fielding flight-control systems is that actuator failures can occur but no significant disturbances of aircraft attitude are allowed as a result of such failures. Processes are at work to identify and isolate the effects of, those failures, but those processes involve the details of actuator system mechanization. Typical details are found in a functional breakdown of the various elements contained in Fig. 3. In that figure, each computer channel output, via its command bus to the ACE or smart actuator terminal, may already be a voted output based on receiving the computed output command of

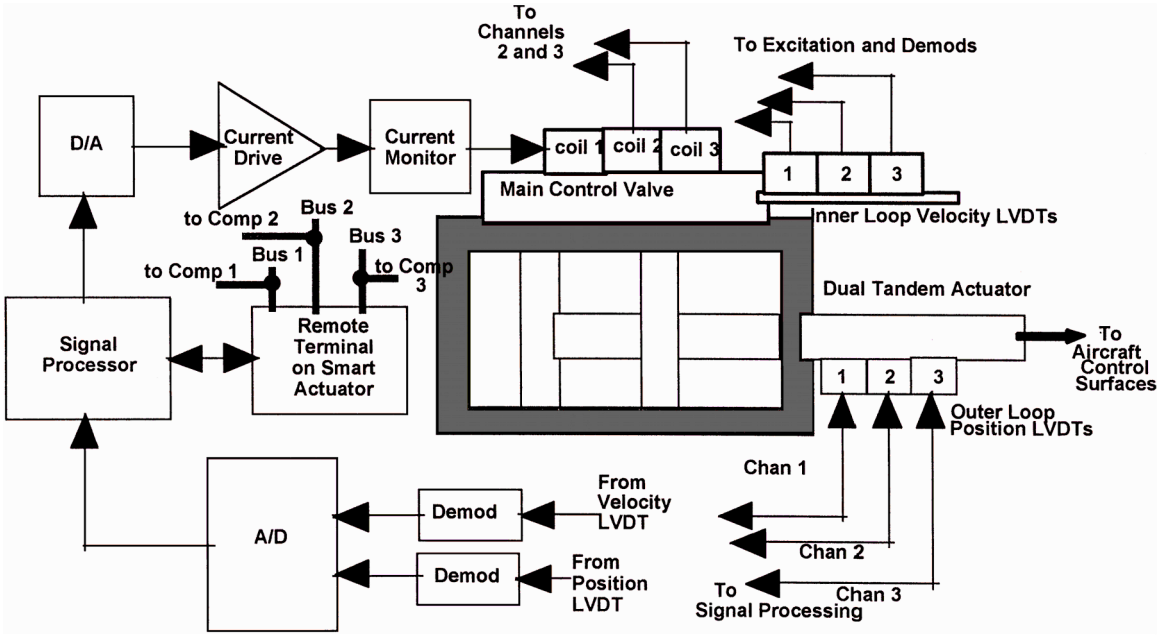

Fig. 5 One channel of a typical interface to a triplex actuator.

the other channels via the CCDLs. (This type of output voting typically is avoided because it tends to introduce added transport lag. In such a case, the comparison with the other channel outputs is delayed until the next iteration cycle.) Because each computer channel already will have computed its output using the same sensor estimates or measurements as the other channels, the only time that outputs on the three command buses of Fig. 3 will not be identical will be for the case of a computer failure. That failure typically is detected in about one iteration (5 to 20 ms), and its transient effect will be suppressed in the redundant actuator channels. Details of that redundancy are illustrated in Fig. 5. However, even with each servo channel operating on identical commands, tolerance buildup in the position feedback transducers can cause saturating-type current fights at the valve. Thus, in many cases, channel equalization is needed to allow fault detection based on current voting at the valve coils. This implies the need for CCDLs at the actuator electronics, or data transmission back to the computers where the branch balancing can occur.

Shown in Fig. 5 are the details of channel 1, indicating how the remaining two channels interface with the servo valves and with the position transducers. Figures 3 and 5 illustrate what may be called a brute-force redundancy architecture, in that any computer can drive all three actuator channels from any of the three ACE boxes. There are nine remote terminals, three in each ACE box, although box 1 drives valve control coil 1 only, box 2 drives coil 2 only, and so on. Thus, if two ACE boxes have failed, and only box 3 has survived, it will vote all three computer commands and drive the actuator through control coil 3. In addition, each ACE box must contain three independent power supplies to allow isolation and autonomy of all three of its bus terminals. However, a single power supply is used by each ACE box to excite, via its audio frequency oscillators, one of the three position transducers (LVDTs) on both the ram and the control valve (inner- and outer-loop positions). The channelized redundant processing of position measurements is discussed later when we consider the problems associated with implementing AR. Note that the dual tandem actuator interfaces are typical of contemporary fly-by-wire systems of the late 1980s.[17,18] Moreover, in those actuators, each control valve and associated drive coil is typically dual to improve fault detection and survivability. Also, there are several simpler variations of the redundancy architecture illustrated, most notable being more channelization in each ACE box, where each box communicates only with its associated computer channel, resulting in three bus terminals rather than nine.

A fundamental point illustrated by Fig. 5 is that if there is any servo channel failure, including hardover failures, the output position transducers will continue to hold their correct values. Thus, actuator monitoring cannot be accomplished by comparing actuator position with the commanded position. Monitoring is accomplished by observing the currents in the control coils and, in some cases, by observing internal valve positions. The fault detection technique therefore must be tailored to the specific actuator mechanism and its internal redundancy mechanisms. Fault detection is needed to isolate a failure and thereby to improve the observability and detection of subsequent faults.

B. Sensor Monitoring

Flight-control computers generally contain a front-end I/O processing subsystem to manage the redundant sensor inputs. This includes the signal-processing hardware such as analog signal demodulators, A/D converters, and bus terminals with serial-to-parallel data conversions. In early digital fly-by-wire designs, the sensor fault detection algorithms were resident in the primary processor, but with processing costs dropping dramatically in the past decade, newer design trends include a separate front-end processor for sensor redundancy management functions. With sufficient computer throughput, they provide exhaustive examination of all sensor anomalies, to the point that all faulty measurement is removed from use by the control laws, and detailed diagnostics regarding the faults are available immediately, or for postflight examination. The following examples of monitoring algorithms are derived from the MD-80 system which was designed circa 1977 (Ref. 19), but its techniques are still relevant to more contemporary application. The general concept is illustrated in Fig. 6, which emphasizes the use of an update screen to prevent suspect measurements from reaching control laws. In Fig. 6, x_i refers to the value of sensor i, subscripts A and B refer to dual sensors A and B. A screen exists in the data path from the measurement functions to the use of those measurement values by the control laws. A measurement for iteration n will not be transmitted downstream if anomalous value is discerned by a variety of processes. First, the sensor validity status is examined and, if the "valids" generated within that sensor disappear, that measurement will be screened out of the control computations, regardless of voter or comparator algorithm decisions. In many flight-control sensors, the validity discrete picks up 95 to 99% of possible failures. For example, about 95% of probable failures of the (now almost obsolete) rotating wheel rate gyro are detected by that instrument's self-contained wheel speed rotation detector. That detector reveals the consequences of failed bearings, even before the motor comes to a halt, but decreases in wheel speed also will be detected by voting or comparison monitors, because lower wheel speed results in a reduction in scale factor. A comprehensively monitored system

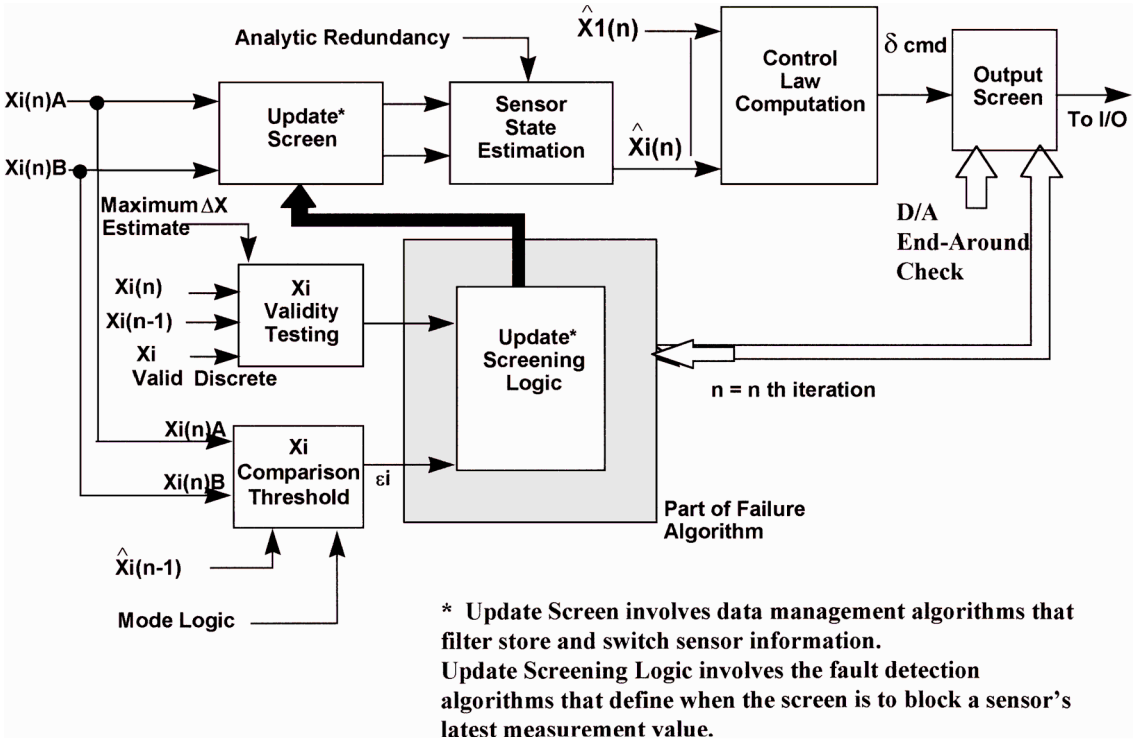

Fig. 6 Monitoring concept for screening faulty measurements.

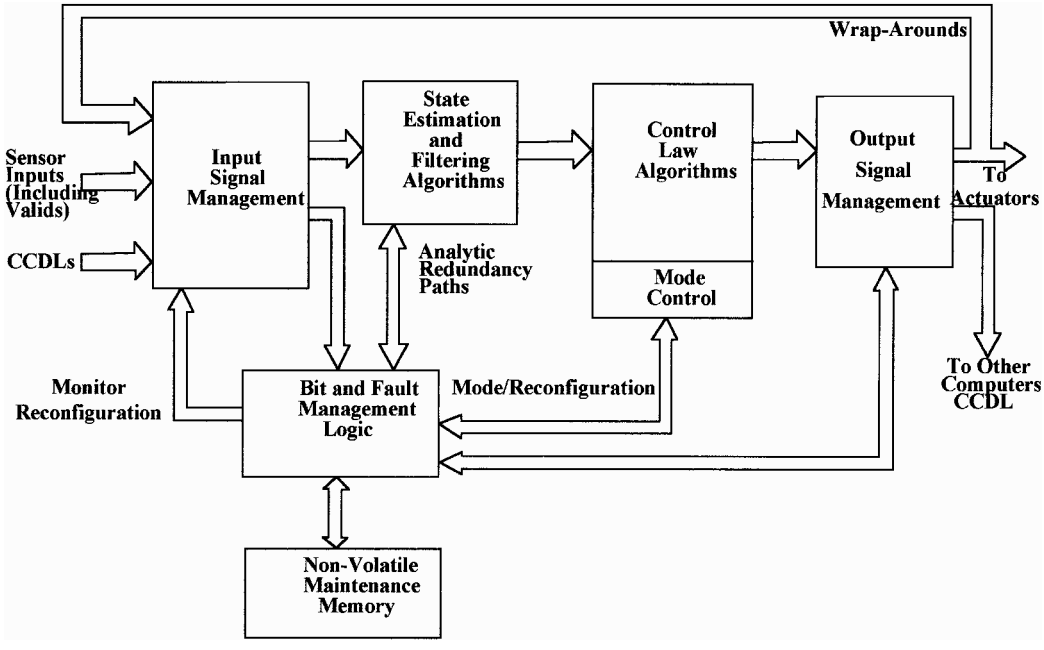

Fig. 7 Relation of monitoring functions to other flight-control software functions.

such as that implied by Fig. 6 uses all of the available intelligence to make the failure decisions. Likewise, ring laser gyro failures are usually in the lasing function and associated electronics, which are self-monitored with near 100% comprehensiveness. Also shown in the validity testing block of Fig. 6 is a reasonableness constraint on the allowable incremental change in a measurement over one iteration time T. For example, if the measurement is pitch rate q, the maximum allowable change over one iteration $(x_n - x_{(n-1)})$ is $(\dot{q})_{\max} T$, where the maximum allowable pitch angular acceleration is based on the specific aircraft capability. This type of function becomes useful only when multiple failures have occurred and a system has reverted to single-string operation. This is also the time when AR becomes most useful.

A more detailed description of how monitoring functions relate to the other flight-control computer software is given in Fig. 7. Note that the input signal management and output signal management can be performed in a separate front-end I/O processor. This figure shows how embedding fault management logic within the overall flight-control computation process leads to a natural flow from input to output, with that logic affecting mode control as well as the screening of faulty data. (The update screen of Fig. 6 is contained within the input signal management block). All output signals (hardware) are fed back to the inputs, where they exercise the conversion hardware, and are compared with the commands generated by the software in the control law and output signal management blocks. This process is referred to as end-arounds, and it actually permits diagnostics to

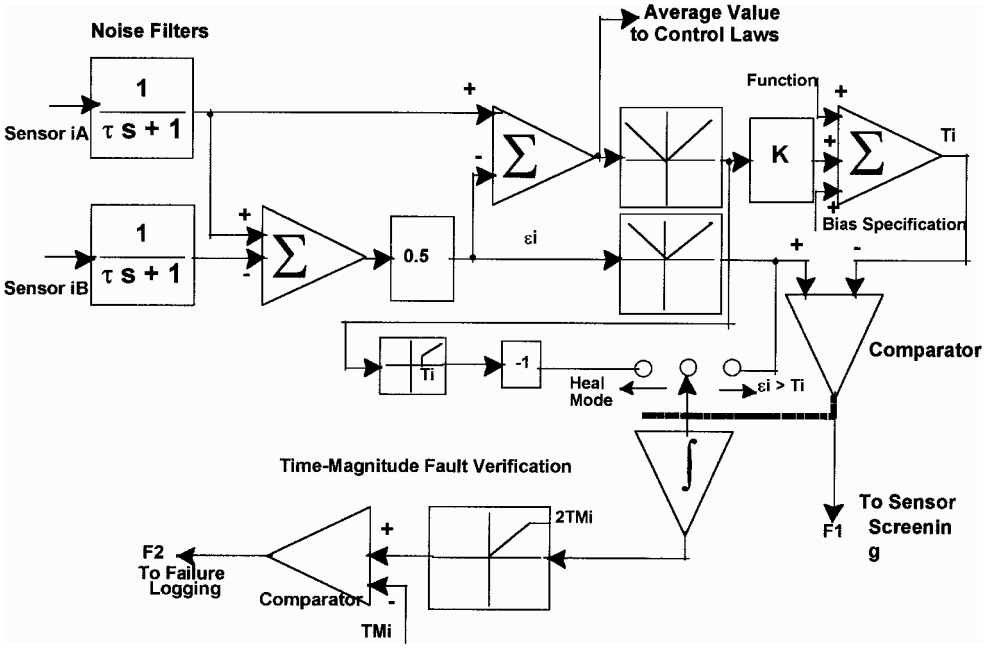

Fig. 8 Details of fault detection/isolation in a dual sensor set.

detailed functions within a subassembly element of the system hardware. Detected faults are stored in nonvolatile memory, where they can be recalled with a snapshot of significant aircraft states existing at the time of fault detection. In general, failures of all hardware elements associated with I/O processing can be detected and diagnosed by system software, without resorting to hardware redundancy.

C. Sensor Fault Decision Algorithms

A key issue relates to definition of failure thresholds that reflect calibration tolerances, and environmental effects on component specifications. There are some contemporary products that continue the practices derived from the obsolete analog designs that used fixed, high-magnitude thresholds, selected so that the possibility of a false alarm is negligible. Many types of statistical decision criteria have been developed and used successfully, often in the context of applying AR techniques.[20,21] A practical application of simple statistical principles is described later to illustrate a state-of-the-art example dating from the mid-1970s (Ref. 19). It also shows how AR is applied effectively in the context of a realistic redundancy architecture. The example is for dual sensors, where the estimate forwarded to the control laws is the average of the two measurements, with appropriate noise filtering. These dual sensors actually may be the reversion following a hard failure of the third sensor of a triplex set. It is instructive to focus on the dual sensors because they provide the best scenario for a practical application of AR. Figure 8 illustrates the processing associated with fault detection, possible automatic healing, and reconfiguration. Although it shows a single sensor pair, it should be understood that, in a complete system mechanization, an orderly process sequences through a large array of input data, performing the same type of operation on each measurement.

Figure 8 illustrates the processing software used to enhance measurement integrity for the A and B channels of sensor I. Front-end filtering is shown as first order, but these antialiasing and noise reduction techniques may be analog as well as digital, and are usually of greater than first order. The signal is partitioned into an average value, $(A + B)/2$, and difference, $\varepsilon_i = (A - B)$. If ε_i exceeds a computed threshold T_i, then a failure state, $F1$ is indicated. Existence of $F1$ will cause the sensor screen to close, thereby sending the control laws the prior estimate of measurement i, propagated one time frame forward. A fault has not yet been declared. Two diagnostic actions now are initiated. First, the error ε_i is integrated to generate a time-magnitude product that measures the severity of the error. If the time-magnitude product of the error exceeds a second threshold TM_i, then failure state $F2$ is declared. This recognizes that a fault has been identified and that sensor reconfiguration is needed. We do not yet know whether sensor A or B is at fault, unless sensor valids have provided that isolation. (If sensor B's valid discrete had disappeared, regardless of whether the fault states $F1$ or $F2$ had been declared, the bad sensor will have been removed from the data paths and sensor A will have become the sole remaining good sensor.) As soon as state $F1$ is declared, an AR routine will be initiated, if appropriate for that sensor. It activates comparisons between sensors i_A and i_B with the analytic estimate of state i. Analytic estimates of the aircraft state vector $\boldsymbol{x}$ can be derived from the linear approximations given by

$$\dot{\boldsymbol{x}} = A\boldsymbol{x} + B\boldsymbol{u}, \qquad \boldsymbol{x} = (sI - A)^{-1}B\boldsymbol{u} \tag{4}$$

where we measure the control vector $\boldsymbol{u}$ (surface deflections), and know a priori, at a given flight condition, the system matrix A and control effectiveness matrix B. For Eq. (4) to be useful, we must be able to accommodate measurement and process noise. The theoretical literature[2] has explored many techniques for estimating the desired states in the presence of these disturbances. However, it is difficult to find any appreciation in that literature that Eq. (4), when applied to an aircraft, is a perturbation equation about a trim reference point. Thus, if we measure surface deflections to represent our knowledge of the $\boldsymbol{u}$ vector, we must be able to account for the often-dominant magnitude of the trim terms, which must be removed from the measurement in order for Eq. (4) to be valid.

In most practical implementations of AR, it has not been necessary to solve the full state matrix equation, but we have been able to obtain the desired information via several shortcuts. For example, if the sensor in question is pitch rate, then pitch rate q is one component of the $\boldsymbol{x}$ vector. Because the discrepancy between a good and a failed measurement does not require a precise third reference for most failures, including the pitch-rate measurement, that failure can be resolved using relatively crude estimates of incremental normal acceleration N_Z, aircraft speed, and altitude

$$N_Z = Vq/(\tau s + 1) \qquad \text{or} \qquad q = (\tau s + 1)(N_z/V) \tag{5}$$

where τ is a function of V, altitude (or density), aircraft mass, and lift curve slope. Another analytic estimate of q that often is used[20] is a derivation of q in terms of Euler angle rates obtained from independent inertial system references:

$$q = \dot{\theta}\cos\phi + \dot{\psi}\sin\phi\cos\theta \tag{6}$$

where θ = pitch attitude, ϕ = roll attitude, and ψ = heading.

When fault state $F2$ is declared, reversion to the single good sensor will occur, based on the AR algorithm or the validity processing described in Fig. 6. The failure threshold of Fig. 8 is defined as

$$T_i = K/2(iA + iB) + \text{Bias} + \text{function}$$

where K = scale factor tolerance (often the 2σ to 3σ specification tolerance), Bias is the allowable bias tolerance in the sensor spec, and function is a deterministic error function that is used to accommodate systematic measurement errors. For example, if heading were derived from a two-gimbal directional gyro (now obsolete in most aircraft), the function would be the gimbal error resulting from the instantaneous pitch and roll attitudes. Typically, most flight-control sensors are calibrated so that the statistical tolerance distribution is normal, but truncated between 1.0σ and 2.0σ. Therefore, use of 3.0σ for the T_i computation is generally adequate for avoiding false alarms. Moreover, the $F1$ state that occurs at the T_i threshold is not a declaration of a fault. If ε_i is large, the fault $F2$ will be declared within one to two iteration cycles; but if ε is small or noisy, then the $F1$ declaration will screen its effect, and the $F2$ fault state will occur only if the error is persistent.

After a fault has been declared, and the sensor reconfiguration has occurred, the faulty sensor is allowed to heal. Obviously, we cannot assume that the sensor has healed if ε_i has returned to zero because many sensors fail to zero output. Healing is allowed to occur only if the average of A and B measurements exceeds a value such as T_i. If that occurs, and $\varepsilon_i < T_i$, then the time-magnitude integrator is allowed to decay, and if it falls below the TM_i threshold, then healing will have occurred. The maintenance memory retains the history of fault declarations, so that, if this healing process occurs often, there is a record of a marginal device that warrants investigation.

D. Processor Monitoring

When it required a large avionics rack to mechanize a computer, the processor was the dominant cost element, and there was a strong motivation to minimize hardware redundancy. Technology was developed to permit 100% self-monitoring of a computer, including its processor, without the use of redundant hardware.[19] However, the validation process for such systems is very costly, and the drop in processor costs, plus associated miniaturization, now permit system implementations using large amounts of redundant hardware. Typical contemporary applications use dual microprocessors, running in lock step, for each computer channel. These possible configurations are beyond the scope of this paper.

IV. System Reconfiguration

We thus far have avoided discussion of total system configuration following major loss of a control function. It has been shown that failures are isolated within a subsystem so that they will inherently reconfigure themselves, and the total system will continue to perform properly, or with only minor degradation. There are, however, catastrophic events that can disrupt entire systems so that the inherent reconfiguration strategies are no longer applicable. In military combat aircraft, the possibility of battle damage eliminating the availability of a primary control surface has stimulated interest in reconfiguration strategies that make use of other surfaces or control devices to compensate for that loss. Thus, if more than one surface is able to generate pitching or rolling moments on the aircraft, it is not difficult to visualize a solution that exploits these redundant moment-generating devices. A more dramatic catastrophic event has occurred in civil aviation, and that event has inspired much theoretical and practical work. It is the total loss of hydraulic power due to damage at the aircraft tail. The pitching-moment control using tail surfaces is therefore lost. Theoretical work often has approached this problem as one of fault detection, but the flight crew of the DC-10 involved in the well-known Sioux City, Iowa, accident was well aware of the fact that their tail controls were not operating. They attempted to recover the aircraft using moments produced by the engine thrust. There is a history of similar events, and the need for an orderly approach to moment control via the engines has been recognized for many years. A true practical solution was demonstrated in flight tests recently, where the existing flight-control computers were modified to incorporate the necessary aircraft recovery algorithms, and a DC-10 aircraft was landed successfully without any tail surface controls, thereby duplicating the Sioux City, Iowa, conditions.[22]

V. Redundancy Management Theory

For the past two decades, a large body of literature has been created on the subject of fault detection and isolation from the viewpoint of system identification and decision theory. Usually, that literature does not, in general, recognize the methodologies of failure mode and effects analysis (FMEA), which is the key ingredient of practical system design. Part of a system's certification or qualification is to show how every possible failure mode is accommodated by the system's redundancy management architecture and related fault detection devices. For example, Willsky,[2] in his excellent survey of this field, mentions that hardware voting of redundant sensors is limited because of common faults, such as common power supplies for two instruments. This type of defect is precisely precluded by any properly designed hardware redundancy, where avoiding common-mode (or single-point) failures, or coupled failures, are primary design considerations, with FMEAs used to verify that such design deficiencies do not exist. Unfortunately, this same type of scrutiny has not been applied to most AR concepts that appear in the literature. Indeed, common-mode failures are glaring deficiencies in many such papers that depend upon measurements of the control vector to estimate system states. This can be demonstrated by showing how a fault in an actuator can mistakenly be assigned to a sensor when failure-mode effects are not considered properly.

Patton and Chen[1] use the solid-line blocks of Fig. 9 to generalize the process of comparing the output as computed from the input control vector $\boldsymbol{u}$ with the output measurement. The difference, or residual, is examined by the decision process, to arrive at a fault diagnosis.

Estimation theory results in a variety of approaches to the residual generation function, whereas decision theory is concerned with defining thresholds beyond which a fault can be declared and diagnosed. The system problem is envisioned as three sequential elements: the actuator, the plant dynamics, and the sensors, each of which can have distinct failure vectors, designated F_A, F_P, and F_S, respectively. This view of the problem neglects the coupling between all of these elements, as shown by the dotted sections on Fig. 9. Plant dynamics affect actuator loading, and hence the actuator dynamic and static response. Measured $\boldsymbol{u}$ is a dominant feedback that defines actuator characteristics. Sensors couple back into the system via closed-loop feedbacks, and the plant dynamics are typically nonlinear where the A matrix elements of Eq. (4) are typically variable functions of the state vector $\boldsymbol{x}$. Let us illustrate these points by considering the characteristics of a reasonably accurate actuator model: The reference actuator has inner- and outer-loop feedbacks, as described in Fig. 5. It is representative of the AH-64 longitudinal cyclic control in that aircraft's backup fly-by-wire mode. The model includes valve nonlinearities, velocity saturation, and hard position stops. The failure to be examined is an open-position feedback from the output ram. In the real operational design, this failure is easily detectable by the existing monitors, but in AR theory, these hardware methods of observing failures are not relevant. We use a simple linear model of the helicopter dynamics, plus a 1.0-rad/s pitch-attitude control loop, as representative of a good pilot. Figure 10 shows the response to a large 1.0-s pulse disturbance, with and without the position feedback signal.

When the position feedback is opened, u_m of Fig. 9 is held at zero. Because that is the only available measurement of the longitudinal cyclic control's contribution to the $\boldsymbol{u}$ vector, an AR algorithm will estimate the aircraft state on the basis of an incorrect control vector. Because the actual pitch-attitude measurement will not agree with the estimated value of pitch change based on $u_i = 0$, a diagnostic system derived from the $\boldsymbol{u}$ measurement would conclude that the pitch-attitude measurement had failed.

We therefore have illustrated that the failure mode that caused the actuator fault also caused the AR algorithm to fail. That position-measurement failure mode changed the actuator's dynamics from that of a position servo to a velocity servo. In that failure condition, the added integration provided by the actuator caused the closed-loop pitch stabilization to become divergently unstable. The

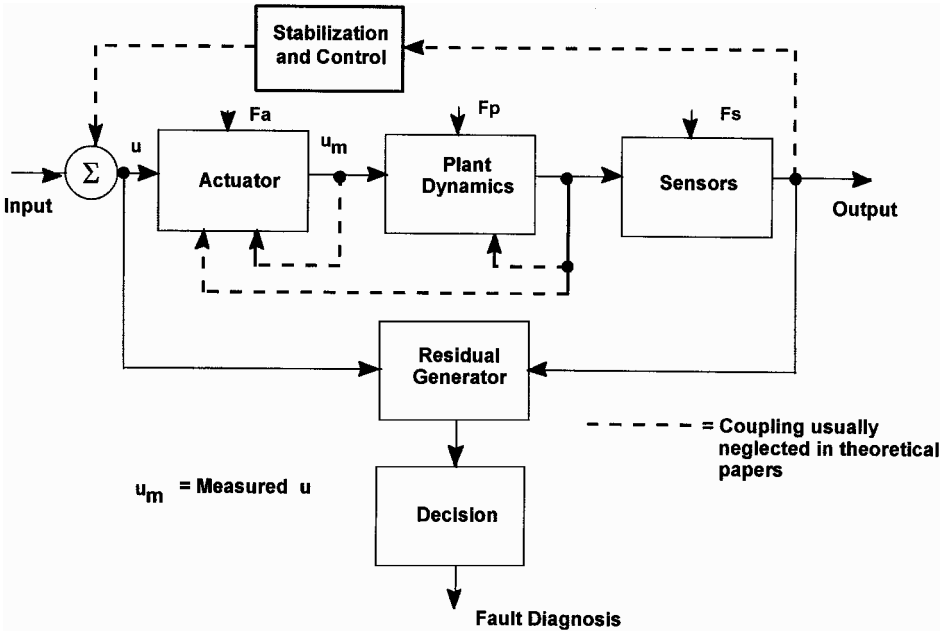

Fig. 9 Simplified view of the AR problem plus added complications often neglected.

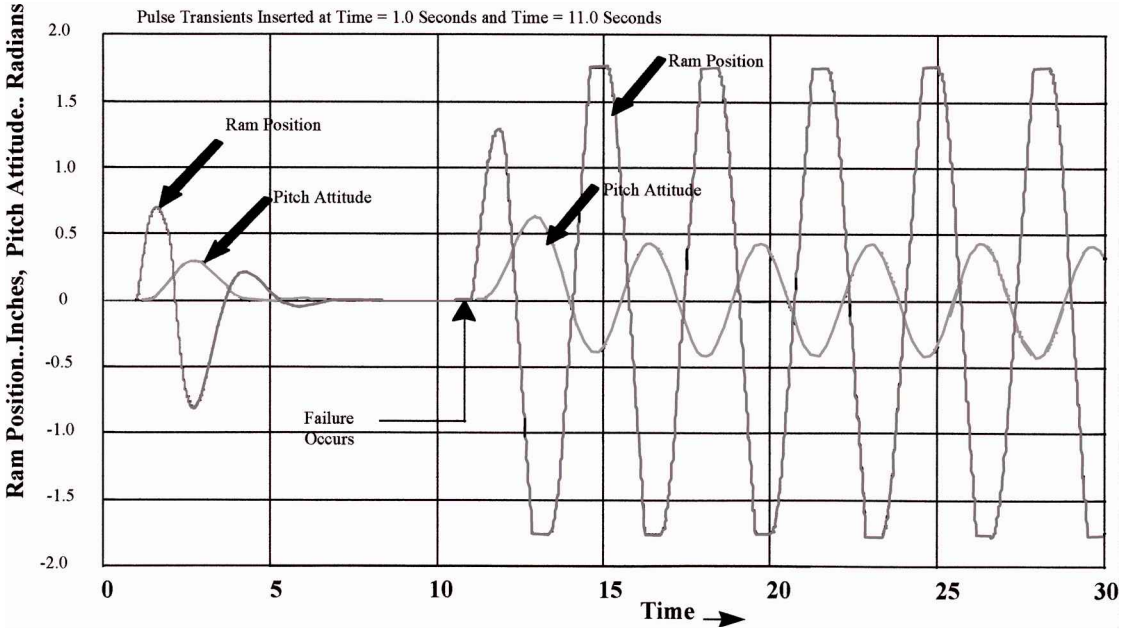

Fig. 10 Response of aircraft to pulse disturbance, with and without actuator open-position feedback failure.

divergence was bounded by position limits inherent in the actuator stops, and by velocity limits inherent in the valve's flow saturation. Note that the linear plant model is no longer even approximately valid in the range of attitude excursions shown, and the instability illustrated by this figure, if it really could occur, would be catastrophic. Many papers found in the published literature on theoretical AR show several radians of attitude and control-surface excursions during failure detection simulations, without regard to physical reality. In the real world, there are multiple measurements of the actuator's output position, as shown in Fig. 5. AR algorithms can be implemented safely only if the measurements associated with these algorithms are performed with sufficient redundancy and fault monitoring to meet the same survivability criteria as the primary elements of the flight-critical system.

VI. Conclusions

Physical redundancy is required to ensure safety of flight-critical control systems, and the state of the art has evolved using various types of triplex and quadruplex architectures. FMEA are needed to validate the safety of such redundant systems. AR does not substitute for physical redundancy but it can be a useful supplement to fault isolation, if implemented in a manner that properly exploits the physical redundancy.

References

[1]Patton, R. J., and Chen, J., "Robust Fault Detection of Jet Engine Sensor Systems Using Eigenstructure Assignment," *Journal of Guidance, Control, and Dynamics*, Vol. 15, No. 6, 1992, pp. 1491–1497.

[2]Willsky, A. S., "A Survey of Design Methods for Failure Detection in Dynamic Systems," *Automatica*, Vol. 12, 1976, pp. 601–611.

[3]Patton, R. J., Frank, P. M., and Clark, R. N. (eds.), *Fault Diagnosis in Dynamic Systems, Theory and Applications*, Control Engineering Series, Prentice–Hall, Englewood Cliffs, NJ.

[4]Frank, P. M., "Fault Diagnosis in Dynamic Systems, Using Analytical and Knowledge Based Redundancy . . . A Survey of New Results," *Automatica*, Vol. 26, No. 3, 1990.

[5]IATA 15th Technical Conference on All Weather Landing and Take-Off (Lucerne, Switzerland), Various Working Papers, 1963.

[6]Negre, Y., "All-Weather Automatic Landing Concepts and Operations: Review of Aerospatiale Experience," *Journal of Aircraft*, Vol. 14, No. 4, 1977, pp. 323–326.

[7]Throndsen, E. O., "Lockheed L-1011 Avionic Flight Control Redundant Systems," *Advanced Control Technology and Its Potential for Future Transport Aircraft Symposium*, pp. 779–802; also NASA TM X3409, Aug. 1976.

[8]"Preliminary Flight Control System Engineering Report," General Dynamics, Rept. FZM-12-874, Fort Worth, TX, Oct. 1964.

[9]Carlton, D. L., "F-16 Flight Control System Development," *Integrity in Electronic Flight Control Systems*, AGARDograph 224, April 1977.

[10]Osder, S. S., Mossman, D. C., and Devlin, B. T., "Flight Test of a Digital Guidance and Control System in a DC-10 Aircraft," *Journal of Aircraft*, Vol. 13, No. 9, 1976, pp. 676–686.

[11]Jarvis, C., Deets, D., and Szalai, K., "Description of Flight Test Results of the NASA F-8 Digital Fly-By-Wire Control System," NASA TN D-7843, 1975.

[12]Tomlinson, L. R., "Control System Design Considerations for a Longitudinally Unstable Supersonic Transport," *Journal of Aircraft*, Vol. 10, No. 10, 1973, pp. 594–601.

[13]Osder, S. S., "Chronological Overview of Past Avionic Flight Control System Reliability in Military and Commercial Operations," *Integrity in Electronic Flight Control Systems*, AGARDograph 224, April 1977.

[14]Youssey, W. J., Arabian, A. M., Schindler, T. M., and Whitmoyer, R. A., "AFTI/F-16 DFCS Development Summary—Redundancy Management System Design," NAECON, Dayton, OH, May 1983, pp. 1220–1226.

[15]Osder, S. S., "Generic Faults and Architecture Design Considerations in Flight-Critical Systems," *Journal of Guidance and Control*, Vol. 6, No. 2, 1983, pp. 65–71.

[16]McLaughlin, R. J., "Smart Actuator Development Program," Naval Air Development Center, Rept. NADC 90076-60, Warminster, PA, Aug. 1990.

[17]Harschburger, H. E., "Development of Redundant Flight Control Actuator Systems for the F/A-18 Strike Fighter," Society of Automotive Engineers Aerospace Congress, Long Beach, CA, Oct. 1983.

[18]Osder, S. S., "Digital Fly-By-Wire System for Advanced AH-64 Helicopters," AIAA Paper 88-3922, Oct. 1988.

[19]Osder, S. S., "DC-9-80 Digital Flight Guidance System Monitoring Techniques," *Journal of Guidance and Control*, Vol. 4, No. 1, 1981, pp. 41–49.

[20]Desai, M. N., Deckert, J. C., and Deyst, J. J., Jr., "Dual-Sensor Failure Identification Using Analytic Redundancy," *Journal of Guidance and Control*, Vol. 2, No. 3, 1979, pp. 213–220.

[21]Cunningham, T., et al., "Fault Tolerant Digital Flight Control with Analytic Redundancy," Air Force Flight Dynamics Lab., TR-77-25, May 1977.

[22]Burken, J. J., and Burcham, F. W., Jr., "Flight-Test Results of Propulsion-Only Emergency Control System on MD-11 Airplane," *Journal of Guidance, Control, and Dynamics*, Vol. 20, No. 5, 1997, pp. 980–987.