Table 2 Ratio of standard deviations at different distance

	7 Reflectors	13 Reflectors	Predicted
σ_1/σ_2	2.27	2.06	2.0
σ_2/σ_3	1.49	1.41	1.5
σ_1/σ_3	3.39	2.89	3.0

circles of radius 1, 2, or 3 ft with an additional asymmetric marker 1 ft beyond the circle on the y axis.

There was little change in error with the simulated change in attitude. The values of the error were calculated at each distance and plotted against the log of the distance. The changes over the distance from 6.25 to 200 ft were so small as to not affect the standard error by as much as 0.2 deg. Results from a typical set of runs are shown in Table 1.

The ratios between the standard deviations of the errors were predicted to be inversely proportional to the square roots of the numbers of markers, roughly 1.36. The simulation yielded 1.58, 1.38, and 1.31 for 1, 2, and 3 ft, respectively. The ratios of standard deviations at different distances were compared in Table 2 where σ_j represents the standard deviation for a circle of radius j.

To explore the effect of an increase in the accuracy of the scanning laser radar, in some of the runs it was assumed that each marker was read three times and the result averaged to get a more accurate position rading. The predicted ratio between the standard errors was 0.577. When run with 13 reflectors and a radius of 3 ft, this gave a standard error of 0.49 ± 0.04 deg. The ratio of the corresponding standard errors was 0.568.

Conclusions

A method for determining the relative attitude of two docking unmanned spacecraft has been developed. The method gives predictable and controllable accuracy. If implemented with scanning laser radar and cornercube reflectors or a congruent system, the markers should be arranged as far from their center as possible. This will increase the radius of the circle which will improve the accuracy of the readings in two ways: 1) the coordinates are known more accurately, and just as important, 2) F is increased along with $|Y|$. Increasing the denominators of the calculated quantities directly increases their accuracy. To get F large, it is also necessary that (X, Y) be small, so the markers, except for the asymmetric marker, should be set symmetric to both axes.

There was a great increase in accuracy when the radius, R, of the circle of markers was increased. This was true at all distances and all rotations tested. The standard deviation of the errors appears to decrease as $1/R$, as expected.

Acknowledgments

The problem was suggested by Stan Carroll of the Flight Dynamics Laboratory at Marshall Space Flight Center in Huntsville, Alabama. He provided invaluable support as the author's NASA-ASEE counterpart during the summer of 1979 when most of this work was done. During that period the author was a NASA-ASEE Fellow.

References

[1] Roberson, R.E., "Two Decades of Spacecraft Attitude Control," *Journal of Guidance and Control,* Vol. 2, No. 1, 1979, pp. 3-8.

[2] Shuster, M.D. and Oh, S.D., "Three Axis Attitude Determination from Vector Observations," *Journal of Guidance and Control,* Vol. 4, No. 1, 1981, p. 70.

[3] Shepperd, S.W., "Quarternion from Rotation Matrix," *Journal of Guidance and Control,* Vol. 1, No. 3, 1978, pp. 223-224.

[4] Hoel, P.G., *Introduction to Mathematical Statistics,* 3rd edition, John Wiley and Sons, New York, 1962.

[5] Ostle, B. and Mensing, R.W., *Statistics in Research,* 3rd edition, Iowa State University Press, Ames, Iowa, 1975.

A Core Software Concept for Integrated Control

Ellis F. Hitt,* Michael Kluse,†
and Robert Broderson‡
Battelle Columbus Laboratories, Columbus, Ohio

Introduction

INTEGRATED control[1,2] makes use of digital computer technology, digital data communications, and software engineering to integrate navigation, guidance, communications, energy management, stores management, and for those missions requiring weapon delivery, target acquisition/identification and weapon delivery functions with aircraft control functions (outer and inner loops). While these same functions have been performed since the beginning of military aviation, it is only recently, with the progression from no computers to analog, and now to digital computers, that it has become possible to integrate the functions efficiently and reduce the duplication of sensors and processing.

Digital systems, such as the integrated control system, utilize a digital communication architecture made up of computers, digital data transfer channels, and the associated sensors and controllers required to implement the control process. The management of this network of computers, interconnected via the digital data transfer channels, as well as the other integrated control functions, comprises the primary system management function. The system management function may be implemented with both hardware and software. The software must implement the system management operating system (executive software for control of communication protocol which includes software-implemented fault detection and recovery) and interface with the local executive in each processor. The system management function performance is critical to the successful implementation of integrated control.

System architectures, such as the triple redundant mechanization with self-test and in-line monitoring, are representative of concepts for implementing a fault-tolerant system management function in conjunction with the flight control function. The use of majority voting exemplifies present concepts for avoidance of hardware and software faults.

A particular function, such as the control function, may be done automatically, manually, or by a combination of automatic and manual processes. The combination of processes for a particular function shall be defined as specific subfunctions. Each subfunction may be performed in a variety of modes.

A mode is defined as a specific set of measurements (inputs) and appropriate algorithms which are processed to provide desired outputs. These measurements may be provided by specific combinations of sensor hardware and software, or may make use of optimal estimation theory (such as a Kalman filter) to compute or estimate the vehicle state based on input from sensors which may provide redundant measurements or information. The specific mode that is utilized in performing a function is dependent upon a hierarchy of modes established by the system designer for automatic subfunctions and the crew selection of modes using manual subfunctions, as well as the status of the system where status is defined as the hard-

Submitted Nov. 10, 1981; presented as Paper 81-2256 at the AIAA/IEEE 4th Digital Avionics Systems Conference, St. Louis, Mo., Nov. 17-19, 1981; revision received Oct. 29, 1982.

*Project Manager.
†Associate Section Manager.
‡Principal Researcher.

ware/software vector of information regarding all hardware and software conditions at the present time. Many of these modes are essentially identical from one aircraft type to another. Software is used to implement the various functions/subfunctions, and modes in digital systems. In order to minimize the life cycle cost, it is essential that common software be used not only within all models of a specific aircraft weapon system, but across all aircraft types wherever possible.

Core Software Concept

The costs of developing and maintaining the software for integrated control systems are dominant drivers in the overall total life cycle costs. Similarities in existing application software which implement navigation, guidance, and outer loop control modes for the F-111 series aircraft were noted by Battelle in a recent study[3] conducted for the Air Force Systems Command.

The F-111 study gave rise to the concept of core applications software modules which would be common to all aircraft. These core applications software modules implement basic algorithms for common functions/modes and have interfaces defined with the sensor/equipment software modules and with the executive controlling the overall software as shown in Fig. 1. If these interface standards are adhered to, each aircraft can make use of the core applications software modules by developing the sensor/equipment modules required to interface the aircraft's sensors/equipment to the core application modules. Since these core modules primarily implement physical relationships, it is appropriate to formally define these modules in order to reduce the tendency to "reinvent the wheel" in functions such as navigation, guidance, and control.

Analysis Results

Documentation on operational flight programs of the A-7D, F-111F, F-15, and F-16 aircraft was collected and analyzed. Each information processing function currently performed on each aircraft type was broken down into its most fundamental elements and cataloged as function performed, inputs required, equations solved, logic executed, subroutines called, and output produced. Detailed information is given in the Flight Dynamics Laboratory final technical report.[4]

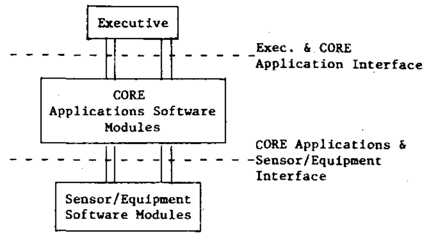

Fig. 1 Diagram of core applications software concept.

Table 1 synopsizes all the major software modules analyzed. Detailed results for each of these module's functions for each aircraft analyzed are also contained in Ref. 4.

The initial analysis revealed each aircraft's software used different coordinate systems or different symbols or mnemonics for the same variable when a common coordinate system existed. Many algorithms are functionally equivalent except for the differences due to use of dissimilar coordinate systems.

Requirements

In order to develop and apply the core software concept across more than a single aircraft type, a set of standard coordinate systems must be used. The functionally equivalent algorithms identified as core candidates through the analysis of the four operational flight programs must be transformed into these common coordinate frames.

Standard definitions, symbols, and mnemonics must be used. Software interface standards between sensor/equipment modules and core modules must be established. Each input/output variable's mnemonic, units, range, and resolution must be established. This permits the designer the flexibility of combining standard data words into the MIL-STD-1553B bus messages required for transfer of data between software modules in different processors.

The system hardware architecture selected by a designer determines primarily the address/subaddress of the data to be transmitted/received between software modules, whether core or sensor/equipment modules. Repartitioning the software in a multiple processor system can be done with relative ease if the standard data words to be transmitted between core

Table 1 Software modules analyzed

A-7D	F-111F	F-15	F-16
Power-up initialization	Executive	Executive	Initialization and error handling
Interrupt processor and control			Executive
			Systems control
			Bus control
Navigation	Navigation sensor	Navigation	Navigation support
Navigation subroutines	Navigation steering	Flight director	
	Mission planning	Controls and displays	
	Fixtaking/target acquisition		Fixtaking
	Miscellaneous		Stores data select
Weapons module 1	Weapon delivery	Air-to-ground	Air-to-ground attack
Weapons module 2			
Mathematical subroutines	Math subroutines	Mathematical subroutines	Support utility
Tactical computer panel	Data entry/display		Data entry/display
		Air-to-air	Air-to-air missile
			Air-to-air gunnery
Self-test	System test	Computer self-test	Self-test
Bit pick			Cruise energy management
			Combat energy management

Table 2 Library of core elements—functions

Navigation	Navigation update
Trajectory generation	Guidance
Steering	Control (outer loop)
Stores management	Weapon delivery
Energy management	Communications
Display	Math subroutines

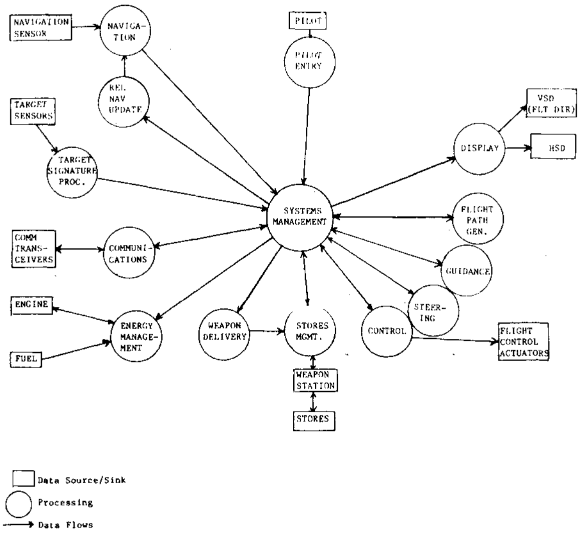

Fig. 2 Integrated control core software data flow.

modules are adhered to and core element integrity is preserved. Attempts to further partition core elements (a single processing function involving solution of an equation set yielding a single output or vector components) into different processors will reduce the processing efficiency and efficacy of the core concept.

Core Elements Library

Each of the processing functions of the F-111F, F-15, F-16, and A-7D was broken down into its most fundamental elements (e.g., $\sin(X)$, $\cos(X)$, platform-to-body direction cosine matrix components, ground speed, etc.). The library of core elements consists of groupings of the lowest level algorithms (fundamental elements) by function (e.g., navigation, guidance, etc.). Within a function, the lowest level algorithms that can be combined into intermediate and high-level modules to yield a specific mode or subfunction are identified. The library of core elements is in the form of a user's manual organized by the high-level functions given in Table 2.

Not all subfunctions of a function nor all modes of a subfunction are amenable to the core concept. In some cases, a function, such as self-test, is computer hardware dependent and is therefore not a core element. Pilot entry (data entry/display) is currently hardware-dependent. It need not be since microprocessors in the data input device (whether keyboard, discrete switches, thumbwheel dials, etc.) can check all pilot inputs for validity (format error, range error, etc.) before the data are transmitted to the using software module. Since none of the data analyzed contained a common approach to data entry, it has not been included in the initial library of core elements.

Core Software Modules

A core software module consists of logical combinations of the appropriate core elements, which are the smallest sequences of code that can be scheduled as a task, needed to perform a specific integrated control function. The logic needed to switch between core modules or a sequence of modules which comprise specific modes, such as true-airspeed/inertial, inertial, and true-airspeed/dead reckon, tests the validity of the input data from each sensor and selects the correct core module sequence for continued processing. This combination of logic, control structures, and core modules comprises the modes for specific functions such as navigation.

Core Software Application

The application of the core software concept across different aircraft types is possible, providng the previously discussed requirements are satisfied. The core software concept is based on physical laws and good software engineering practices. Applying structured analysis principles to the integrated control concept and the software functions needed to implement integrated control yielded the high-level data flow depicted in Fig. 2. The application of the core concept is independent of the many bus network architectures as long as the core processing functions maintain their integrity. This requires that the system management subfunctions controlling fault detection and error correction, core module scheduling, and data transfer at the needed rates between modules be fault tolerant. As long as the required core element modules needed during the current mission phase can be correctly executed and the data transfer maintained, the integrated control functions required will be available.

Conclusions

The integrated control core software concept is feasible based upon a detailed analysis of the operational flight programs of four modern military fighter aircraft. Functionally equivalent algorithms (differing primarily due to use of different coordinate frames and mnemonic labels for identical signals) have been identified and cataloged in a core elements library. These core elements (once put in a common coordinate system) will be the basic processes that will be implemented with standard control structures as core software modules. These core software modules will have a direct payoff in two areas of vital interest to the Air Force—life cycle costs and integrated control research.

Acknowledgment

This work was performed for the USAF/AFWAL under Contract F33615-80-C-3615 with Capt. S. Larimer and Lt. A. Ballenger as Project Engineers.

References

[1] Robinson, A.C. and Hitt, E.F., "Integrated Control—A Unified Approach to Management of an Aircraft," Contract F33615-76-C-3145, Item No. 0002, Sequence No. AFFDL/AC, Request 39, Battelle Columbus Laboratories, Ohio, Dec. 1978.

[2] Hitt, E.F. and Bridgman, M.S., "Formulation of a Methodology for Evaluation of the Mission Effectiveness of Fault Tolerant Integrated Control Systems on a Single Aircraft," Contract F33615-76-C-3145, Item No. 0002, Sequence No. AFFDL/AC, Request 50, Battelle Columbus Laboratories, Nov. 1979.

[3] Hitt, E.F. and Bridgman, M.S., "F-111A/E Digital Bomb-Nav System Software Analysis," AFAL-TR-79-1043, Battelle Columbus Laboratories, Ohio, Jan. 1979.

[4] Hitt, E.F., Broderson, R.L., and Kluse, M., "Integrated Control Core Software Concept Study," AFWAL-TR-81-3141, Battelle Columbus Laboratories, Ohio, Dec. 1981.