NASA Technical Memorandum 4279

Feasibility of Using a Knowledge-Based System Concept for In-Flight Primary Flight Display Research

Wendell R. Ricks

SEPTEMBER 1991

NASA

NASA Technical Memorandum 4279

Feasibility of Using a Knowledge-Based System Concept for In-Flight Primary Flight Display Research

Wendell R. Ricks
Langley Research Center
Hampton, Virginia

NASA

National Aeronautics and
Space Administration

Office of Management

Scientific and Technical
Information Program

1991

Summary

A study was conducted to determine the feasibility of using knowledge-based system architectures for in-flight research of primary flight display information-management issues. Results of an earlier study showed knowledge-based programming techniques reduced development problems experienced when using traditional programming techniques for *workstation* research of primary flight display issues. Plans were then made to use knowledge-based systems for *in-flight* research. However, the feasibility of using knowledge-based systems for in-flight research remained an issue. The feasibility relied on the ability to integrate knowledge-based systems with existing onboard aircraft systems. And, given the hardware and software platforms available for this study, the feasibility also depended on the ability to use interpreted-LISP software with the real-time operation of the primary flight display. In addition to evaluating these feasibility issues, this study determined whether the software-engineering advantages of knowledge-based systems found for this application in the earlier workstation study extended to the in-flight research environment. To investigate these issues, two integrated knowledge-based systems were designed to control the primary flight display according to preexisting specifications of an ongoing primary flight display information-management research effort. These two knowledge-based systems were implemented onboard the Transport Systems Research Vehicle Boeing 737 at the Langley Research Center to assess the feasibility and software engineering issues described above. Flight test results were successful in showing the feasibility of using knowledge-based systems in flight with actual aircraft data. Additionally, the ease and efficiency of implementing and maintaining the systems for the flight tests confirmed the software engineering advantages of the knowledge-based system approach for the in-flight research environment.

Acronyms

DATAC	digital autonomous terminal access communication
KBS	knowledge-based system
PFD	primary flight display
TSRV	Transport Systems Research Vehicle

Introduction

The amount of information already on the primary flight display (PFD) of a commercial "glass cockpit" (see fig. 1) and the likelihood of increasing the amount of information on future PFD's have led to information-management concerns (e.g., display clutter). Research efforts that address these PFD information-management concerns have been underway at NASA Langley Research Center (refs. 1 and 2). One such effort was exploring the management of information on the PFD by dynamically tailoring the optional information to the task(s) the pilot performed during flight. This task-tailored approach to PFD information management was the application used for the study described in this paper and is described in more detail in the project description in reference 3.

Each of the research efforts exploring different PFD information-management approaches required very complex logic to automate. In past efforts, automating different PFD information-management strategies relied on the ability to implement the complex logic with traditional procedural programming techniques in compiled computer languages like FORTRAN. With these earlier implementations, some algorithms (e.g., searches) and some system characteristics (e.g., explanation) were difficult to achieve when using traditional programming methods. Furthermore, with the complexity of the logic and the frequent changes that occurred to the research systems, many software engineering problems resulted.

Most of the software engineering problems resulted from the frequent changes to system specifications that occur in all research environments. The lengthy and involved process necessary to make the changes to the procedural programs, coupled with the complex nature of the logic itself, resulted in systems that were hard to trace, modify, and verify (ref. 4). These problems would worsen with each change that occurred to the logic in the course of the research effort.

Since changes to specifications and subsequently to the code are integral parts of a research system life cycle, software engineering issues are a primary cost and concern. This concern led to the investigation of alternate implementation techniques for the PFD information-management research efforts. As an alternative to the traditional procedural approach to implementing the PFD information-management logic, a knowledge-based system (KBS) approach was explored. The decision to explore a KBS approach was based in part on an earlier study that found KBS architectures easier to manage than the traditional architecture, given complex logic like that needed for the PFD information-management research (ref. 4).

While software engineering successes of KBS approaches for *workstation* research have been well documented (refs. 5–7), uses of KBS architectures for research in the actual *application* environments were far less common. KBS's and the more general category of "rapid prototyping" use a nontraditional software development cycle based on a principle of early systems development (usually incomplete systems) and easy to make and understand incremental revisions to the system. KBS architectures are often used with rapid prototyping because they provide the rich programming environment (e.g., tools for examining declarative knowledge) conducive to rapid prototyping. Even though a few KBS architectures have been taken to their application environment (refs. 8 and 9), using them for in-flight research had not yet been explored. Therefore, feasibility was a concern when plans were developed to use KBS's for in-flight research.

Since PFD information-management research using a KBS architecture was the first study to involve KBS's running in LISP (ref. 10) in real time on the NASA Langley Boeing 737-100 research aircraft, specific feasibility questions arose. These questions concerned the ability to integrate KBS's with the existing onboard aircraft systems and the ability to use interpreted-LISP software for the real-time operation of the PFD. In addition, there was the question of whether the software engineering advantages found in the workstation environment could transfer to the in-flight research environment.

These questions were addressed in the study described in this paper to evaluate the use of KBS architectures for in-flight research of PFD information management. The objectives of this study were to design, implement, and test (in flight) a KBS approach to PFD information management to determine the feasibility of using KBS's as in-flight research tools for this application. To address the software engineering question, an informal evaluation of the advantages of the KBS approach for this application was done while preparing and maintaining the software for flight test.

Research Systems

Aircraft Description

The study described in this paper used the Transport Systems Research Vehicle (TSRV) Boeing 737 at the Langley Research Center (see fig. 2). The TSRV is a specially configured Boeing 737 twin-engine subsonic commercial jet transport. It has two flight decks: a conventional Boeing 737 flight deck for operational support and safety backup, and a fully operational flight deck positioned in the aircraft cabin (see fig. 3) for advanced flight systems research.

At the time of this study, the systems onboard the TSRV consisted of a digital flight control system, a digital navigation and guidance system, and a "glass cockpit" representation of cockpit information in the research flight deck (see fig. 4). The digital flight control system provided both automatic and fly-by-wire control-wheel (stick) steering options. The navigation and guidance systems provided position estimates, flight route definition, guidance commands to the flight controls, and flight data storage for multiple navigation purposes. The "glass cockpit" research flight deck presented information to the flight crew via eight 8-in-square electronic displays that represent the technology already available in commercial transports today (e.g., Boeing 757/767 and Airbus 320). Systems onboard the TSRV were driven by onboard computers and specially developed computer software. The onboard computers consisted of a MicroVAX II, two Nordens (i.e., flight-hardened PDP 11/70), GRiD 386-based laptops, and several 8086- and 8088-based processors for the display systems.

Simulator Description

Some preliminary work (i.e., knowledge acquisition and early testing) in this study took place in the TSRV Simulator for the Boeing 737 airplane at the Langley Research Center (see fig. 5). At the time of this study, the TSRV Simulator was a fixed-base cockpit closely representing the research flight deck of the TSRV aircraft.

The simulation used a full six-degree-of-freedom set of nonlinear equations of motion containing a detailed aerodynamic package, an engine model, a landing gear model, and functional representations of the advanced flight control configurations available to the airplane (with nonlinear models of the servo-actuators). The aerodynamic model incorporated two- and three-dimensional table look-ups for aerodynamic coefficients and adjusted these coefficients for ground effects. The engine model included detailed ram air and temperature effects. The landing gear model included provisions for braking and for steering.

Processing the simulation equations was done by a Control Data Corporation (CDC) CYBER 175 digital computer at a 32-Hz iteration rate. A standard-atmosphere model with no winds was used. As with the airplane, electronic primary and navigation displays were provided as an over-and-under arrangement for vehicle control and guidance, and center-mounted displays for systems management. The formats for the simulator displays were generated by Adage AGT 340 graphics computers. For added realism, the simulator was also equipped with four out-the-window display systems (three were used in this study) driven by an Evans and Sutherland CT6 Computer Generated Image system.

Experiment Design

KBS Design, Implementation, and Integration

The KBS architecture in this study was designed to provide a richer software architecture for exploring PFD information-management issues. The information-management approach employing the KBS's was a task-tailored approach, meaning that information was presented on the PFD when the tasks of the pilot required it.

With this approach, *basic* information on the PFD—that information necessary for the basic guidance and control of the aircraft (e.g., attitude, airspeed, heading, and altitude)—was continuously presented during the flight. The presentation of *optional* guidance and control information (e.g., reference altitude, glide-slope deviation, and vertical path) was tailored to the task(s) of the pilot, so that optional information was presented only when needed. For example, if a task were following a localizer signal, the pilot would be given localizer-deviation information on the horizontal scale of the PFD in place of all other available optional horizontal guidance.

The final software system design for the task-tailored PFD information-management approach consisted of two KBS's: one for flight-phase detection and one for information selection. See figure 6 for a data flow diagram of the system.

Figure 6 shows (moving from the bottom of the diagram to the top) that the final control and guidance information was provided for the pilot via the PFD. The optional information on the final PFD configuration was based on the intersection of two sets of data. One set contained all the sensor and system information residing in the various aircraft computers. The second set identified the optional information from the first set of data that was to be presented on the PFD.

Selecting which optional information to present on the PFD was done by the KBS labeled in figure 6 as "Select Optional Information." It used the following input in its decision logic: the phase of flight, the status of the display switches, the control mode configuration, and the various sensor and system information. The status of the display switches, the control mode configuration, and the various sensor and system information were provided by preexisting onboard systems. The flight-phase data were determined by the KBS labeled in figure 6 as "Determine Phase of Flight," which based its decision on various sensor and system information (e.g., gamma and engine pressure ratio).

The two KBS's consisted of both *passive* and *active* knowledge. Passive knowledge consisted of the facts known a priori, while active knowledge was composed of any methods (e.g., rules and daemon functions) used to make, delete, or modify facts during run time.

Passive knowledge asserted facts known before the program inference began. Some passive knowledge changed during program execution (e.g., initializations) while some did not (e.g., physical laws). For this application, passive knowledge was used primarily for initializations. For example, the assertion

(now-is in-phase taxi)

was passive knowledge used to initialize the phase of flight.

Active knowledge was used to assert facts during execution. Assertions made by active knowledge could either override existing assertions or just be added to them. One use of active knowledge in this system involved the rules for detecting the phase of flight. For example, the rule for takeoff would override any preexisting assertions about phase of flight and assert the fact

(now-is in-phase takeoff)

when the facts supporting the following conditions were true: the automatic detection of flight phases was engaged, the previous phase of flight was taxi or landing, the engine reversers were not engaged, the engine pressure ratio was greater than 1.8, the flaps were at less than or equal to 30° extension, and the radar altitude was less than or equal to 400 ft.

The two final flight-test KBS's consisted of 9 frames; 10 predefined instances; 46 rules varying in number of conditions, types of dependencies, and priority rankings; 4 daemon functions; and 18 miscellaneous functions. More detailed information concerning the implementation (including a listing of the code) can be found in reference 3, which gives a description of the entire project.

The preliminary domain-knowledge acquisition and rule development was done using a PC-based workstation and the TSRV simulator. A PC-based workstation and a KBS shell written in LISP (ref. 4) were used to develop and do preliminary tests of the information selection rules.

The TSRV simulator was used to develop, do preliminary tests of, and to refine the rules needed to automatically detect the phase of flight. For the simulator sessions, seven pilots participated, with some in more than one session. The pilots were NASA test pilots, a United States Navy pilot, an Army Reserve pilot, and NASA employees with various flight ratings. For more information concerning the simulator sessions, refer to the project description in reference 3.

The KBS architecture was implemented and integrated for flight tests onboard the TSRV airplane by using a commercially available knowledge-based expert system development shell. During flight, the software for both KBS's operated in interpreted-LISP mode using an add-on computer card installed in an 80286-based computer. The add-on card was an 80386-based CPU with six megabytes of memory that could be housed in any 8088- or 80286-based computer.

The KBS architecture communicated with the display computer (a Norden for this study) by sending display symbol control words via a digital autonomous terminal access communication (DATAC) bus. The DATAC bus was also the means for retrieval of the information needed as input into the KBS's. The DATAC bus was a 1-MHz serial bus that operated in broadcast mode—every terminal on the bus had access to the transmissions of all other terminals on the bus.

In addition to the KBS development, this study required modification of the input/output (I/O) handlers for the displays on the aircraft. New I/O routines were written to format the input discretes used by the KBS and to unpack the output discretes that dictated what guidance and control information and flight phase to present on the PFD. Existing aircraft software modules were modified so that presentation of the information symbols could be controlled by either the previous procedural implementation or the KBS implementation. Control between the procedural and the KBS implementation was switched during flights via a configuration word set interactively by the experimenter.

Test Procedure

Issues concerning the use of KBS architectures for in-flight research of PFD information issues were evaluated onboard the TSRV in two stages. The first stage of flight tests was designed to isolate the feasibility issues only (i.e., the ability to integrate KBS's with existing onboard systems and the ability to use interpreted-LISP software for the real-time operation of the PFD). This meant only introducing the KBS for information selection (refer to fig. 6). The second stage of flight tests involved the integration of the flight-phase detection KBS to look further at the feasibility issues and to assess the benefit of KBS architectures when developing new flight system software (i.e., new functionality). The reason for separate stages was to maintain the functionality of the earlier procedural implementation of the same PFD information-management approach when looking strictly at the feasibility issues.

In the earlier procedural implementation, the current flight phase of the aircraft was provided to onboard systems by the pilot or test engineer, not by automation. To test the feasibility of the KBS concept, it was desired to compare the new KBS implementation against the already flight-proven procedural implementation. To isolate the feasibility issues, changes to the functionality could not take place. If additional functionality were added to the KBS implementation, deviations between the performances of the traditional and KBS implementation could be attributed either to problems with the functional changes or to the feasibility issues. So, to isolate the feasibility issues, the functionality of the KBS for the initial flight tests did not differ from that of the traditional implementation. Appendixes A and B provide the flight-test envelopes used for the stage 1 and stage 2 tests, respectively.

Stage 1 flight tests. Designing the tests where both the KBS and the traditional system had the same functionality allowed a successful feasibility evaluation of the stage 1 tests to be defined as a KBS implementation and integration that duplicated the behavior of the traditional implementation. Duplicate behavior would indicate the successful integration of the KBS software with the existing onboard software and also the ability to use interpreted-LISP software for the real-time operation of the PFD. To obtain this comparison data during the flight tests, both subjective pilot evaluation and system-generated output discretes were used.

During flight, the KBS implementation controlled the PFD. So, getting a pilot evaluation depended on the pilot's previous knowledge of the operation of the traditional implementation. The test pilot for this study had flown many hours in the TSRV research cockpit and had also been a major contributor to the logic used for the task-tailored management of PFD information. Therefore, the pilot was familiar with how the traditional implementation controlled the optional information on the PFD in the past and how the logic was to work in general. This enabled the pilot to give immediate feedback concerning deviations on the PFD from what was expected. Pilot comments during the flight tests were manually recorded for postflight analysis.

Comparison data for the traditional and KBS implementations were also recorded throughout the flight as discrete display control words (see table I). Two discrete words were sufficient to represent the information elements driven by the task-tailored approach. When a bit in a control word was set (i.e., equal to 1), the relative display element was active. For example, when bits 2 and 3 in control word 0 were set and the remaining bits were 0, then horizontal deviation and glide-slope deviation were the only active elements of word 1.

Table I. PFD Optional Information Control Words

Control word	Bit	Indication
0	0	Reference altitude
	1	Waypoint star
	2	Horizontal deviation
	3	Glide-slope deviation
	4	Localizer deviation
	5	CAS reference (dial)
	6	CAS reference (buffer)
1	0	Runway image
	1	Radar altitude
	2	Vertical path
	3	Flare guide
	4	Track-angle error 1
	5	Track-angle error 2
	6	Track-angle error 3

The comparison data used for postflight analysis consisted of the display elements active under both the traditional and the KBS implementation; each implementation generated and stored its own display control words. While the optional information on the PFD was driven by the KBS implementation during the flight tests, both the traditional and the KBS implementation were independently generating display control words for postflight analysis. Recording the comparison data in encoded word form eased the postflight comparison analysis.

Stage 2 flight tests. The second stage of flight tests was designed to assess the utility of KBS architectures when adding new system functionality, namely the automatic flight phase detection. In the stage 2 tests, a successful evaluation of both the flight-phase detection logic and the integration of the new KBS would again confirm the feasibility of the KBS approach for this application. Additionally, it would show the feasibility of using KBS architectures for research test beds of new system functionality.

The metrics used to evaluate the implementation and integration of the new KBS with the other KBS and TSRV systems were the same measures used with the stage 1 evaluation—the PFD behaviors (assessed again by pilot evaluation and comparison data). Since the automatic flight-phase detection KBS was not designed to change the PFD behavior but to eliminate the need for the pilot to enter flight phases manually, the performance of the PFD should have been the same as the traditional procedural implementation, given correct manual entries.

The flight-phase detection logic during these tests was evaluated by comparing the phases detected by the KBS architecture with those expected. Two additional control words were added for postflight evaluation of the stage 2 tests (see table II). The control word representing the flight phase was decoded and displayed on the PFD screen during the flight. This presentation of flight phase was used by the test engineer to note whether the logic was correctly detecting flight phases during the flights. Video recordings of the PFD were used with the other recorded data for postflight analysis.

Table II. Flight-Phase Control Words

Control word	Bit	Indication
2	0	Takeoff
	1	Terminal climb
	2	Cruise
	3	Terminal descent
	4	Land
	5	Taxi
	6	Enroute climb
	7	Enroute descend
3	0	Error flag

Only one flight phase was true at any given time; therefore only one bit in word 2 should have been set at any one time. The set bit corresponded to the active flight phase. For example, when bit 2 of control word 2 was set (i.e., equal to 1), then the current flight phase was cruise. The error bit (word 3) was set when errors were reported by the flight-phase-detection KBS (e.g., two phases true at one time). Again (as with stage 1), the use of encoded computer words for comparison data eased postflight analysis.

Software engineering issues. Throughout the flight tests and their preparation, experiences with the implementation and integration of the flight-test KBS's were used for an informal validation of the software engineering advantages identified in a previous workstation study comparing KBS with traditional implementation of the same application (ref. 4). In the development and maintenance process, frequent modifications to the KBS's were necessary to get the functional equivalent of the earlier procedural implementation as well as to add the new flight-phase detection functionality. This provided adequate data for evaluating the software engineering advantages of the KBS approach for in-flight research.

To provide the software engineering advantages of the KBS environment while in flight, the environment used to develop the KBS software on the ground was put in its entirety on the airplane. So, all the development tools (e.g., explanation) provided on the ground were on the airplane during the flight.

Results and Discussion

Stage 1 Flight Tests

Major irregularities (e.g., not displaying the correct piece of information) did not occur during the flight tests. However, when the pilot's attention was directed toward the PFD for the purpose of looking for delays, he was able to detect slight delays with the KBS implementation in the first depiction of some optional PFD information. Throughout the flight tests, short delays (of a few seconds or less) could be noted with the first appearance of a few of the PFD information display elements (randomly distributed).

The increased time needed to initiate PFD information formats was attributed to the slower nature of an "interpreted" (as opposed to compiled) language and hardware, as well as to the simple addition of a new module in the TSRV data communications. (The traditional implementation of the task-tailored information management was embedded in the graphics code of the display computer.) The addition of a new module meant extra steps were required to retrieve the input information from the DATAC bus, process the information, and then send it to the display computer (via the DATAC bus) for formatting. Even with the overhead of a new node in the DATAC bus, a compiled version of the code would have sped up updates considerably, and the use of newer, faster hardware now available would have lessened, if not eliminated, these delays altogether.

These same delays were also evident in the recorded comparison data. Like the pilot evaluation, the delays were the only deviations. The delays, although possible to minimize given the changes mentioned above, caused no problem given the application used for this study. Flight operations were not interrupted since these delays were with optional information and in most cases not noticeable. The delays averaged less than a second, and were no more than approximately 3 seconds at most.

Had the delays occurred with more time sensitive information, operations may have been adversely affected. Instead, feedback from the pilot concerning overall PFD performance was positive, and the recorded output data of the KBS implementation and the traditional implementation were equivalent. These results confirmed the ability to successfully integrate and operate the KBS architecture for in-flight research with real aircraft data.

Stage 2 Flight Tests

Results of the stage 2 flight tests were also favorable. The flight-phase detection logic was successful for all elements within the flight-test envelope except one (see appendixes A and B for segments of the flight-test envelopes). This one point in the flight-test envelope called for a touch-and-go where the KBS was supposed to detect the transition from landing to takeoff. However, a transition to taxi occurred because of an error in the knowledge—the value given for the flaps setting in the takeoff rule.

The erroneous flap condition in the takeoff rule was not immediately obvious; yet, isolating the problem was easy given the KBS environment. With the interactive tools available in the KBS environment (e.g., traces and explanations), it was easy to query the system and isolate the erroneous condition. It was also possible to make the change while protecting the rest of the system from negative ramifications (i.e., negative side effects of the new conditions).

The tests were successful. The correct mapping of PFD behavior and pilot evaluation again confirmed the feasibility of using KBS architectures for in-flight research. The successful implementation of the flight-phase detection logic showed the feasibility of using KBS for the implementation of new system functionality. And, the ability to interactively isolate the error and modify the code while in flight was a software engineering advantage not previously available with the TSRV onboard software.

Software Engineering Evaluation

The KBS structure and programming environment proved to be as advantageous in the in-flight research environment as it was in the earlier workstation study. The preliminary workstation study showed that the software engineering advantages of KBS architectures were primarily due to structural differences between KBS's and more conventional software systems (i.e., ones that use traditional programming techniques). In conventional software systems, the knowledge pertinent to the problem and the methods for using the knowledge are intertwined, making it more difficult to isolate and modify a specific operation of the system. In a KBS, there is a clear distinction and separation between the knowledge of the problem (the knowledge data base) and the methods for applying the knowledge to the problem (the inference mechanism). This separation allows for simple modification of the program by providing an architecture easily susceptible to the development of routines that explain the execution and produce information needed to verify performance.

Again, these KBS features helped during the initial development and continuous maintenance, and in the explanation of system performance during the flight tests. Positive programmer feedback and the additional data point of isolating the logic error in the flight-phase detection KBS during the flight tests were further evidence of the software engineering advantages of KBS architectures for this application.

Concluding Remarks

A study to evaluate the feasibility of a knowledge-based system (KBS) concept for in-flight research of PFD issues was completed. The objectives of the study were to test the KBS architecture in flight, to demonstrate the ability to integrate KBS's with existing onboard aircraft systems, and to use interpreted-LISP software for the real-time operation of the PFD. In addition, this study addressed the issue of whether the software engineering advantages of KBS architectures found for this application while in a workstation environment also would hold true in the in-flight research environment.

Results of the flight tests showed the feasibility of using KBS architectures for in-flight research of PFD information management issues. Flight tests were also successful in validating the implementation and integration of an additional KBS used to detect flight phase (an input into the other KBS). Preparing the KBS's for flight tests provided the information necessary to confirm the software engineering advantages of KBS architectures in the in-flight research environment. This, coupled with the ease in which the one logic error in the flight-phase detection KBS was isolated during the flight tests and the ability to make a modification during the flight tests, was even further evidence of the software engineering advantages of KBS architectures for this application.

NASA Langley Research Center
Hampton, VA 23665-5225
July 24, 1991

Appendix A

Stage 1 Flight-Test Envelopes

The stage 1 flight tests were done during the baseline verification flights on June 7 and 13, 1989. The flight-test envelopes for June 7 and 13, 1989, are given below. The following acronyms apply to both this appendix and appendix B.

Acronyms

ADIRS	air data inertial reference system
AOA	angle of attack
CAS	calibrated airspeed
CL	coefficient of lift
CL_{max}	maximum lift coefficient
CR	cruise
EC	enroute climb
ED	enroute descend
FFD	forward flight deck
HOR	horizontal (control mode path command)
LD	land
MLS	microwave landing system
PMC	panel-mounted controller
RFD	rear flight deck
SAC	side arm controller
STAR	standard terminal arrival route
T/O	takeoff
TC	terminal climb
TD	terminal descend
TTFIM	task-tailored flight information manager
TX	taxi
VCWS	velocity control-wheel steering
VERT	vertical (control mode path command)
WFBx	Wallops Flight Center waypoint x (where x takes a numerical value)

June 7, 1989, TSRV Flight-Test Envelope

Flight deck	Configuration	Purpose	Procedure
RFD	Cruise-trimmed Sidearm control	VCWS mode check Display system check	Engage RFD Select VCWS Check pitch, roll, and yaw
FFD	STAR WFB13 Altitude = 4000 ft CAS = 210 knots ADIRS Autothrottle	MLS autoland evaluation New throttle	Engage automatic HOR and VERT path Enter STAR at first waypoint Enable MLS when MLS is valid Pilot call out altitude and crosstrack errors at MLS engage Arm land mode in final turn Continue landing through rollout
RFD	STAR WFB13 Altitude = 4000 ft CAS = 210 knots SAC configuration 1.0 ADIRS	Manual MLS Approach and landing Pilot evaluation of SAC configuration	Engage VCWS and autothrottles Enter STAR at first waypoint Enable MLS when valid Pilot call out altitude and crosstrack errors at MLS engage Continue approach to go-around
RFD	SAC configuration 2.0	Pilot evaluation of SAC alternate configuration	Repeat previous procedure Exercise lateral trim switch

Flight deck	Configuration	Purpose	Procedure
RFD	SAC configuration 3.0	Same as previous	Repeat previous procedure

RFD	Gear up
	Flaps up
	CAS = 210 knots
	Altitude = 10 000 ft
		Stick shaker check
		AOA vane check
			Engage RFD
			Select altitude hold
			Set idle thrust; FFD safety pilot should check that throttles are on aft stop; FFD pilot should call out CL/CL$_{max}$ values in 0.1 increments; RFD pilot initiates recovery

RFD	Gear up Flaps 1 CAS = 200 knots	Same as previous	Repeat previous procedure
RFD	Gear up Flaps 5 CAS = 195 knots	Same as previous	Repeat previous procedure
RFD	Gear up Flaps 10 CAS = 180 knots	Same as previous	Repeat previous procedure
RFD	Gear up Flaps 15 CAS = 165 knots	Same as previous	Repeat previous procedure
RFD	Gear up Flaps 25 CAS = 160 knots	Same as previous	Repeat previous procedure
RFD	Gear down Flaps 25 CAS = 160 knots	Same as previous	Repeat previous procedure
RFD	Gear down Flaps 30 CAS = 155 knots	Same as previous	Repeat previous procedure
RFD	Gear down Flaps 40 CAS = 140 knots	Same as previous	Repeat previous procedure

RFD	Cruise-trimmed
	
	Altitude = 10 000 ft
	
	CAS = 250 knots
	SAC configuration X.1p*
		VCWS performance
		Longitudinal axis
		
		Display evaluation
			Engage RFD VCWS
			Apply 1/4 PMC step input up allowing 2° increase in attitude
			
			Release and allow to stabilize
			Return to level
			Displays verify: flight path angle command, symbol and position, pitch attitude, VCWS indication, drift angle indication

RFD	Same as previous	Same as previous	Repeat previous procedure for 1/4 PMC down and 2° attitude decrease

*Either 1, 2, or 3 depending on results of previous runs.

June 13, 1989, TSRV Flight-Test Envelope

Flight deck	Configuration	Purpose	Procedure

RFD	Cruise-trimmed
	
	Altitude = 10 000 ft
	
	CAS = 250 knots
	SAC configuration 4
		VCWS performance
		Longitudinal axis
		
		Display evaluation
			Engage RFD VCWS
			Apply 1/4 PMC step input up allowing 2° increase in attitude
			
			Release and allow to stabilize
			Return to level
			Displays verify: flight path angle command, symbol and position, pitch attitude, VCWS indication, drift angle indication

RFD	Same as previous with CAS = 300 knots	Same as previous	Same as previous
RFD	Same as previous with CAS = 250 knots	Same as previous	Same as previous with 1/4 PMC down and 2° attitude decrease
RFD	Same as previous	Same as previous	Same as previous with 1/4 PMC up and 5° attitude increase
RFD	Same as previous with CAS = 300 knots	Same as previous	Same as previous
RFD	Same as previous	Same as previous	Same as previous
RFD	Same as previous	Same as previous	Same as previous
RFD	Same as previous	Same as previous	Same as previous

RFD	Cruise-trimmed
	
	Altitude = 10 000 ft
	
	CAS = 250 knots
		VCWS performance
		Longitudinal axis
		
		Precise control
		Display symbology checks
			RFD engage VCWS
			Increase flight path angle in 1° steps to 5°, stabilizing at each step
			Return to level

RFD	Same as previous	Same as previous	Same as previous with decrease in 1° steps to −5°, stabilizing at each
RFD	Same as previous	Same as previous	Same as previous with increases and using manual click trim switch
RFD	Cruise-trimmed Altitude > 10 000 ft SAC configuration 4 Manual throttle fixed CAS = 250 knots	Manual electric pitch stability check	Disconnect RFD Trim aircraft Reengage RFD Apply 1/4 PMC step input up allowing 2° increase in attitude Release, allow to stabilize Return to level Display verify: flight path angle command, symbol and position, pitch attitude, VCWS indication, drift angle indication
RFD	Same as previous with CAS = 300 knots	Same as previous	Same as previous

Appendix B

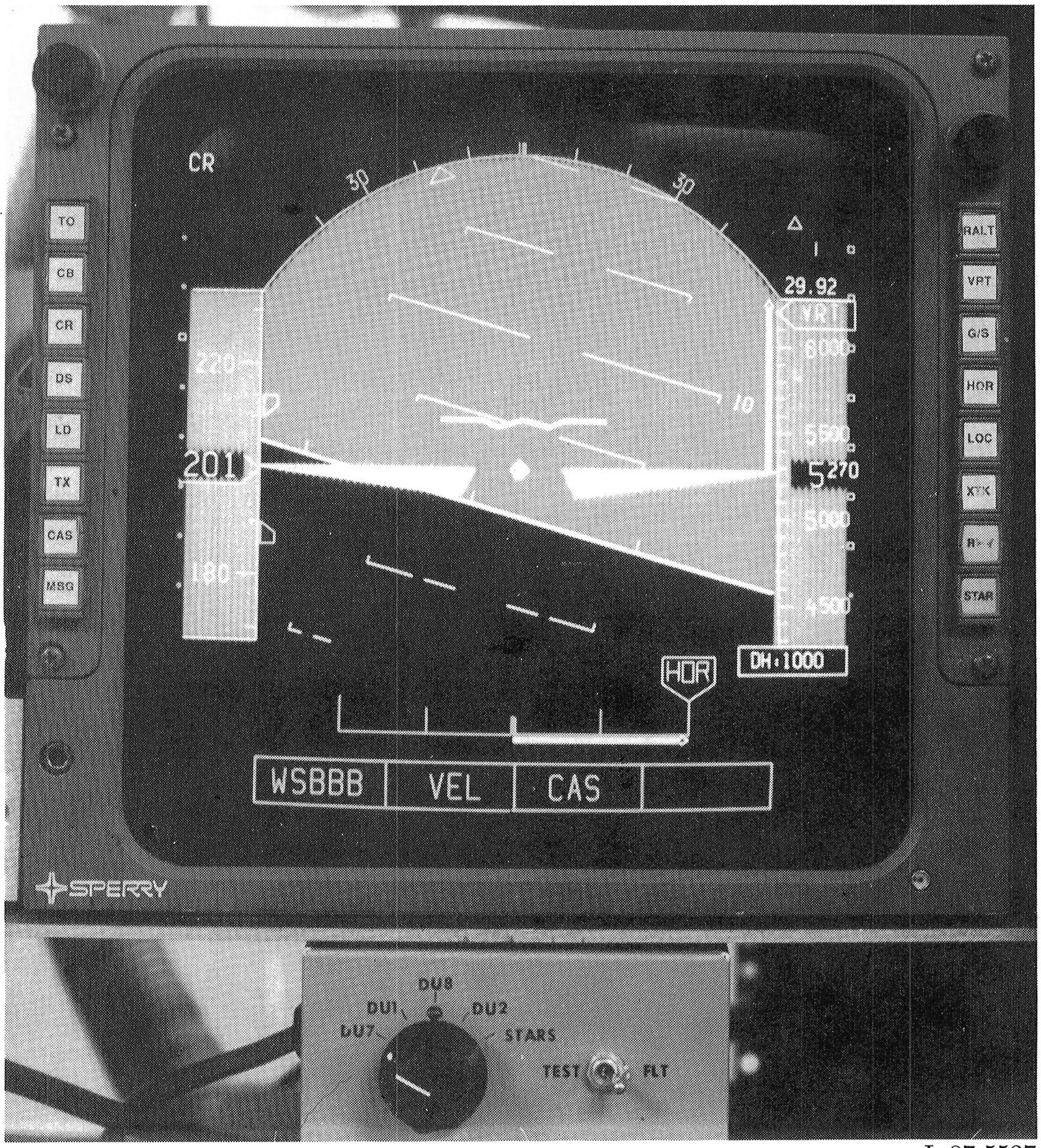
Stage 2 Flight-Test Envelope

November 13, 1989, Portion of the TSRV Flight-Test Envelope

Flight deck	Configuration	Purpose	Procedure
RFD	Altitude < 10 000 ft	TTFIM automatic flight-phase detection check in terminal area	TX T/O TC, then level off TD, then level off
RFD	Same as previous	Same as previous	TC, no level off TD, no level off
RFD	Altitude > 10 000 ft	TTFIM automatic flight-phase detection check outside terminal area	TC EC CR, level off, vary gamma (-1° and 1°) ED CR EC CR EC, no level off, straight to next run
RFD	Touch and go STAR WFB13	Same as previous for touch and go	ED TD LD (touch and go) T/O
RFD	Full mission phase detection	Same as previous for full mission	TC EC CR ED TD LD TX

References

1. Steinmetz, George G.: *Development and Evaluation of an Airplane Electronic Display Format Aligned With the Inertial Velocity Vector.* NASA TP-2648, 1986.
2. Abbott, Terence S.; Nataupsky, Mark; and Steinmetz, George G.: *Effects of Combining Vertical and Horizontal Information Into a Primary Flight Display.* NASA TP-2783, 1987.
3. Ricks, Wendell R.: *Knowledge-Based System for Flight Information Management.* NASA TM-102685, 1990.
4. Ricks, Wendell R.; and Abbott, Kathy H.: *Traditional Versus Rule-Based Programming Techniques: Application to the Control of Optional Flight Information.* NASA TM-89161, 1987.
5. Rushby, John: *Quality Measures and Assurance for AI Software.* NASA CR-4187, 1988.
6. Rouse, William B.; Geddes, Norman D.; and Hammer, John M.: Computer-Aided Fighter Pilots. *IEEE Spectr.*, vol. 27, no. 3, Mar. 1990, pp. 34–41.
7. Buchanan, Bruce G.; and Shortliffe, Edward H., eds.: *Rule-Based Expert Systems—The MYCIN Experiments of the Stanford Heuristic Programming Project.* Addison-Wesley Publ. Co., c.1984.
8. Bachant, Judith; and McDermott, John: R1 Revisited—Four Years in the Trenches. *Readings from AI Mag.*, vols. 1–5, 1980–1985, pp. 177–188.
9. Barr, Avron; and Feigenbaum, Edward A., eds.: *The Handbook of Artificial Intelligence, Volume II.* William Kaufmann, Inc., c.1982.
10. Winston, Patrick Henry; and Horn, Berthold Klaus Paul: *LISP*, Third ed. Addison-Wesley Publ. Co., c.1989.

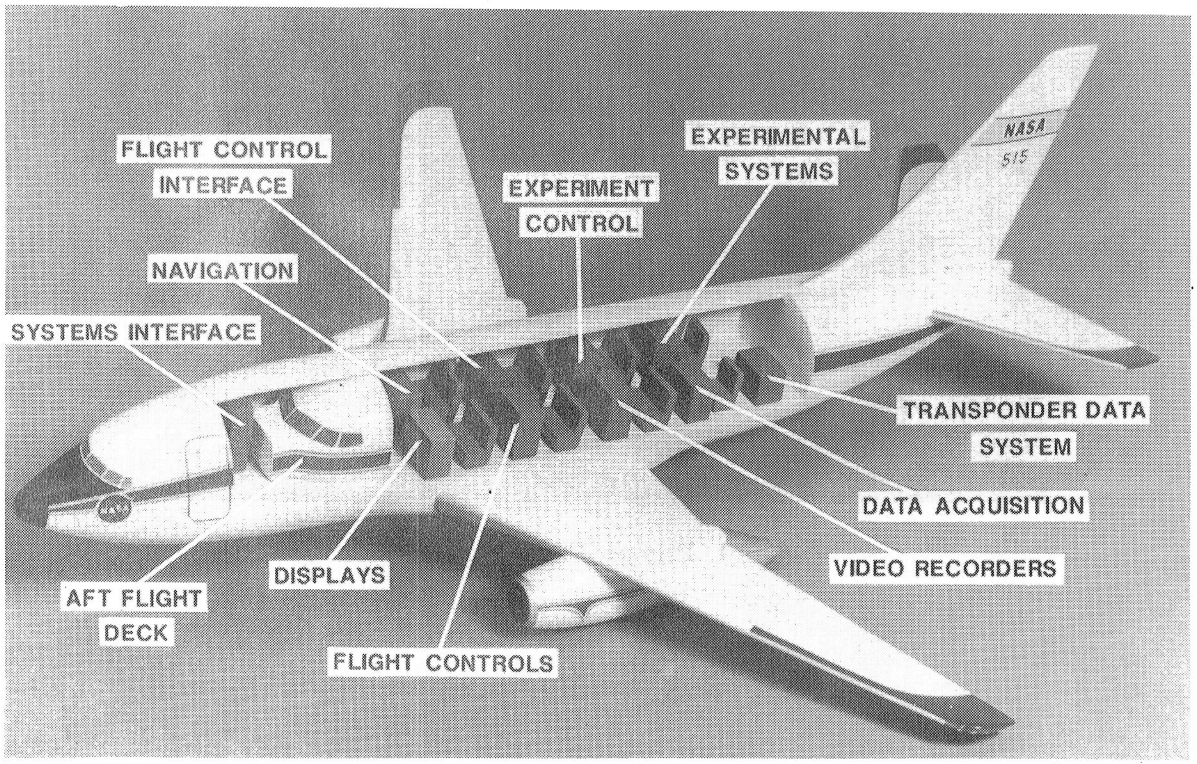


L-87-5527

Figure 1. Primary flight display.

L-89-12405

Figure 2. TSRV Boeing 737 airplane.



L-85-10,992

Figure 3. TSRV airplane configuration.

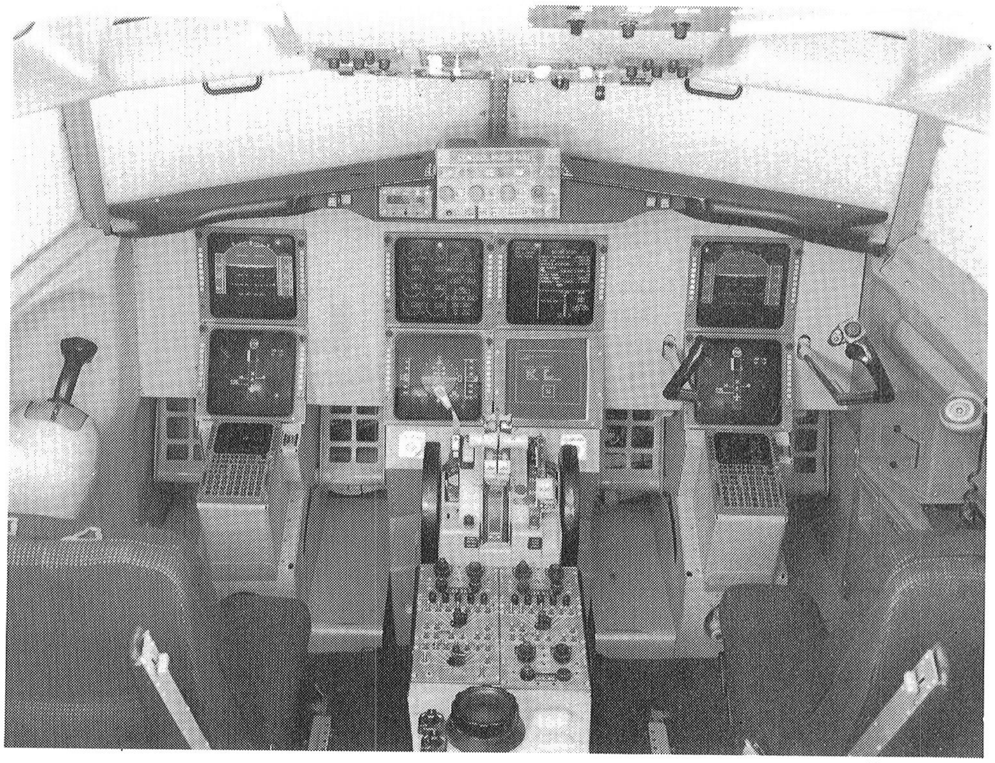

L-90-8321

Figure 4. TSRV airplane experimental cockpit.

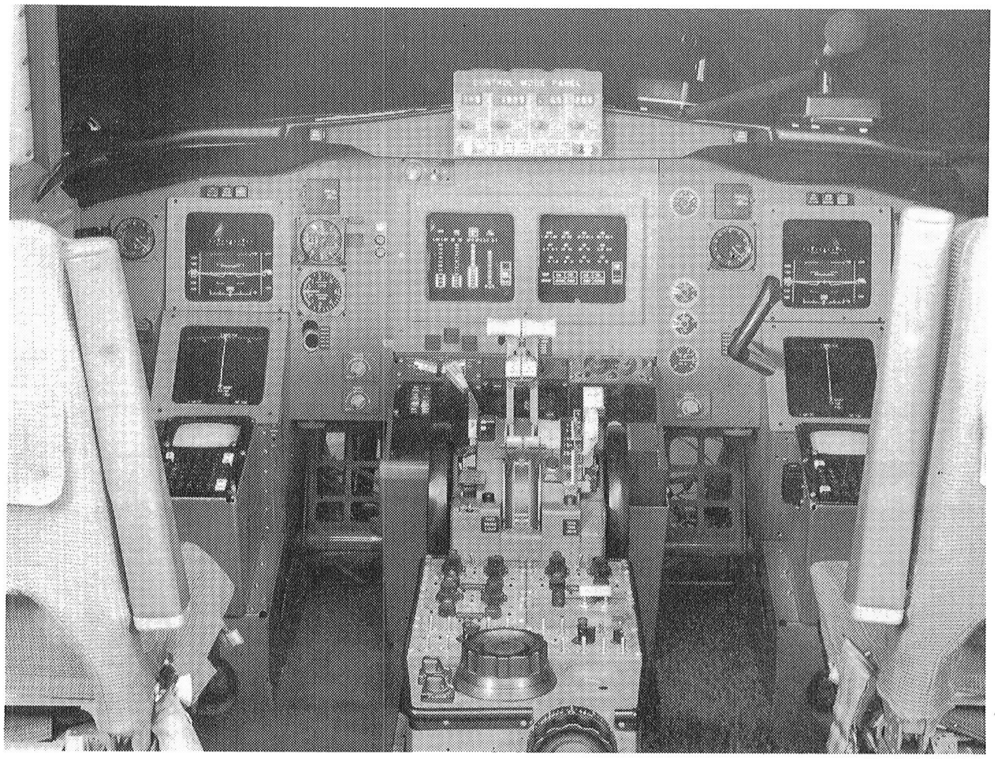

L-86-3593

Figure 5. TSRV simulator cockpit.

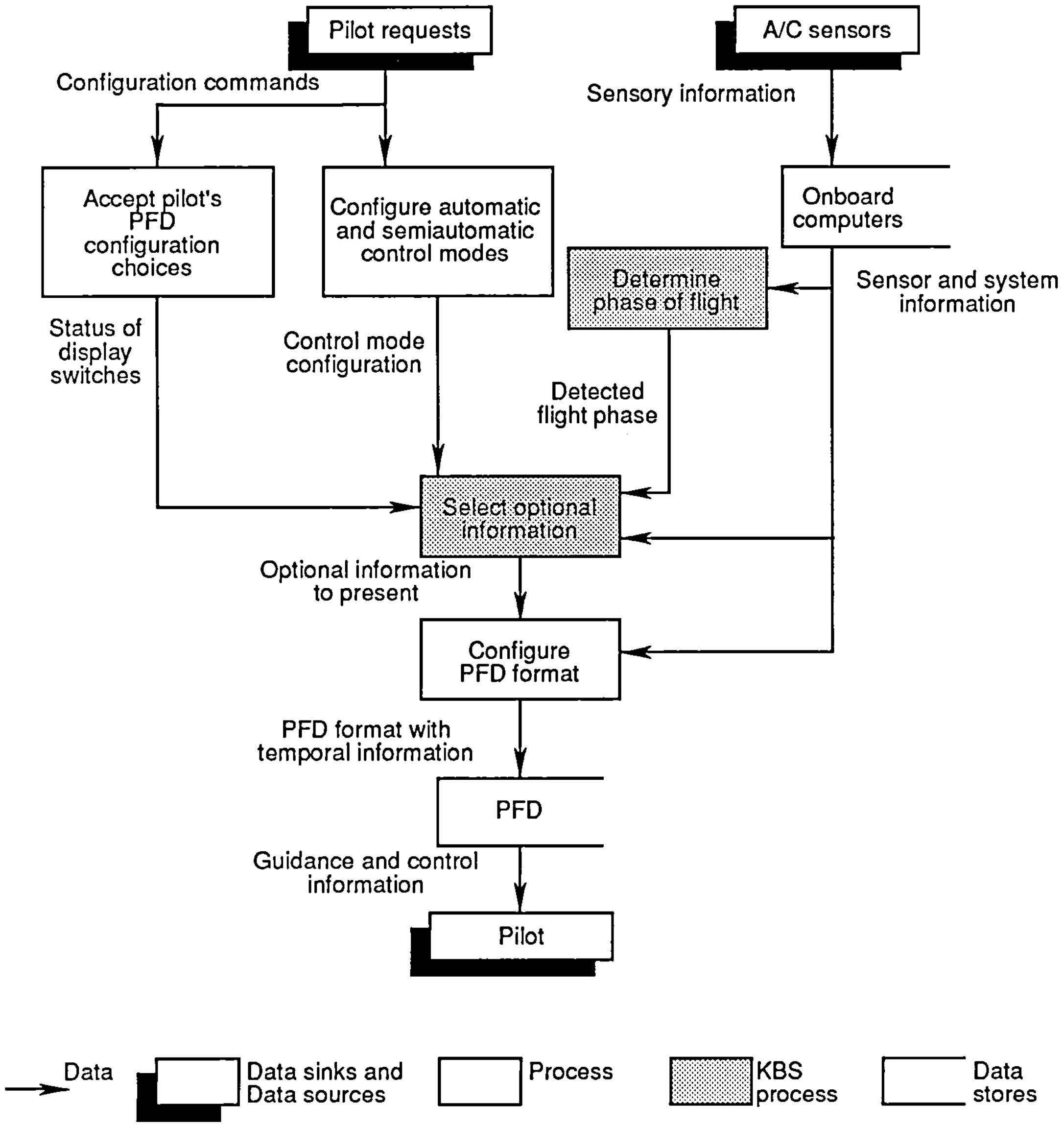

Pilot requests
A/C sensors
Configuration commands
Sensory information
Accept pilot's PFD configuration choices
Configure automatic and semiautomatic control modes
Onboard computers
Determine phase of flight
Sensor and system information
Status of display switches
Control mode configuration
Detected flight phase
Select optional information
Optional information to present
Configure PFD format
PFD format with temporal information
PFD
Guidance and control information
Pilot
Data
Data sinks and Data sources
Process
KBS process
Data stores

Figure 6. Data flow diagram of onboard system.

NASA
National Aeronautics and Space Administration

Report Documentation Page

1. Report No. NASA TM-4279	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle Feasibility of Using a Knowledge-Based System Concept for In-Flight Primary Flight Display Research		5. Report Date September 1991
		6. Performing Organization Code
7. Author(s) Wendell R. Ricks		8. Performing Organization Report No. L-16917
9. Performing Organization Name and Address NASA Langley Research Center Hampton, VA 23665-5225		10. Work Unit No. 505-64-13-22
		11. Contract or Grant No.
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, DC 20546-0001		13. Type of Report and Period Covered Technical Memorandum
		14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

A study was conducted to determine the feasibility of using knowledge-based system architectures for in-flight research of primary flight display information-management issues. The feasibility relied on the ability to integrate knowledge-based systems with existing onboard aircraft systems, and the ability to use interpreted-LISP software with the real-time operation of the primary flight display. In addition to evaluating these feasibility issues, the study described in this paper determined whether the software-engineering advantages of knowledge-based systems found for this application in an earlier workstation study extended to the in-flight research environment. To investigate these issues, two integrated knowledge-based systems were designed to control the primary flight display according to preexisting specifications of an on-going primary flight display information-management research effort. These two knowledge-based systems were implemented onboard the NASA Langley Boeing-737 Transport Systems Research Vehicle aircraft to assess the feasibility of software-engineering issues listed above. Flight test results were successful in showing the feasibility of using knowledge-based systems in flight with actual aircraft data. And, the ease and efficiency of implementing and maintaining the systems for the flight tests confirmed the software engineering advantages of the knowledge-based system approach for the in-flight research environment.

17. Key Words (Suggested by Author(s)) Knowledge-based systems Flight tests Feasibility LISP		18. Distribution Statement Unclassified—Unlimited Subject Category 03	
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 17	22. Price A03

NASA FORM 1626 OCT 86 NASA-Langley, 1991