FINAL DRAFT

R-700

MIT's ROLE IN PROJECT APOLLO

FINAL REPORT ON CONTRACTS
NAS 9-153 and NAS 9-4065

VOLUME V

THE SOFTWARE EFFORT

by

Madeline S. Johnson

with

Donald R. Giller

March 1971

CHARLES STARK DRAPER LABORATORY
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
CAMBRIDGE, MASSACHUSETTS 02139

309

ACKNOWLEDGMENTS

This report was prepared under DSR Project 55-23890, sponsored by the Manned Spacecraft Center of the National Aeronautics and Space Administration through Contract NAS 9-4065.

The authors express appreciation to their many colleagues, both within and without the Draper Laboratory, who gave of their time and contributed essential information to this Final Report. These persons are cited at the end of this Volume in a section entitled, "A Note on Sources". We offer special thanks to Joseph A. Klawsnik, who, though with us only briefly, left considerable impact on the format and contents of Sections II and III herein.

The publication of this document does not constitute approval by the National Aeronautics and Space Administration of the findings or conclusions contained herein. It is published for the exchange and stimulation of ideas.

REPORT R-700

MIT's ROLE IN PROJECT APOLLO

Final Report on Contracts
NAS 9-153 and NAS 9-4065

VOLUME V
THE SOFTWARE EFFORT

ABSTRACT

Seventy-six days after the President of the United States committed the nation to a massive lunar-landing program, the Charles Stark Draper (formerly Instrumentation) Laboratory of the Massachusetts Institute of Technology received the first major contract of the Apollo program. This volume of the Final Report discusses the efforts of Laboratory personnel in developing the specialized software for the Guidance, Navigation and Control System. Section I presents the historical background of the software effort. Section II discusses the software architecture developed for the Apollo Guidance Computer. Section III treats the methods of testing and verification of the flight programs, and the Laboratory's mission-support activities. Four appendices present functional descriptions of some major program capabilities—coasting-flight navigation, targeting, powered-flight navigation and guidance, and the digital autopilots.

PREFACE

Rarely has mankind been so united as in its awe at one man's step onto the lunar surface. When Neil Armstrong placed his left foot in the dust of the moon, engineers and scientists at the Massachusetts of Technology Instrumentation Laboratory felt a special pride for their significant contribution to this accomplishment in the design of the Primary Guidance, Navigation and Control System for the Apollo spacecrafts.

This report discusses the efforts of Instrumentation Laboratory personnel in developing the special software for the Guidance, Navigation and Control System. Although it is part of a multi-volume series documenting the total Project Apollo efforts of the Instrumentation Laboratory, this section may be read independently of the other volumes; the authors intend it to be meaningful to the general reader who may or may not have read the preceding volumes.

In January 1970, this facility became the Charles Stark Draper Laboratory, named in honor of its founder and current President. Throughout this report, "MIT" and "Draper Laboratory" are used interchangeably, in reference to the former Instrumentation Laboratory.

CONTENTS

ILLUSTRATIONS

SECTION I

HISTORY OF THE SOFTWARE EFFORT

1.1 Introduction

Seventy-six days after John Fitzgerald Kennedy committed the United States to participation in a massive lunar-landing program, the Instrumentation Laboratory* of the Massachusetts Institute of Technology received the first major contract of the Apollo program. Steps leading to this award, however, did not begin 25 May 1961—the day of the President's special message to Congress; the footprints of this history trace back at least several years earlier.

In the Fall of 1957, a group of scientists and engineers at MIT began the investigation of a recoverable interplanetary space vehicle. Under contract to the U.S. Air Force, the MIT group collaborated with AVCO Corporation, the Reaction Motors Division of Thiokol Chemical Corporation, and MIT's Lincoln Laboratory. As reported in the MIT/IL document R-235, "A Recoverable Interplanetary Space Probe", this investigation established the feasibility of designing a vehicle which would journey to a neighboring planet, take a high-resolution photograph there, and return for recovery on earth. The investigators studied the navigational techniques and interplanetary orbits which would be required for a variety of such missions. This study served to bring the engineering problems of interplanetary navigation, attitude control, communications, reentry, and space exploration into sharp focus. R-235 argued that the "early execution of a recoverable interplanetary space probe is an effective means for advancing the state-of-the-art in self-contained interplanetary navigation and control needed for later scientific and military achievements". Furthermore, the report stressed that the "successful physical recovery of a small vehicle which has navigated itself around the solar system and which brings back

*As explained in the Preface, the Laboratory was renamed the Charles Stark Draper Laboratory in January 1970.

photographic evidence of its close and well-controlled passage by another planet is certain to enhance the prestige of this nation".

Following the publication of the study in July 1959, the newly established National Aeronautics and Space Administration undertook its first contract with the Draper Laboratory. In September 1959, MIT agreed to investigate guidance and navigation concepts for a variety of interplanetary missions. Placing emphasis on unmanned missions, the Draper Laboratory devised a system for automatic guidance, including the design of an automatic sextant. Upon completion of this contract in March 1960, several months of discussion ensued between representatives of MIT and NASA's Space Task Group, headquartered at Langley Field, Virginia. A second study contract resulted, this one for another six-month effort: MIT was to present a preliminary guidance and navigation design for a manned lunar-landing mission. This study ran concurrently with several industry investigations of the overall Apollo spacecraft mission.

Although work on the preliminary guidance and navigation design for a manned mission began in late 1960, the actual contract was not announced until 7 February 1961. Midway through the contractual period, President Kennedy declared that a manned lunar landing and return would be a national goal for the 1960s. The President's decision opened the way for formal contractual designations by NASA for design, development and manufacture of the various Apollo spacecraft systems.

Thus, by the time of the Presidential message to Congress, the Draper Laboratory had demonstrated scientific and engineering competence in three space studies: the early recoverable space-vehicle investigation; the six-month unmanned guidance and navigation study; and the preliminary manned guidance and navigation examination. Another factor which proved influential in NASA's assessment of MIT's capabilities was the Laboratory's responsibility for the design and development of guidance and navigation systems for the Polaris guided missile. MIT's experience with the U.S. Navy's Polaris project included engineering and managerial techniques which, it appeared, might be implemented during Project Apollo. Indeed, during the month of July 1961, representatives of NASA and the Laboratory studied the development and scheduling of the Polaris guidance and navigation system, from original conception through production. The group plotted a rough schedule for a similar program on Apollo. NASA representatives also expressed interest in MIT's subcontractor philosophy on Polaris: through significant support by subcontractors, the Draper Laboratory had been able to build up a working force and achieve substantial results in a relatively short period of time. Thus, though Project Apollo would undoubtedly prove to be a much larger and more complex task than Polaris, MIT had demonstrated achievement on a qualitatively similar project.

As a result of the preliminary manned guidance and navigation study, NASA's Space Task Group recommended that the guidance and navigation portion of the Apollo spacecraft mission be negotiated as a contract separate from the development of the Apollo spacecraft. Shortly after this decision was made, and following a noncompetitive, sole-source procurement procedure, the Space Task Group designated MIT to implement the guidance and navigation system of the Apollo spacecraft. Announced 9 August 1961, the first major Apollo contract awarded by NASA called on MIT to conduct a Navigation and Guidance System Development Program which would "meet the intermediate as well as the ultimate objectives of Project Apollo", and which would "provide a general on-board guidance capability for the various earth-orbital and cislunar missions".

Although, by the end of 1961, a great deal of theorizing and experimenting had already been accomplished, and the major Apollo spacecraft contractors had been chosen, a significant unknown remained to be answered: how would men actually land on the moon—and equally important, how would they return to earth? The time had come to forecast the amount of rocket power that could be achieved by the end of the decade, to estimate how much weight the lunar surface could actually support, and to devise a means for leaving the moon after a safe landing.

By early 1962, three types of mission plans were being discussed by NASA planners. These methods were called direct ascent, earth-orbit rendezvous (EOR) and lunar-orbit rendezvous (LOR)

The direct-ascent scheme would place a 150,000-lb manned spaceship directly into lunar trajectory, using the boosting power of a still-to-be developed rocket with an initial thrust of about 12 million lb. From lunar trajectory, the spacecraft would enter lunar orbit; braking rockets would fire and the vehicle would back down toward the lunar surface. The same vehicle would later blast off the surface and land back on earth from an earth orbit. But two problems faced this type of mission.

First, there was considerable doubt that the necessary rocket power could be harnessed by 1970. The so-called "Nova" would have required about twice as much power as any rocket then being discussed. Second, planners were concerned that so large a spacecraft might break through the lunar crust—or, indeed, that its high center of gravity (the spacecraft itself would have measured about 90 ft) would cause it to topple upon landing.

A second method of lunar landing and earth return avoided the requirement of so massive an initial rocket thrust. Earth-orbit rendezvous would have placed two payloads in orbit around the earth. First, a "tanker" rocket would be launched, containing fuel that would eventually be fed into the second payload. After the tanker had achieved its requisite orbit, the second payload would be launched; this would be the manned Apollo spacecraft, propelled by a "Saturn V" rocket whose third stage lacked the liquid-oxygen fuel necessary for the lunar trip. After the payloads had rendezvoused, the spacecraft would dock with the tanker, and the fuel delivery would be accomplished. The advantage of this method was that it involved rocket power then considered likely by the end of the decade. But the same problems of landing on the lunar surface as faced the direct-ascent method still remained.

The third method of lunar landing at first appeared the least likely, probably because it intuitively seemed the most risky. A Saturn V rocket would propel an Apollo vehicle containing three astronauts, plus something new—a detachable craft designed specifically for landing on the moon (e.g., it would possess a low center of gravity and special landing "legs"). After stabilizing in earth orbit, the combined spacecraft and landing vehicle would enter a lunar trajectory and finally stabilize into a lunar orbit. At that point, two astronauts would move into the lunar landing craft, detach it from the mother ship, and descend toward the moon's surface. To rejoin the orbiting Apollo vehicle, the two astronauts would fire rockets for the lunar craft to reinsert into lunar orbit. After the two vehicles had rendezvoused and docked, the astronauts would reenter the main Apollo spacecraft, the landing vehicle would be scuttled, and the Apollo ship would fire its rockets for a return to earth.

The differences between earth-orbit and lunar-orbit rendezvous were immense. EOR plotted a rendezvous in earth orbit before embarking onto a lunar trajectory; LOR involved rendezvous in lunar orbit after the actual landing. The idea of doing a rendezvous (which itself at the time seemed a hazardous maneuver) so far away from earth as planned in the LOR method was initially a frightening proposition. Eventually, however, a team of Langley scientists and engineers demonstrated that, despite outward appearances, LOR would result in substantial savings in earth boost requirements. In addition, it would offer substantial simplification in all phases of a mission—development, testing, manufacture, erection, countdown, launch and flight operations.

With the selection of the lunar-orbit rendezvous method in July 1962, NASA filled in the most significant void then facing the major Apollo contractors. The myriad of scientists and engineers planning for man's eventual landing on the moon could now follow a specific plan. More specifically, the software effort ongoing at MIT at last was able to proceed toward a specific goal. For the most part, conception and development of the Guidance, Navigation and Control hardware did not depend upon the specific mission plan chosen; software, on the other hand, most assuredly had been hampered by the lack of a definitive goal. Landing on the moon and returning via lunar-orbit rendezvous—this was the Apollo mission; the software effort could now begin in earnest.

1.2 Software Programs for the Apollo Missions

The Draper Laboratory's software efforts culminated in a series of flight programs for the Apollo Primary Guidance, Navigation and Control System. Each flight required its own set of software, defined by the mission objectives and constraints. In general, however, the flight programs were comprised of mission programs and routines which remained rather fixed in approach and technique. Thus, such mission programs as rendezvous, targeting and landing are now part of every lunar-landing flight; their underlying techniques are relatively constant, but, in general, control data change with each mission.

Before work could begin on the first flight program—indeed, even before the Apollo mission had been finalized—basic software techniques had to be developed. Many of these early software efforts are briefly discussed in Section 2.2.1. A completed flight program represents the assembly of mission programs and routines. In common parlance, the completed assembly of hard-wire fixed and erasable memory is known as a "rope", a name taken from the weaving process by which the fixed memory is manufactured; the result of this weaving process actually resembles a rope.

An intriguing aspect of the rope developmental history is the means by which the ropes acquired their given names. At first, virtually all of the rope names derived from their association with the name given the entire lunar-landing mission —Apollo: Greek god of the sun*. Those early ropes without "SUN" in their name generally related to astronomical phenomena: thus, ECLIPSE (developed at the time of a major solar eclipse, in 1963), CORONA and AURORA. (RETREAD was an extensively revised version of SUNRISE.) Assigning the early rope names was the treasured prerogative of those most intimately concerned with each rope's development. After the succession of the "SUN" names given the next ropes—SUNDIAL, SUNSPOT, SUNBURST, SUNDISK and SUNDANCE (and SOLARIUM, with its direct sun association, as well)—it became somewhat difficult to differentiate which of the ropes were for the Command Module and which were for the Lunar Module. Accordingly, NASA requested, and MIT agreed, that all Command Module ropes begin with a "C", and all ropes for the Lunar Module with an "L". After a lively intramural competition, the names finally chosen for the LM and CM series were LUMINARY and COLOSSUS, respectively (but not until such names as "Lewis" and "Clark" and "Lemon" and "Coughdrop" had been, for more or less obvious reasons, disqualified).

The following sections summarize the development of flight programs for the Apollo Guidance Computer (AGC). As the result of a NASA decision emanating from a Guidance and Navigation System Implementation Meeting (see Section 1.3.1.2), MIT began digital-autopilot design in late 1964. Two decisions—to integrate an autopilot function into the Guidance, Navigation and Control System, and to enlarge and redesign the AGC—occurred at about the same time, requiring software to fit that computer. Thus, two basic designs of the AGC evolved. Ropes for the earlier, Block I computer, are discussed in Section 1.2.1. The next section discusses the programs developed for the Block II AGC. Section 1.2.3 presents a summary of the Apollo flights, including the names of the flight programs, the launch dates and

*No satisfactory explanation has yet been offered for naming a project aimed at landing on the moon after the sun god. Apollo's sister, Diana (also called Artemis), goddess of the moon, might well feel offended.

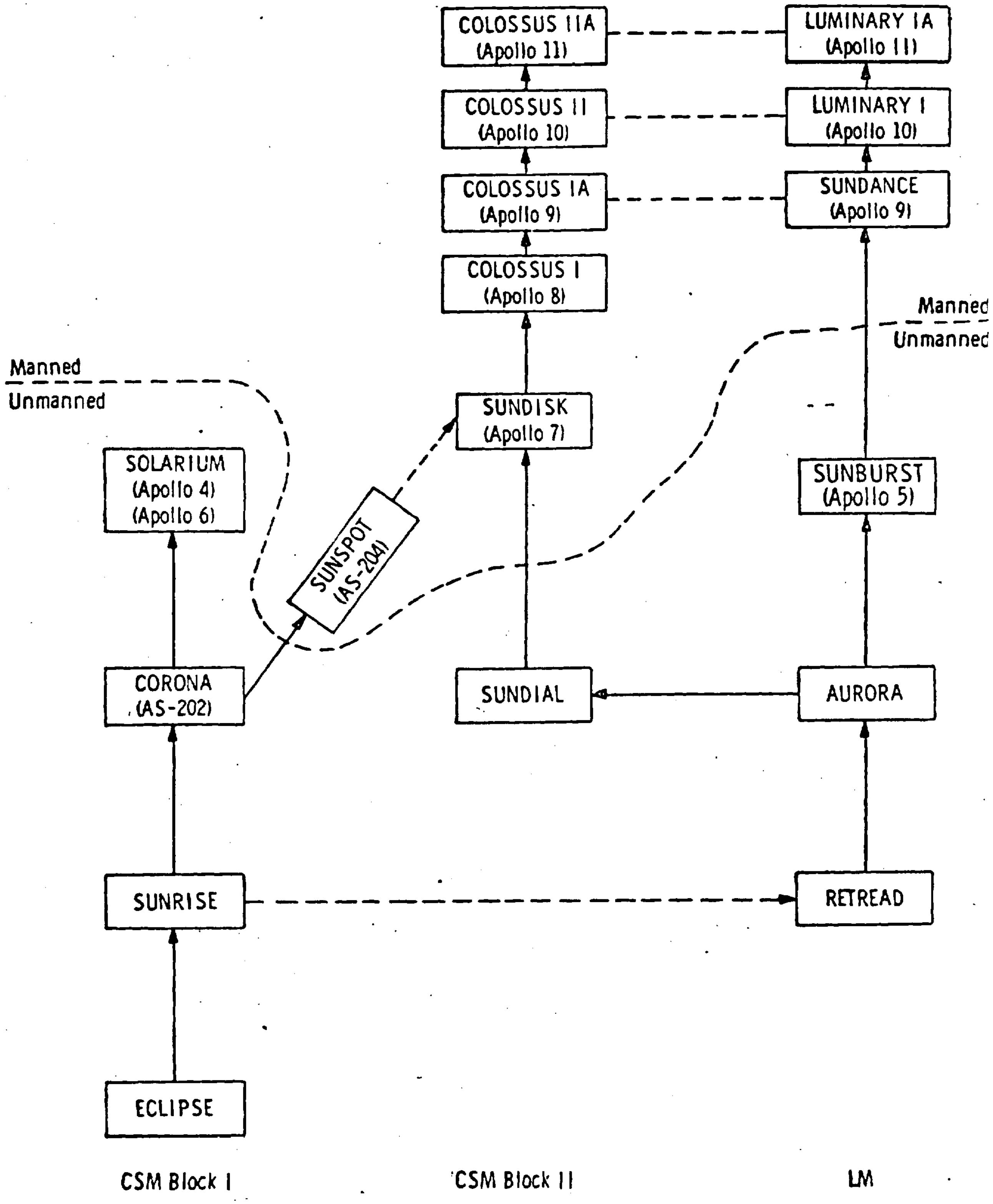


Figure 1.2-1 Rope Tree

crews, and flight descriptions. Figure 1.2-1 depicts the interrelationship of the Block I and Block II ropes discussed in Section 1.2.1 and 1.2.2.

1.2.1 Block I Rope Summary

ECLIPSE is generally ascribed as the first test program designed for use in an early Block I Apollo Guidance Computer. ECLIPSE was, in fact, an assembly of many fundamental routines. It brought together such routines as the Executive, Interpreter and Waitlist. (See Section II for a description of the AGC computer architecture.) In addition, ECLIPSE included Program PINBALL GAME BUTTONS AND LIGHTS, which processes the buttons and illuminates the lights of the spacecraft's Display and Keyboard. Because ECLIPSE was intended only as a test of the Block I AGC, it contained no routines to exercise the Guidance, Navigation and Control System (GN&CS) hardware.

By adding fundamental guidance and navigation functions to ECLIPSE, MIT engineers designed and developed SUNRISE, the first G&N systems-test program for the Block I computer. SUNRISE was the first Block I program suitable for operation in a laboratory-based guidance system. Included in SUNRISE were such G&N-specific routines as an IMU mode-switching program, interface-monitoring programs, down telemetry, and routines to measure gyro-drift coefficients and the bias and scale factors of the three accelerometers. SUNRISE also contained a program for prelaunch alignment. Although not destined for an actual mission, SUNRISE served as a building block for the first flight programs that followed. Programs under development could be interfaced with SUNRISE, and thus tested and changed in a working computer environment.

The program designed for the first Apollo flight was known as CORONA; it was used on the unmanned mission, AS-202. CORONA interacted with an onboard Mission Control Programmer, a series of relays connected to the computer interface to simulate certain astronaut functions. Also, CORONA included an earth-orbital reentry program which served as the model for all future such programs.

Two developmental extensions of CORONA occurred at about the same time. The more straightforward evolution led to SOLARIUM, the flight program for the unmanned missions, Apollo 4 and Apollo 6. SOLARIUM contained few major changes from CORONA, except that rescaling occurred to replace the elliptical trajectory of CORONA with parabolic and hyperbolic reentry trajectories for SOLARIUM.

The second evolution from CORONA led to SUNSPOT, the program intended for what would have been the first manned mission, AS-204. The major change represented by SUNSPOT allowed for elaborate astronaut-interface display programs. Whereas programs were sequenced automatically in the previous unmanned missions, allowing only a certain preordained series of events, SUNSPOT introduced the flexibility of astronaut selection of programs. Most of the automatic sequences provided in CORONA were removed in SUNSPOT.

1.2.2 Block II Rope Summary

To a great extent, the development of programs for the Block II Apollo Guidance Computer resembled the path taken in developing the Block I programs (Fig. 1.2-1). The most obvious differences resulted from the added presence of the Lunar Module (LM), which was to contain an Apollo Guidance Computer identical to that in the Command Module (CM). Following the testing of a Block II program which contained basic guidance and navigation functions, programs for the CM and LM computers evolved simultaneously.

For the initial development of a Block II program, the basic Block I systems-test programs were adapted and assembled into the rope appropriately known as RETREAD. Because the Block II computer contained a larger and more powerful instruction repertoire than that of the Block I AGC, recoding of the basic Block I programs resulted in increased speed and efficiency. Analogous to Block I's ECLIPSE, Block II's RETREAD contained the system-software programs required to test the potential of the computer—Executive, Waitlist, Interpreter. As in ECLIPSE, no provision for mission- or spacecraft-specific programs was included in RETREAD.

From RETREAD evolved the main on-line ropes, beginning with AURORA. In many ways equivalent in purpose to the Block I SUNRISE, AURORA included programs which interfaced with LM GN&CS hardware. AURORA included the monitoring routines for the Inertial Measurement Unit, prelaunch alignment programs, radar-manipulation routines, and various means to control the Display and

Keyboard logic, altitude and altitude-rate meters, and the turn-on and turn-off processes. Like the Block I SUNRISE, AURORA provided a software environment for testing and development of future ropes.

As an offshoot from AURORA, a rope called SUNDIAL tested the GN&C System for the Command Module. SUNDIAL naturally resembled AURORA, except that the LM-specific functions of AURORA were replaced with the CM-specific functions of SUNDIAL. SUNDIAL and AURORA both grew out of RETREAD and they "fathered" two lines of flight programs specific, respectively, to the Command and Lunar Module computers.

The first rope for a manned mission using the Block II AGC was SUNDISK, developed for Apollo 7. Although this program was developed for an earth-orbital flight, it contained many translunar programs in their formative stages. COLOSSUS I, the rope for Apollo 8, the first mission to orbit the moon, included operational cislunar and return-to-earth targeting and navigation programs. Apollo 8 orbited the moon without a Lunar Module, however. CSM/LM rendezvous programs were exercised in earth orbit in COLOSSUS IA, the rope developed for the Apollo 9 mission. COLOSSUS II, developed for the Apollo 10 mission, allowed for the first CSM/LM rendezvous in lunar orbit and included a revised model of the lunar-gravity potential. COLOSSUS IIA, flown on Apollo 11, was virtually the same as COLOSSUS II.

Programs for the LM Apollo Guidance Computer evolved from the early AURORA assembly. SUNBURST was developed for Apollo 5, an unmanned flight test of the Lunar Module and its flight rope. The SUNDANCE rope was developed for the first manned Lunar-Module flight, Apollo 9. Although the Apollo 9 mission was strictly earth-orbital, SUNDANCE exercised lunar-landing, lunar-ascent and rendezvous routines for the first time.

Employing the rope LUMINARY I, Apollo 10 marked the first low pass (to 50,000 ft) over the lunar surface by a solo LM. LUMINARY I represented a refinement of SUNDANCE, and included scaling for the lunar descent. On 20 July 1969, LUMINARY IA finally guided the Lunar Module to its safe touchdown on the moon's surface, thus fulfilling the nation's commitment to a lunar landing in the 1960s.

1.2.3 Overview of the Apollo Flights

Figure 1.2-2 is a summary of the missions flown during Project Apollo, through the flight of Apollo 11. Included are the flight name, the flight program(s) employed, a description of the objectives, the launch date and the crew for each flight. For those flights where two ropes are listed, the first is for the Command Module and the second for the Lunar Module*.

1.3 Control of the Software Effort

This section describes the various means by which MIT's software activities were monitored—internally, through several operating committees; and externally, through formal contact with the customer, NASA. Linking these types of control was the Guidance System Operations Plan—a multi-volumed document that served several functions, including specification control of each succeeding mission flight plan. This document was prepared by the Draper Laboratory for NASA approval, and reflected the technical decisions emanating from internal and external monitoring operations. Section 1.3.1 below discusses external control; Section 1.3.2 describes the Guidance System Operations Plan; and Section 1.3.3 comments upon other types of control, including internal control.

1.3.1 Control by NASA

Much of NASA's control of MIT's software activities occurred in the form of regular series of meetings conducted among representatives of NASA, MIT, North American Rockwell, Grumman Aircraft Engineering Corporation, and other relevant contractors and subcontractors. These meetings served as a vehicle for communications among the prime contractors and the customer, and apparent conflicts were often settled through unhampered discourse. When contractors were unable to agree on technical issues or future directions, NASA would often use the forum of these meetings to issue its decisions on such matters.

*For an insight into all of the phases which comprise a lunar-landing mission, the reader may choose at this time to continue with Section 2.2.2.

Flight	Flight Program Name	Description	Launch Date	Crew
AS-202	CORONA	Suborbital; supercircular entry with high heat load	8-25-66	unmanned
Apollo 4	SOLARIUM	High apogee; suborbital; supercircular entry at lunar return velocity	11-9-67	unmanned
Apollo.5	SUNBURST	First Lunar Module flight; earth orbital	1-22-68	unmanned
Apollo 6	SOLARIUM	High apogee; suborbital; supercircular entry at lunar return velocity; verification of closed-loop emergency detection system	3-4-68	unmanned
Apollo 7	SUNDISK	First manned flight; earth orbital	10-11-68	Schirra Eisele Cunningham
Apollo 8	COLOSSUS I	First manned lunar-orbital flight; first manned Saturn V launch	12-21-68	Borman Lovell Anders
Apollo 9	COLOSSUS IA SUNDANCE	First manned Lunar Module flight; exercise of lunar landing, ascent and rendezvous techniques in earth orbit; EVA (Extra Vehicular Activity)	3-3-69	McDivitt Scott Schweikart
Apollo 10	COLOSSUS II LUMINARY I	First lunar-orbit rendezvous; Lunar descent to 50,000 ft	5-18-69	Stafford Young Cernan
Apollo 11	COLOSSUS IIA LUMINARY IA	First lunar landing (7-20-69)	7-16-69	Armstrong Aldrin Collins

Figure 1.2-2 The Apollo Flights

1.3.1.1 G&N System Panel Meetings

The earliest series of discussions was known as G&N System Panel Meetings. This series occurred from August 1962 through February 1964, under the direction of the Apollo Systems Project Office of NASA/MSC. Participants represented NASA, MIT, North American Rockwell, Grumman and Bellcomm. Throughout this period, three subseries of Panel Meetings met regularly, each focusing on a separate issue: lunar-orbit operations of the Lunar and Command Modules; earth-orbit and cislunar activities of both vehicles; and the reentry activities of the CSM. Through the medium of vigorous discussion and debate, these meetings collated the technical decisions being made in the design and development of the Guidance, Navigation and Control System.

1.3.1.2 G&N System Implementation Meetings

The next set of meetings served to define the required interfaces between the GN&C System and the spacecraft. The Guidance and Navigation System Implementation Meetings were a means of negotiating the Interface Control Documents (ICDs) which were binding upon all contractors. Implementation Meetings focusing on interfaces for the CSM occurred from June 1964 through February 1965. Implementation Meetings responsible for LM interfaces occurred from September 1964 through April 1966. In addition to physical interfaces, among the topics discussed were kinds of data being sent across the interfaces; the formating of data transmission; data rates; and accuracies of data.

The Implementation Meetings monitored the integration of guidance, navigation and control. Out of these discussions came a decision which had a major impact on MIT's Apollo responsibilities. Originally, the Apollo autopilot function had been the responsibility of the Honeywell Corporation, under subcontract to North American Rockwell. The Honeywell autopilot was analog and was deemed by the NASA monitors to lack the flexibility and versatility required for the complex Apollo mission plan. Consequently, NASA directed the Draper Laboratory to develop a digital autopilot which would have none of these limitations. The existing Block I computer hardware did not have sufficient storage capacity to accommodate an addition of such import; however, at about this same period, another significant decision was made to enlarge the computer capacity and at the same time make its computer architecture more powerful than had heretofore been possible. Therefore, through the forum of the G&N System Implementation Meetings, the Block II Apollo Guidance Computer and the digital autopilots were conceived.

1.3.1.3 Data Priority Meetings

As a means of relieving the problem of customer-contractor and inter-contractor communications, the concept of Data Priority Meetings emerged in 1967. The Planning and Analysis Division of NASA/MSC regularly gathers together the flight crews, flight directors, flight controllers, various MSC software, hardware and analytical specialists, and appropriate contractor representatives. There are thus brought into a single room three significant components: those with questions; those with answers; and those with authority to render decisions.

The group meticulously reviews the guidance and control details for each succeeding mission. Data Priority Meetings define how the various data can be used and the priority which can be imposed to effect the nominal and backup execution of each mission phase.

MIT's role is restricted to the Guidance, Navigation and Control System, but this is one of the most complex subsystems in the Apollo spacecraft. MIT's representatives to the Data Priority Meetings oversee the Laboratory's follow-up to each meeting. Questions arising from these meetings elicit formal responses, usually in the form of Mission Techniques Memoes.

1.3.1.4 "Tiger" Teams

A fourth type of NASA control of MIT's software activities occurred thorugh a means less formal than that of an organized meeting. In late 1967, the Flight Operations Directorate of NASA/MSC organized so-called "Tiger" Teams to hasten technical decisions on MIT's rendezvous and display techniques. The Tiger Teams were aptly named, for despite their relatively informal approach, they were extremely effective. The first Tiger Team spent several days in Cambridge in a successful attempt to clarify the rendezvous displays and operations. Display interfaces between the crew and the landing and rendezvous maneuvers were determined, and rendezvous-display compatibility (e.g., scaling, polarity) between the LM and the CM were established. Targeting programs were made consistent from one program to th next. The second Tiger Team addressed itself to the same issues, but since th decisions of greatest import had already been made, its impact was less pervasiv —hence, this Tiger Team was dubbed the "Pussycat" Team.

1.3.1.5 "Black Friday" Meetings

Shortly after MIT evidenced its dismay over the rapidly-saturating fixed memory storage capacity of the AGC, joint MIT/NASA meetings were held to purg the mission programs then under development of any routines deemed "nonessential" Three such meetings took place—on 13 May 1966, 13 January 1967 and 28 Augus 1967. These meetings became emotional because of disagreement about what was in fact, nonessential. Nonetheless, difficult compromises resulted in the curren fixed-storage capacity being reduced sufficiently to allow inclusion of every essentia routine.

1.3.2 GSOP Concept and History

Beginning with CORONA, the computer program for the AS-202 mission, document known as the Guidance System Operations Plan (GSOP) served as th specification toward which the software efforts were directed. Development an control of the GSOP were important activities in planning the release of a flig program. The format for the GSOP evolved through a series of discussions amon key personnel at NASA and the Draper Laboratory. During preparation of th CORONA rope, several alternative mission profiles had been considered: orbita short-ranged suborbital, and long-ranged suborbital. MIT provided NASA wit estimates of navigational difficulty that might be encountered on each type of missio whereupon NASA chose the long-ranged suborbital trajectory. The CORONA GSO represented an integration of inputs from MIT, NASA and North American Rockwe (the manufacturer of the CSM spacecraft), further defining the mechanics of achievin such a trajectory. NASA reviewed the document, modified it where necessary, an finally approved it as the specification for MIT's software effort.

In comparison to the GSOP format which would follow, the AS-202 documen was relatively informal, encompassing in one small volume the same type o information which would later require six separate volumes for each rope. Th

CORONA GSOP discussed the general description of the mission, the logic diagrams defining the operation of the Apollo Guidance Computer, the uplink and downlink that would interface with the guidance system, and the guidance equations and routines which MIT considered of potential interest to NASA.

Further evolution of the GSOP structure resulted from the additional requirements, constraints and capabilities of later missions. For instance, the SUNSPOT rope developed for the AS-204 mission was the first to allow for manned Apollo flight. With astronauts involved for the first time, more time was required for the GSOP discussions, and more personnel participated in the GSOP development. SUNBURST, the rope for the Apollo 5 flight, contained the first routines developed specifically for the Lunar Module, and thus the GSOP for SUNBURST was the first of the LM GSOPs. Beginning with SUNDISK (Apollo 7, CSM) and SUNDANCE (Apollo 9, LM), successive GSOPs generally represented merely changes from the preceding version, and did not require a completely fresh start. Most of the effort in Guidance System Operations Plans currently involves accounting for changes, with relatively little rewriting.

As mentioned above, the GSOP is published separately for the Lunar Module and the Command Module, and is updated with each new program release, thus providing NASA with current and accurate control over the software and system operations. In addition to these functions, the GSOP has served as an internal working document to coordinate the efforts of the various MIT groups, and as a testing guide for simulation personnel. Finally, the GSOP serves as a GN&C software description and a crew training aid for MSC personnel and contractors. A more detailed description of the GSOP is contained in Section III.

1.3.3 Additional Software Control

The Draper Laboratory monitored the incorporation of mission-program requirements into the mission programs through the actions of a Mission Design Review Board (MDRB), a formally-constituted group comprised of the directors of all software groups. Under the direction of each rope's Project Manager, the MDRB approved, internally, all mission-related documentation. The Project Manager was charged with the responsibility for MDRB coordination and participation to ensure proper processing of control documentation. The specific function of the MDRB was to provide a mechanism for internal control and coordination of mission-related activities. Program Change Requests (PCRs) and Program Change Notices (PCNs) were used as interim revisions of the GSOP, and to document departures from the published GSOP until such a time as MSC-approved changes were incorporated in official GSOP revisions. A NASA-comprised group known as the Software Control Board (with representatives of MIT) initiated and approved specific change concepts, whereupon the MDRB would monitor MIT compliance with these changes.

1.4 Man and Machine Loading Requirements

The story of Project Apollo's successful completion represents, in the end, a myriad of individual successes, most of which are based upon an intricately-tuned interaction among men and machines. For its own part, the story of MIT's software-development effort demonstrates the essential interdependence of talented scientists, engineers, mathematicians and technicians with increasingly complex, versatile and powerful computing equipment. As the tempo of the Laboratory's involvement in software tasks changed, these changes were reflected in the number and types of personnel participating in the effort, and in the power and speed of computers which the Laboratory acquired. This section discusses, in general terms, the history and philosophy of MIT's personnel and computing requirements.

1.4.1 History of Man Loading

1.4.1.1 Initial Philosophy

At the beginning of the Laboratory's participation in Project Apollo, a simple philosophy guided the staffing of the software-development group. Essentially, this philosophy placed a premium on engineers and scientists who, in addition to original, conceptual work, would put their own ideas into a form which machines could understand. Thus, in the early days of the Apollo work, there were no "programmers", as such. Instead, engineers and scientists learned the techniques of programming. At this stage, a relatively small group was thought capable of handling what was then considered a practicable task. It was believed that competent engineers with a credible, solid mathematical background could learn computer programming much more easily than programmers could learn the engineering aspects of the effort. The small size of the initial staff dictated that integration of engineering and programming talents in a few individuals would be preferable to attempts at intercommunication by individual engineers and programmers. Thus, the original intent was to have the project's basic core of engineers follow the program through, from conception to actual flight support.

With the passage of time, however, it became clear that the philosophy could best be followed in spirit, rather than in letter. As desirable as it might be to have a staff composed solely of multidisciplinary personnel, it was clearly impossible to shape such a staff beyond a certain size. Individuals talented in both engineering and in computer programming were not readily available. Also, as the software tasks became better delineated, it was apparent that a major underestimation of program-testing requirements had initially occurred. Because the Apollo Guidance Computer has a comparatively small erasable memory, the problem of having various people using the same registers for different tasks, the problem of overlaying memory —these all required extensive precautionary measures to avoid conflict. Optimally, one dedicated engineer/programmer assumed responsibility for ensuring that no erasable-memory conflict occurred, and for integrating the individual flight programs.

1.4.1.2 Creative Use of Subcontractors

Part of the solution to the problems discussed in the preceding section developed through the extensive use of subcontracted personnel. From the very beginning of MIT's participation in Project Apollo, the Laboratory had stressed that its frequent and extensive use of subcontractors would allow it manpower leverage essential to its responsibilities under the Apollo contract. Through the use of subcontracted personnel, the Laboratory would not be required to assemble and disassemble its own staff to meet the time-varying responsibilities of the Apollo program. Subcontractors would serve as a buffer for the Laboratory's staffing requirements. Importantly, Draper Laboratory personnel have traditionally enjoyed the benefits of long-term employment, so the use of subcontractors would permit Laboratory management to carry a mainline staff of a size that would assure maximum security to all personnel. As detailed in Section 1.1, MIT's extensive hiring of subcontractors during the Polaris project had been a strong point in its presentation to NASA in advance of the Apollo program. Thus, when it became apparent that work loads were greater than initially estimated—especially in the areas of testing and verification—subcontracted personnel were made available for virtually immediate deployment.

Throughout MIT's participation in Project Apollo, subcontractors have served in a variety of roles. They have provided a complement to the talents of the Laboratory's own staff. Except in the area of direct administration, subcontractors have played parts in virtually every phase of the software effort, including design, analysis, testing, verification and simulation. Perhaps most significantly, the ready availability of subcontracted personnel facilitated quick solutions to unexpected personnel requirements, since the Laboratory could hire such personnel without necessarily promising any long-term commitment. The costs—direct and indirect —relating to in-house staffing levels were therefore kept to a mimimum throughout.

1.4.1.3 Review of Man Loading

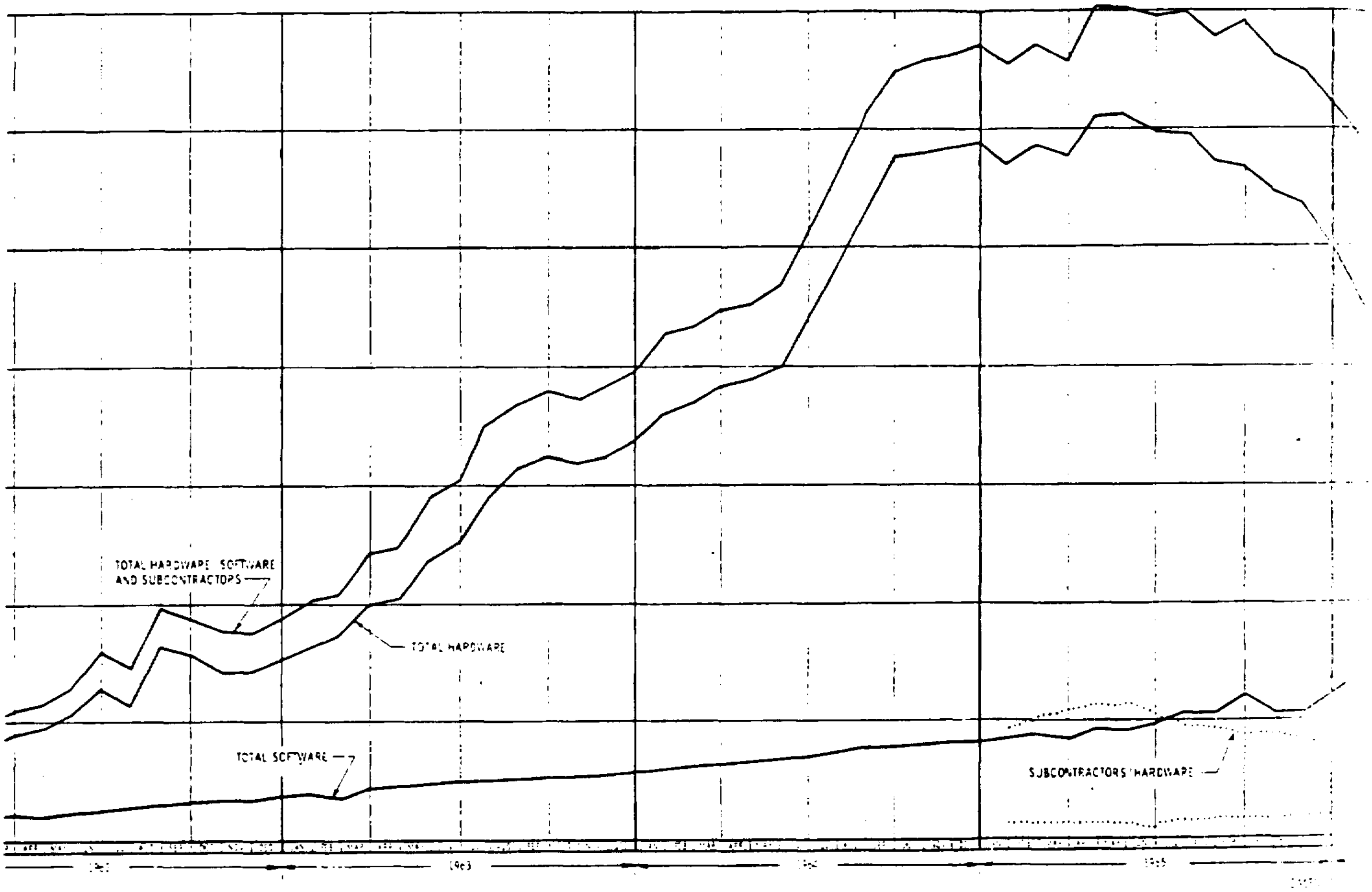
Figure 1.4-1 depicts the man-loading history of the Apollo program at MIT from September 1961 through March 1970. As well as containing a curve for the total personnel levels, the figure shows separate breakdowns for subcontracted hardware and software and total hardware and software levels.

Inclusion of the hardware-personnel figures demonstrates the relative personnel requirements for the hardware and software tasks under MIT's Apollo contract. Thus, the project manpower resources were concentrated on developing system hardware from 1961 through 1965. In 1966, this hardware-development effort rapidly tapered off, and the requirements for designing and developing the mission computer programs increased. In November 1966, the software effort captured precedence as the primary task of the Laboratory's Apollo division.

Figure 1.4-2 demonstrates some of the reasons for the rapid buildup of software personnel. In 1966, no fewer than five ropes were being developed at one time. During the following year, when much of the software buildup had already occurred, six ropes were worked on simultaneously. Figure 1.4-2 is also a milestone chart of the many decisions and events germane to the Apollo software efforts of MIT.

1.4.2 History of Digital Machine Loading

Digital-computation facilities have played a significant role in MIT's development of software for the Primary Guidance, Navigation and Control System of the Apollo spacecraft. As will be discussed in Sections II and III, digital computers


TOTAL HARDWARE, SOFTWARE AND SUBCONTRACTORS
TOTAL HARDWARE
TOTAL SOFTWARE
SUBCONTRACTORS' HARDWARE
1962
1963
1964
1965
1966

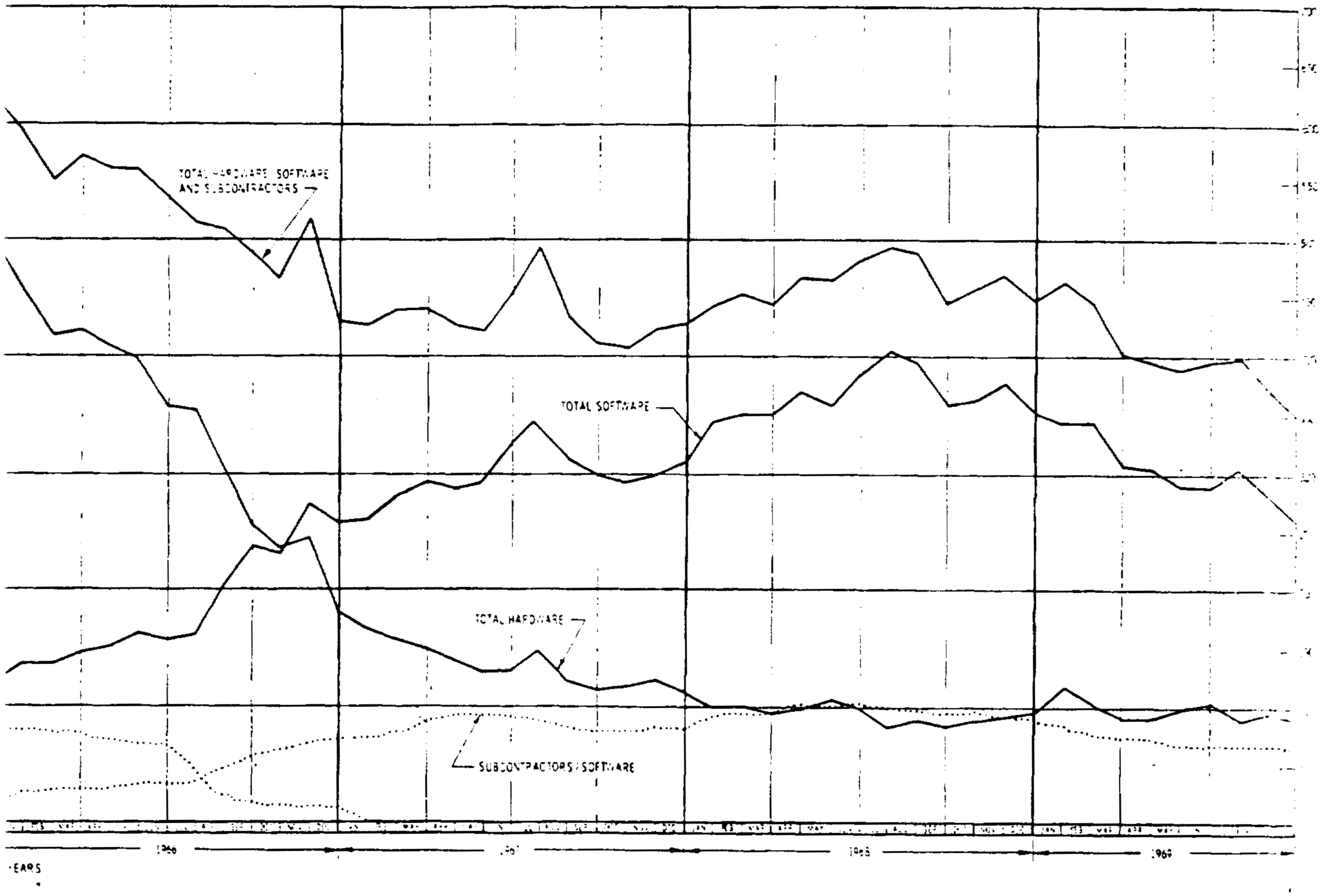
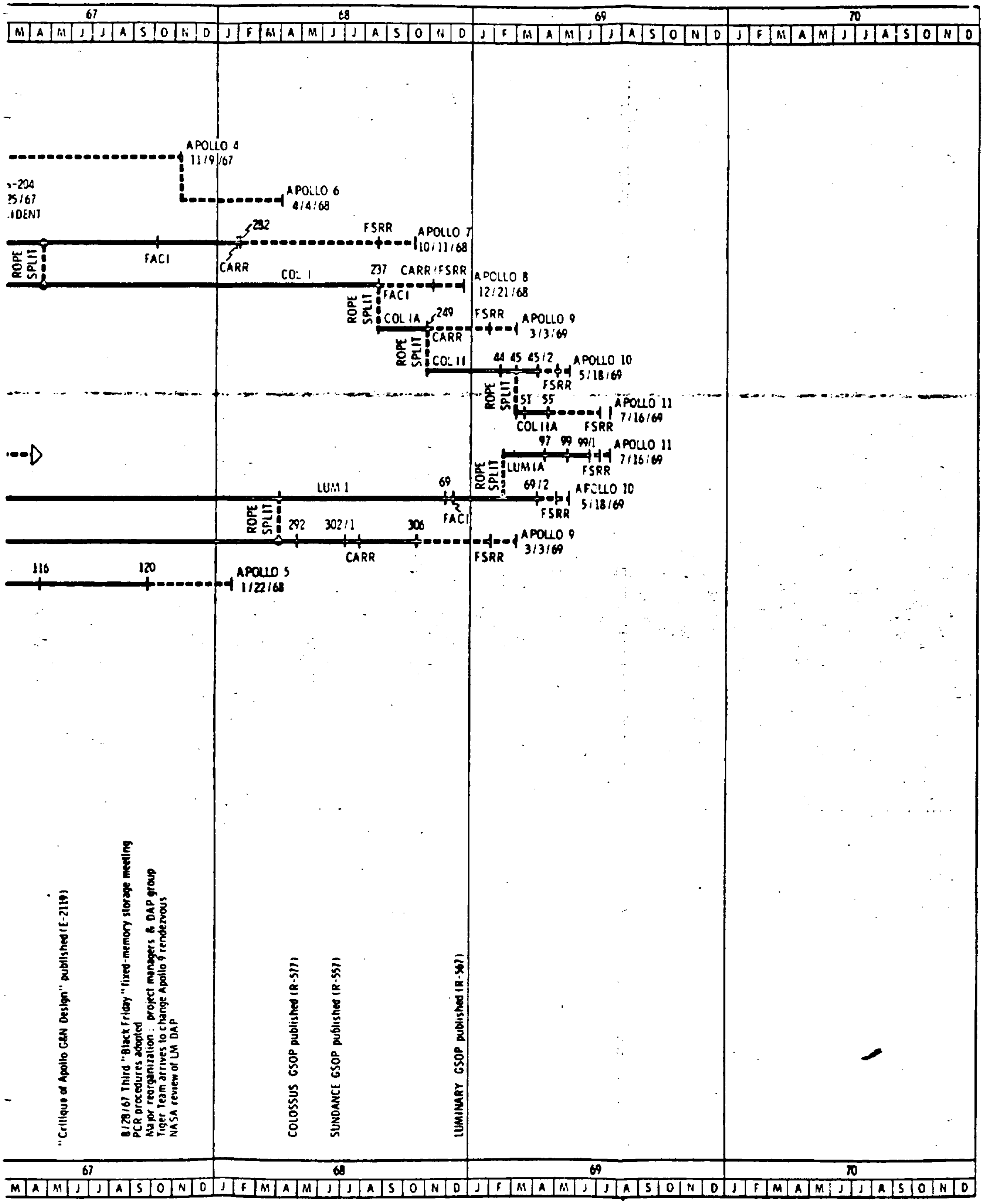
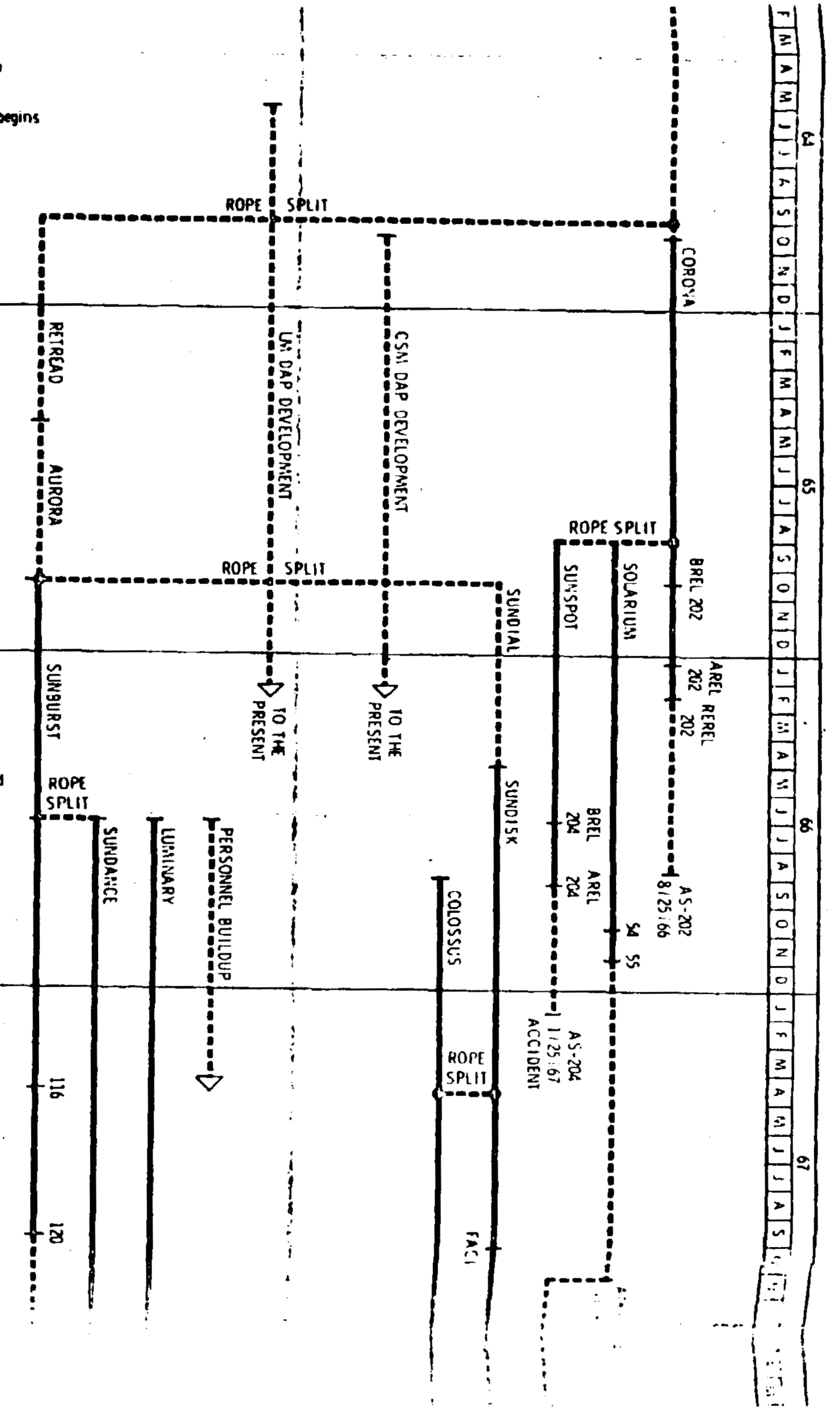


Figure 1.4-1 APOLLO Manloading - Charles Stark Draper Laboratory


67
68
69
70
M A M J J A S O N D J F M A M J J A S O N D J F M A M J J A S O N D J F M A M J J A S O N D
APOLLO 4
11/9/67
APOLLO 6
4/4/68
-204
25/67
IDENT
ROPE SPLIT
FACI
282
CARR
FSRR
APOLLO 7
10/11/68
COL I
237
CARR
FSRR
APOLLO 8
12/21/68
ROPE SPLIT
FACI
COL IA
249
CARR
FSRR
APOLLO 9
3/3/69
ROPE SPLIT
COL II
44
45
45/2
APOLLO 10
5/18/69
FSRR
ROPE SPLIT
51
55
COL IIA
FSRR
APOLLO 11
7/16/69
97
99
99/1
APOLLO 11
7/16/69
ROPE SPLIT
LUM IA
FSRR
LUM I
69
69/2
APOLLO 10
5/18/69
FSRR
FACI
ROPE SPLIT
292
302/1
306
CARR
FSRR
APOLLO 9
3/3/69
116
120
APOLLO 5
1/22/68
"Critique of Apollo G&N Design" published (E-2119)
8/28/67 Third "Black Friday" "fixed-memory storage meeting
PCR procedures adopted
Major reorganization: project managers & DAP group
Tiger Team arrives to change Apollo 9 rendezvous
NASA review of LM DAP
COLOSSUS GSOP published (R-577)
SUNDANCE GSOP published (R-557)
LUMINARY GSOP published (R-567)
67
68
69
70
M A M J J A S O N D J F M A M J J A S O N D J F M A M J J A S O N D J F M A M J J A S O N D

up-to-date

"MILESTONES"



G&N Systems Panel Meetings end • Work on powered-flight guidance, LM terminal trajectory, earth illumination, self-contained capability • Digital-simulation astronaut routine 30 % complete

"Primary G&N System for Lunar Orbit Operations" published (R-446)

G&N Systems Implementation Meetings begin (first CAP meeting) • SUNRISE testing on AGC prototype begins

Block II DAP studies underway • AS-202 GSOP published (R-477)

Improve AS-202 entry-steering equations • RETREAD, Block II test program, complete

New method of orbital navigation

Hybrid Simulator equipment in checkout

RETREAD verification (Executive, Waitlist, Interpreter)

"Design Principles of the LM DAP" published (R-499)

"Block II Instructions" published (AGC-4 Memo No. 9)

"Powered Flight Guidance of the Apollo CSM" published (R-521)

AS-204 GSOP published (R-507)

TVC DAP progressing • NASA management changed from G&C to FOD

Apollo 4 nominal simulation complete • G&N Systems Implementation Meetings end

"Apollo CSM Reaction Control by DAP" published (E-1964)
5/13/66 First "Black Friday" fixed-memory storage meeting • Major reorganization – Group 23B formed

Apollo 5 GSOP published (R-527)
Apollo 4 GSOP published (R-537)

Separate work into subroutines • Apollo 7 GSOP published (R-547)

1/13/67 Second "Black Friday" fixed-memory storage meeting
Verification plans – definitions of levels of testing

"Critique of Apollo G&N Design" published (E-2119)

8/28/67 Third "Black Friday" fixed-memory storage meeting
PCR procedures adopted
Major reorganization: project managers & DAP group
Tiger Team arrives to change Apollo 9 rendezvous
NASA review of LM DAP

APOLLO HISTORY "M

60

Space-study report started in September 1959 is issued

6-month NASA contract to MIT for preliminary G&N design

61

NASA announces Apollo GN&CS contract award to MIT

Software effort rolling-investigating software techniques • Apollo bidders conference in Washington
Earth-orbit rendezvous, launch-timing, translunar-injection studies begin
"Statistical Optimization for Space Flight" published (R-341)

Preliminary computer program, Executive, Waitlist, Interpreter, reentry, guidance, extra-atmospheric abort studies begin

62

"Analysis of Guidance Techniques for Achieving Orbital Rendezvous" published (E-1106)

Industrial support bidders meetings

"Earth Orbital Rendezvous" published (E-1195)
Work in following areas: "MAC" compiler, AGC simulation for H-800, booster-monitoring, integration, reentry, telemetry, rendezvous, orbital navigation, lunar-landing techniques, etc.

Decision to use micrologic in the AGC

NASA announces lunar-orbit rendezvous method • MAC for H-800 complete • "G&N for LM" published (R-373)

G&N Systems Panel Meetings begin • Software & utility programs virtually complete • MIT orders H-1800
MIT study shows LM computer can be virtually same as CM computer
Software effort phases from technique investigation to actual program development
"Universal Formulae for Conic Trajectory Calculations" published (R-382)

First MSC/MIT radar meeting
"Application of Midcourse Guidance Techniques to Orbit Determination" published (E-1261)

63

"A List Processing Interpreter for AGC-4" published (AGC-4 Memo No. 2)

"Backup Thrust Vector Control" published (E-1287)

"Logic Description for AGC-4" published (R-393)

"Radar Requirements for Primary G&N Operations" published (R-404)
"Sampled-Data Velocity Vector Control of a Spacecraft" published (T-355)
MIT rendezvous concept accepted over NAR concept
"Apollo Reentry Guidance" published (R-415)
AGC utility programs are complete

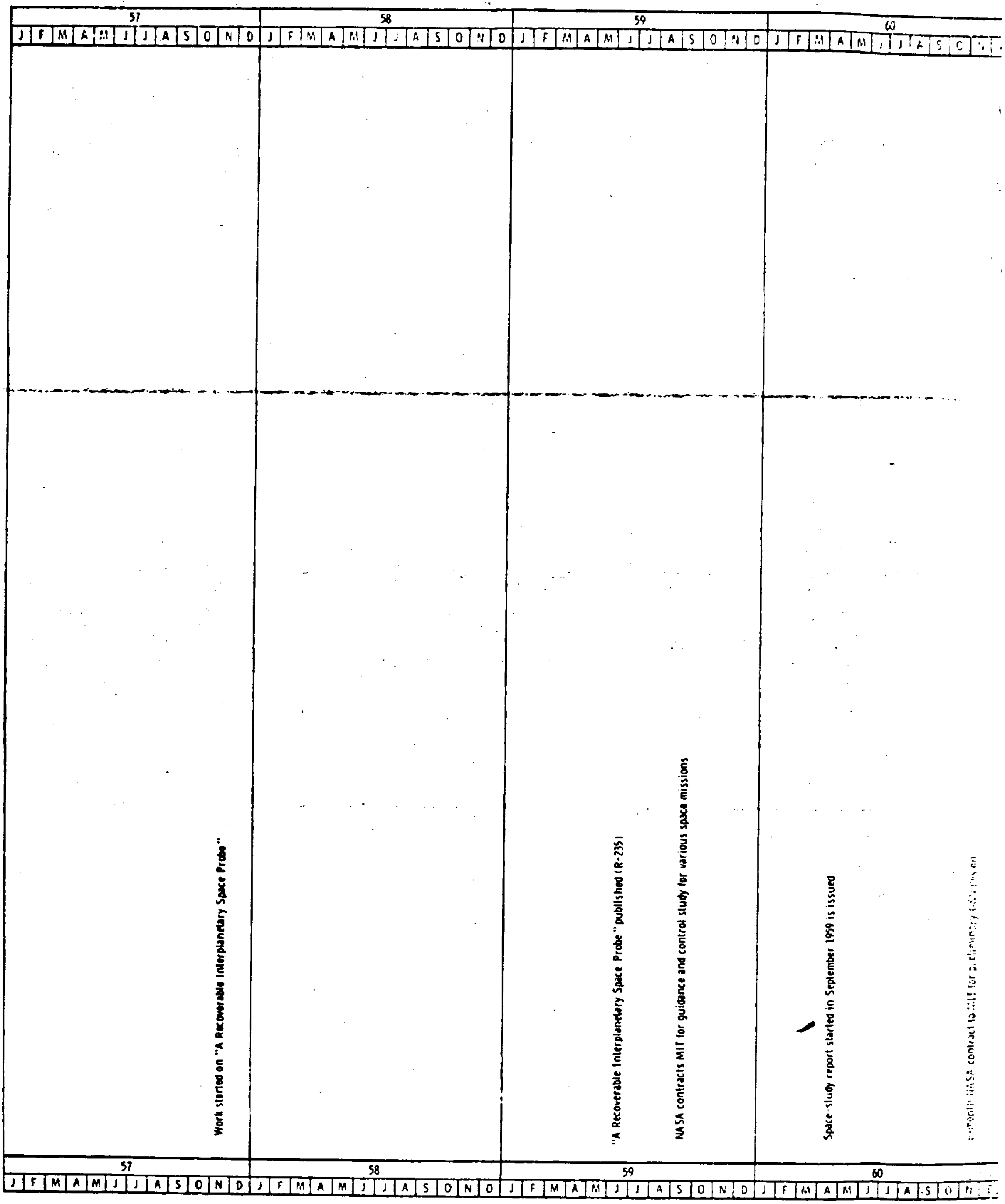
Subsystem test rope "ECLIPSE" ready for use in Systems Test Lab at MIT

SUNRISE systems-test rope being checked out for use at MIT, Raytheon, NAR, ACE, MSC

G&N Systems Panel Meetings end • Work on powered-flight guidance, LM terminal trajectory, earth illumination, self-contained capability • Digital-simulation astronaut routine 30 % complete

ECLIPSE

SUNRISE



Work started on "A Recoverable Interplanetary Space Probe"

"A Recoverable Interplanetary Space Probe" published (R-235)

NASA contracts MIT for guidance and control study for various space missions

Space-study report started in September 1959 is issued

6-month NASA contract to MIT for preliminary design on are used in simulating the Apollo Guidance Computer during design, verification and testing of software. (In addition to this so-called "All-Digital Simulator" function of the digital computer, it serves as a basis for the Engineering Simulator, also described in Section III. A Hybrid Simulator and a Systems Test Laboratory also assisted in the test and verification of computer software and are also discussed in Section III.) This section discusses, in chronological sequence, the four types of digital computers around which the Draper Laboratory fashioned its digital-computation facilities. These computers are the IBM 650, the Honeywell 800, two Honeywell 1800s, and two IBM 360/75s.

During the period in which MIT has participated in the Apollo program, the computing facilities described in this section have served other Draper Laboratory groups in addition to the Apollo division. However, Apollo activities have accounted for about 90 percent of the total use of these facilities.

1.4.2.1 IBM 650

When the Draper Laboratory received its first contract from NASA, in September 1959, an IBM 650 provided the Laboratory with its in-house computing capability. The IBM 650 was a 2000-word-drum central processor, with 60 words of core storage. One tape drive and a disc bar were the only pieces of peripheral equipment. Programmers would write in MAC*, and the IBM 650 was used primarily to compile these programs for computation on much faster and more powerful outside equipment, such as the IBM 704, 709 and 7090. Toward the end of 1959, the burden of the NASA pre-Apollo workload, added to the much larger workload of the Laboratory's Polaris project, stimulated investigation into the possibility of providing additional in-house equipment to accommodate all the work then done by the IBM 650 and the outside rented machines.

*MAC is a high-level programming language for general-purpose computers, developed at MIT for scientific application. It is not to be confused with MIT's Project MAC. The latter was named independently some years later and is unrelated to the MAC language.

1.4.2.2 Honeywell 800

The Honeywell 800 was ordered during Summer 1960, with delivery occurring in December 1961. Based upon the workload of mid-1960, the H-800 was predicted to run about 4 hours per day and to cost no more than the previous total of in-house IBM 650 and outside rented time. By the time the H-800 was placed in production* —in May 1962—it was apparent that even greater speed and power were necessary. Rather than the expected four hours per day, two operator shifts (16 hours/day) were required for the initial H-800 workload. Despite the unexpected demands which the H-800 faced immediately upon being placed in production, it represented approximately a threefold increase over the capabilities of the IBM 650.

To overcome the inadequacy of the Honeywell 800, two approaches were undertaken simultaneously in mid-1962. First, additional memory and peripheral equipment were acquired for the H-800; second, an order was placed for the Honeywell 1800, with expected delivery 18 months later.

The Honeywell 800 had been delivered with a 16,000-word memory, each word having 48 bits. It included a printer, six tape drives and a card reader/punch. To upgrade the H-800 while awaiting delivery of the H-1800, the memory was doubled, additional tape drives and a printer were acquired, and a disc file and a graphic plotter were added.

1.4.2.3 Honeywell 1800

Honeywell's 1800 possesses a 2-μsec access-to-memory, while the H-800's access was on the order of 6 μsec. The H-1800's delivered memory size was 32,000 words, double that of the H-800. These capabilities rendered the delivered H-1800 roughly three times as powerful as the H-800.

Although the Laboratory's H-1800 was delivered in January 1964, it was not until the following May that the system was in total production. In the meantime, failures in hardware necessitated total replacement of the machine's memory. As

* "In production" implies that the equipment is capable of a complete AGC simulation. Computers were "in operation" before they could be "in production".

a result of these difficulties with the new system, between the months of January and May, no in-house digital-computing facilities were available, since the H-800 had been removed to provide space for the H-1800. Consequently, time was rented on outside equipment during this period.

By October 1964, it was becoming apparent that the H-1800 was computing much more rapidly than its peripheral equipment could provide input and output. At that time, Honeywell announced its Model 200 computer, a small machine that could do much of its own computation, could provide its own input and output, and could serve as a buffer for the much more powerful H-1800. MIT ordered a Model 200 for delivery in October 1965.

Two decisions were reached in Summer 1965 regarding the need for additional computing facilities: a second Honeywell 1800 was ordered in June; and a study was begun of the potential advantages offered by even more powerful computers. The second H-1800 was delivered and placed in production in March 1966. The investigation into other computers resulted in the Laboratory's decision to order an IBM 360, Model 75.

The original H-1800's memory had been increased in size from 32,000 to 48,000 words. The second H-1800 was delivered with the larger memory. By the time of the second H-1800's acceptance, a second Model 200 had also been acquired. The final upgrading of the H-1800 facilities occurred with the delivery of a Honeywell Model 2200, a system approximately equivalent to two Model 200s. It was estimated that the addition of the Model 2200 increased the capability of the H-1800 facilities by about 20 percent.

1.4.2.4 IBM 360/75

When the Summer-1965 study of large computing systems began, several systems were under consideration. One was highly valued, but doubts existed that it would ever be manufactured. Another system was by far the fastest machine under consideration, but Laboratory officials were concerned that internal parity checking would not reach the standard necessary to ensure the safety of astronauts —the ultimate customers of the Laboratory's services. Still another system was rejected primarily because it did not allow eventual expansion into an even larger system. Finally, the IBM 360, Model 75 (360/75) was chosen because of its relatively high speed, its degree of internal error checking, and the availability of the more powerful Model 91, should the need for expansion occur. It was estimated that a single IBM 360/75 would be roughly equivalent to four Honeywell 1800s.

The IBM 360/75 was delivered in October 1966, and it became operational two months later. During the first eight months of operation, three basic activities consumed most of the machine's availability: MAC language was adapted for the 360/75, system software was developed, and simulation software was implemented. During these first months of IBM 360/75 operation, it was concluded that the CPU's 512,000-byte memory would not suffice for simulation purposes; memory size was thereafter doubled. Not until September 1967, about ten months after delivery, was the IBM 360/75 in total production for general simulations.

By the time the IBM 360/75 came into total production, the need for a second IBM 360/75 was already recognized. Accordingly, the Honeywell 1800s would be removed. Removal of the second-delivered H-1800 occurred in December 1967, and the original H-1800 was removed in April 1968 to make way for the second IBM 360/75, to be delivered the following month. Thus, during the last quarter of 1967, three complete systems were in operation— the IBM 360/75 and the two H-1800s.

The second IBM 360/75 was placed in total production a mere two weeks after delivery, primarily as a result of the experience gained through the lengthy break-in procedures on the first IBM 360/75. By the time the second system was placed in production, the peripheral equipment originally delivered had also been expanded in power and capacity. For instance, the six original IBM 2311 disc packs were increased to ten. Two printers were added to the original two, and additional tape drives and a card reader were acquired. Finally, three IBM 2314 disc packs were gained, each of which was roughly equivalent to four IBM 2311s.

In August 1969, following Apollo 11's successful lunar mission, the second-delivered IBM 360/75 was removed, thus leaving the original IBM 360/75 and the systems's peripheral equipment as the remaining digital-computing facility of the Draper Laboratory. Although the remaining IBM 360/75 was deemed adequate for the needs after the lunar landing, within seven months it also reached saturation.

1.4.2.5 Loading of the Digital Computing Facilities

Figure 1.4-3 charts the monthly load which was logged on the Laboratory's digital-computing facilities, expressed in equivalent Honeywell-1800 hours. In this figure, monthly saturation of a Honeywell 1800 is 660 hours (=22 hours/day x 30 days). Since the Honeywell 800 was roughly a third as powerful as the H-1800, saturation of the H-800 occurs at 220 hours/month. The IBM 650 was, in turn, about one third as powerful as the H-800—or a ninth as powerful as the H-1800; thus on this graph its saturation is 73.3 hours/month. The IBM 360/75 is roughly four times as powerful as the H-1800, and thus its saturation occurs at 2640 hours/month. This figure also indicates the dates of computer acquisitions and removals.

1.5 Major Recurrent Problems

With the manned lunar landing and return accomplished in July 1969, Project Apollo met the national goal enunciated eight years earlier. Through its design, development and implementation of the Primary Guidance, Navigation and Control System for the Apollo spacecrafts, MIT's Draper Laboratory shared in that eminent success. Along the course of its participation in the Apollo adventure, MIT experienced the kinds of technical and managerial difficulties that can only be expected in undertaking so massive a program—but that nevertheless create uneasiness at the time of their occurrence. This section focuses on the two problems which caused the greatest difficulty in the software effort. Difficulties were encountered in the estimate of time and manpower schedules and in the control of accurate, up-to-date spacecraft data. Both of these problems continually plagued MIT's software efforts, since neither their cause nor their solutions could be found within the Laboratory, alone; ultimate solution would require an extraordinarily well-tempered orchestration among NASA and all of its contractors and subcontractors.

1.5.1 Difficulty in Estimating Time and Manpower Schedules

Throughout much of the Apollo software effort at MIT, managers have experienced difficulty in estimating the time and manpower requirements to design, test and verify the successive mission-flight programs. At the commencement of

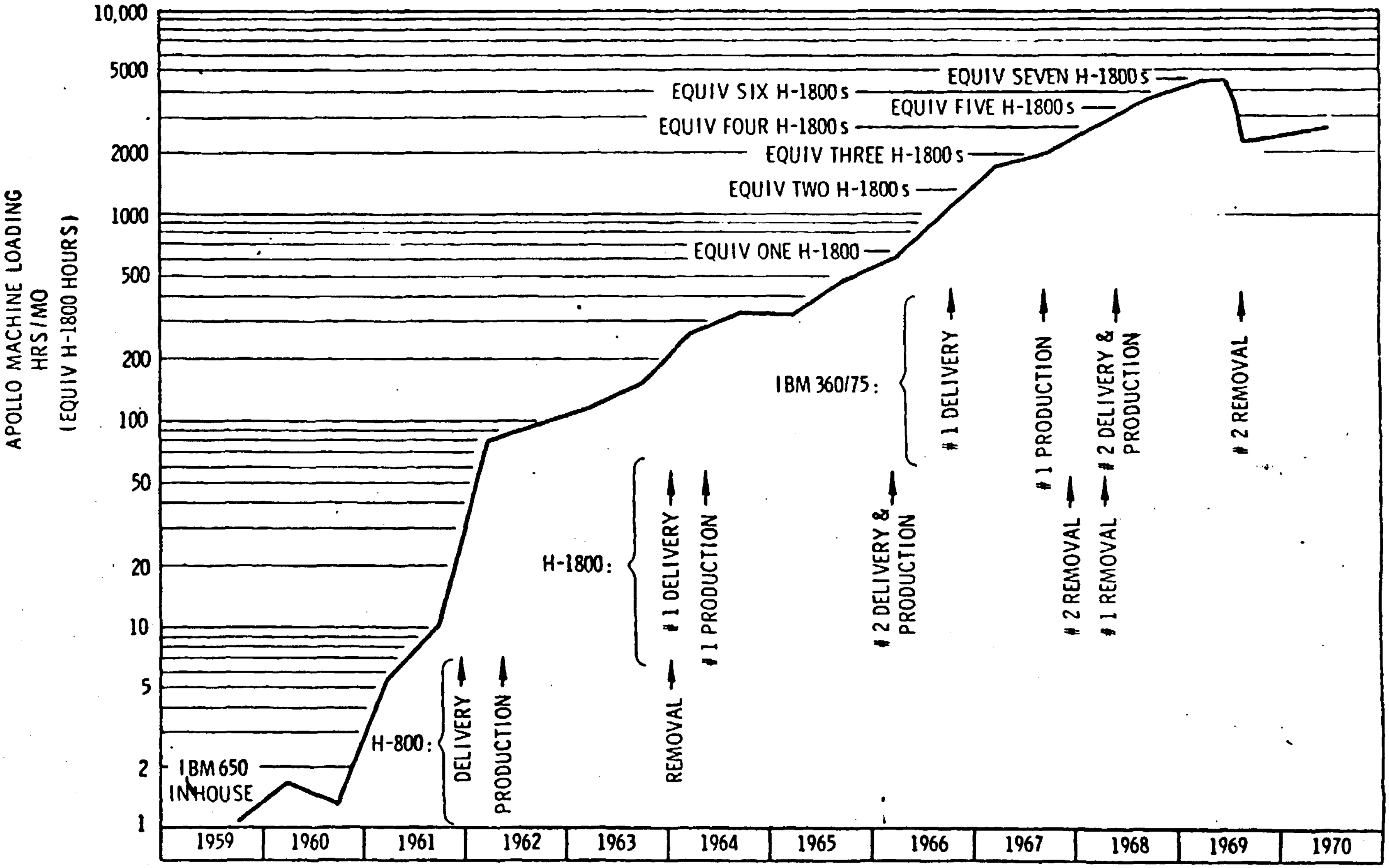


Figure 1.4-3 Apollo Machine Loading

work on a new flight program, it is advantageous—perhaps even essential—to break down the total required effort into a series of smaller tasks, each fitting into a preplanned sequence of steps leading to the required whole. Specialists in each of the subdivided tasks can then be assigned stated responsibilities within a specified time constraint. This description fits the optimal situation—the situation in which the Draper Laboratory more nearly finds itself today than it has in the past.

It is more likely that at the commencement of work on an entirely new mission-flight program, the separate tasks required to lead to the assembled program cannot be known in advance. Indeed, this was the case with virtually every program up to the revisions in COLOSSUS and LUMINARY which currently suffice in the planning of new missions. Part of the development process includes the understanding of what these basic steps should be. In brief, at the beginning no one can forecast all the little pieces which will eventually be required, and thus predicting accurate work schedules is almost a priori an impossible task.

Another probable cause of this overall scheduling problem is that subtasks required an ordered interrelationship. Not all of the tasks could occur simultaneously; some took precedence over others, and certain later tasks could not proceed until the completion of earlier tasks. In other words, the entire sequence of tasks could be completed no sooner than the time required to complete perhaps a certain few "pacemaker" tasks. Perhaps the most difficult estimate to be made in advance is the amount of time required for iteration and retests. Thus, to adequately forecast accurate work schedules, the manager would have had to predict not only all the necessary subtasks, but, in addition, which few of these subtasks would be the pacemakers and which would later be redesigned and require further testing.

Another cause of the work-schedule problem relates to the vagaries of personal dynamics. Throughout much of the software effort, management encountered a problem of deadline definition; that is, when a deadline for rope release became known, a number of intermediate deadlines or goals had to be established, particularly for pacemaker tasks, to ensure that the final deadline be met. After all of the deadlines had been assigned, it was sometimes difficult to convince software personnel of the importance of meeting the earlier deadlines; the tendency was strong for those with the earlier tasks to aim toward the deadline for the completed flight program. Consequently, management was continually required to reemphasize the importance of meeting each assigned deadline.

A final cause of the work-schedule problem also relates to the area of human dynamics. The communication of "bad news"—e.g., news of imminent delays—slows as it goes up the line of management. This difficulty derives from the basic human drive to prefer the communication of positive tidings to that of negative findings. Both the bearer and the receiver of bad news feel uneasy with the experience, but management must encourage its personnel to communicate the bad with the good. When the person responsible for one of the subtasks recognizes that his schedule must slip, it is human nature to defer passing along word of the delay. As this one piece of bad news progresses up the ladder of administrative responsibility, communication of the bad news is further impeded. As the initial step in rationalization, each person along the line attempts to discover for himself whether the bad news is as bad as anticipated—or if, perhaps, some degree of overstatement has occurred. Only through conscious recognition of this process by all personnel can this problem be alleviated.

Thus, four separate causes combined to render the estimation of work schedules an especially vexing problem:

a. the difficulty of predicting all of the required subtasks;

b. the difficulty of pinpointing and hastening pacemaker subtasks;

c. the difficulty of meeting deadlines for individual subtasks;

d. the difficulty of communicating "bad news" quickly through the line of management.

As MIT gained experience through its successive responsibilities in the Apollo program, the work-schedule problem became increasingly more routine—and less annoying. Nevertheless, small remnants of this problem continue to cause occasional difficulties in the scheduling of current ropes.

1.5.2 Control of Timely Spacecraft Data

The second major problem encountered by MIT software planners relates to the acquisition of complete and up-to-date data on spacecraft parameters. In the design, verification and testing of guidance, navigation and control software, it was essential that the responsible MIT engineers possess the most current data obtained by other NASA contractors in the development of the spacecraft components. From the beginning, it was clear that a mechanism for such data exchange was of prime importance.

One of the initial responsibilities of liaison personnel was the development of a data-exchange mechanism. For instance, North American Rockwell's liaison with MIT was to record the most up-to-date information on the Command and Service Modules, and the liaison from Grumman was to do the same for the Lunar Module. In practice, however, this official mechanism broke down quickly, since spacecraft engineers were reluctant to formally release data on parameters still undergoing development, measurement or testing. Such virtually universal reluctance to commit preliminary data, even to discretionary use, rendered the officially-recognized channels rather dinosauric in current-information content. During years of effort to establish a smoothly-functioning, up-to-date data-exchange program, MIT software personnel resorted to other means for learning the parameters and tolerances to which they should design their software and simulations.

As MIT software personnel became acquainted with their peers at the other relevant spacecraft contractors, an informal network of data exchange developed. Rather than relying upon the official mechanism of liaison contact, the engineers responsible for the development of software would place strategic telephone calls to learn up-to-the-minute data being used in the development of the spacecraft systems. Although this informal method of data exchange possessed the disadvantage of consuming much valuable time, it produced the distinct benefit of collecting the most timely information available.

In an attempt to formalize the person-to-person method of data control, a "Data Book" which listed current data was organized at MIT. There were two sections within this document: class A data, which were official and verified by an authority at the originating contractor; and Class B data, the type generally received through telephone and person-to-person communication, but which lacked official verification. But the Data-Book mechanism required personal enthusiasm for the task of collecting data—enthusiasm which virtually all dedicated software engineers feel should better be devoted to the task of designing software.

All of the parameters and tolerances to which the software and simulations were designed were published in Chapter 6 of the GSOP. (See Section 3.2.1 of this report.) In this fashion, the Laboratory kept NASA continuously and officially apprised of MIT's current information—information which could be approved along with NASA's general approval of GSOP revisions.

By no means has the problem of timely data control been solved, but solely because of MIT's increased familiarity with the spacecraft components, it has become somewhat less of a problem. Just as there were elements of human dynamics in the problem of time and manpower scheduling above, so, too, did personal vagaries play a role in this difficulty: people are unwilling to divest themselves of data which they consider not yet final. And the very qualities of technical competence and conscientiousness which one needs to invest in the area of data exchange are difficult to come by, since individuals so endowed generally prefer to apply these qualities in the actual software development.

SECTION II

AGC SOFTWARE

This section describes the software which controls the present LM and CM guidance computers. The computer is the heart of the Apollo Guidance, Navigation and Control System. The software maintains positional knowledge of the vehicle in space, determines the path to a desired destination, and steers the spacecraft along that path by sending commands to the engines. It communicates with the astronauts and the ground, and monitors the performance of the GN&C System.

Mission programs, such as rendezvous, targeting and landing, control some of the phases of an Apollo flight. However, before these can be discussed, it is necessary to examine the underlying computer organization which allows the mission program to operate. Thus, Section 2.1 describes the basic machine architecture, the Executive and service programs which control AGC operations, and the input/output functions which allow the computer to monitor the GN&CS and to communicate with the astronauts and the ground. Although the CM and LM computers satisfy different mission requirements, the underlying system software is quite similar for the two vehicles. Hence Section 2.1 presents a generalized Apollo Guidance Computer, and specific differences are noted when they apply.

Section 2.2 includes a general description of all the phases of the Apollo mission and of the major flight tasks required for that mission. The design effort which produced these mission programs has been a long and challenging task. This report will not attempt to give a complete discussion of this effort, since it has been documented in other sources; however, the rope design philosophy and the problems encountered as it finally evolved are discussed in Section 2.2.4, and the major program capabilities are described in somewhat greater detail in the appendices to this report.

2.1 Computer Capabilities

2.1.1 Storage and Manipulation of Computer Instructions

The AGC contains two distinct memories, fixed and erasable, as well as various computer hardware. The fixed memory is stored in a wire braid which is manufactured and installed in the computer. This memory cannot be changed after manufacture and it can only be read by the computer. Fixed memory contains 36,864 "words" of memory grouped into 36 banks. Each word contains 15 bits of information (a sixteenth bit is used as a parity check). The word may contain either a piece of data, or an instruction which tells the computer to perform an operation. A series of instructions forms a routine or a program. In addition to storing programs, the fixed memory stores data such as constants and tables which will not change during a mission.

The erasable memory makes use of ferrite cores which can be both read and changed. It consists of 2048 words divided into 8 banks. Erasable memory is used to store such data as may change up to or during a mission, and is also used for temporary storage by the programs operating in the computer.

Included in the hardware is a Central Processing Unit (CPU). The CPU performs all the actual manipulation of data, according to the instructions designated by a program. The 34 possible machine instructions include arithmetic operations (add, multiply, etc.) as well as logical operations, sequence control, and input/output operations. Also included are a limited number of "double-precision" instructions which permit two words of data to be processed as a single "word" of greater precision.

The memory cycle time (MCT) in the AGC is 11.7 μsec. Most single-precision instructions (e.g., addition) are completed in two MCTs; most double-precision machine instructions are completed in three MCTs. The unconditional transfer-control instructions, however, operate in one MCT.

To be used as an instruction, a computer word must specify the operation to be performed and give the location of the data to be operated on. However, a 15-bit word does not contain enough information to specify 34 operations and 38,912 fixed and erasable locations. In fact, 15 bits cannot even specify 38,912 locations unambiguously. It is for this reason that both the fixed and the erasable memories are grouped into banks. An instruction may specify any address within its own bank, and may also address the first four banks of erasable and the first two banks of fixed memory. Access to other banks is accomplished using bank-selection registers in the CPU. In many cases a program exists entirely within one bank memory, in which case bank switching is not required.

Many of the tasks the AGC performs can be adequately carried out by machine instructions. However, for extensive mathematical calculations—in such areas as navigation—the short word length of the AGC presents difficulties. It limits the number of instructions available, the range of memory that can be addressed without switching banks, and the precision with which arithmetic data can be stored and manipulated. To alleviate these problems, nontime-critical mathematical calculations are coded in "interpretive language" and are processed by a software system known as Interpreter. Each Interpreter instruction is contained in two or more consecutive computer words. The increased information available allows more possible instructions and a greater range of memory addressable without bank switching. In fact, with some exceptions, all of erasable memory and fixed memory may be addressed directly. Among the available Interpreter instructions are a full set of operations on double-precision quantities, including square root and trigonometric functions, some triple-precision instructions, and a set of vector instructions such as cross product, dot product, matrix multiply, and vector magnitude. Interpreter routines translate an Interpreter instruction into an equivalent series of machine instructions to be performed by the CPU. Thus, one Interpreter instruction may be equivalent to many machine instructions, and much storage space is saved in the computer. The Interpreter also contains software routines for the manipulation and temporary storage of double- and triple-precision quantities and vectors.

Interpreter expands the processing capabilities of the CPU hardware. However, its operation is quite slow, since the CPU must perform all the actual operations, and much time is spent in the translation of instructions and the manipulation of data. Although processing time is slower, much storage space is saved in fixed memory by the more powerful Interpreter instructions; thus, the vast majority of nontime-critical mathematical computations are coded using interpretive language.

2.1.2 Timing and Control of the Computer

Two of the more stringent requirements placed upon the AGC are the need for real-time operations and the necessity for time-sharing of multiple tasks.

Certain computer functions must occur in real time. For example, certain input data must be stored or processed immediately upon receipt; and outputs, such as those which turn the jets on and off, must occur at precisely the correct time. An interrupt system causes normal computer operation to be suspended while performing such time-critical tasks.

Several programs, which are less time-critical, may all be required during a phase of the mission. Time sharing between these programs is controlled by a software executive system which monitors the programs and processes them in order of priority. The Executive can stop one job when a higher priority job is necessary, then resume the low-priority job when time is available.

2.1.2.1 Interrupt System

To permit quick response to time-dependent requests, the AGC has a complex interrupt structure. There are two classes of interrupts, counter interrupts and program interrupts. Counter interrupts have the highest priority of all AGC operations. Counters are locations in erasable memory which can be modified by inputs originating outside the CPU. Some counters are used as clocks, while others interface with spacecraft systems to receive or transmit sequences of data pulses. The counters respond to a set of involuntary instructions called counter interrupts, which may increment, decrement, or shift the contents of the counters. A counter interrupt suspends the normal operation of the CPU for one MCT, while the instruction is being processed. Except for the short time loss, the ongoing program is not affected by the counter interrupt; in fact, it is not aware that the interrupt has occurred. These interrupts are used solely for counter update and maintenance; their priority assures that no information will be lost in the counters.

The use of counters as input/output devices will be described in Section 2.1.3.1; it is appropriate now, however, to discuss the six counters which are used for timing purposes. Two counters, designated TIME1 and TIME2, form a double-precision master clock in the AGC. TIME1 is incremented at the rate of 100 counter interrupts per second. Overflow of TIME1 triggers a counter interrupt to increment TIME2. Since total time that must elapse before TIME2 overflows exceeds 31 days, TIME1 and TIME2 are thus able to keep track of total elapsed mission time.

The remaining clock-counters, designated TIME3 through TIME6, measure time intervals needed by the AGC hardware and software. For example, autopilot computations must be processed periodically whenever the autopilot is in use. Before reaching completion, these computations preset the TIME5 counter so that it will overflow at a specified time in the future. TIME5 is incremented at the rate of 100 counter interrupts per second. When TIME5 overflows, a signal sent to the CPU causes a "program interrupt" which interrupts the program in process and begins the autopilot computations once again.

Program interrupts have lower priority than counter interrupts, but greater priority than normal program operation. Unlike counter interrupts, the purpose of program interrupts is to alter the normal processing sequence. There are 11 program interrupts; they may be triggered by a clock-counter overflow, as in the example given above, or by externally generated signals, such as the depression of a key on the Display and Keyboard (DSKY) by an astronaut. The occurrence of a program interrupt causes the computer to suspend normal operation at the end of the current instruction. The current CPU data are saved, the computer is placed in interrupt mode, and control is passed to a preassigned location in fixed memory. This preassigned location is the beginning of a program which performs the action appropriate to the interrupt. While the interrupt program is running, the computer remains in interrupt mode, and no additional program interrupts will be accepted, although counter interrupts can still occur. (Requests for other program interrupts are stored by the hardware and processed before returning to normal operation.) At the conclusion of the interrupt program, a "resume" instruction is executed. If there are no other program interrupts, the CPU is taken out of interrupt mode, the original contents are restored, and the program returns to the point at which it was interrupted. One program interrupt (restart) takes precedence over all the others, and can even interrupt an interrupt. It results from various kinds of computer malfunctions. (This interrupt will be discussed in Section 2.1.4.)

A computation which takes place by means of a program interrupt is called a task. Since tasks may not be interrupted, they must be short to avoid delaying other tasks. This speed requirement precludes the use of interpretive language.

One class of tasks is initiated by overflow of time counters TIME3, TIME4 TIME5, and TIME6. These are considered time-dependent tasks. The TIME5 interrupt, described above, initiates autopilot computations at precise periodic intervals. TIME6 controls the timing of the autopilot RCS jet firings. TIME4 initiates a series of routines which periodically monitor the IMU, radar, etc., and process input/output commands. The TIME3 counter is under the control of the software executive system (described below). It is available for general use by any program needing to schedule a task for a specific time.

A second class of tasks is initiated by interrupts caused by external action. For example, depressing a DSKY key initiates a task that begins processing DSKY readings and storing the information for later processing. Telemetry and the radar also cause interrupts that initiate tasks to receive or transmit the next data word.

2.1.2.2 Software Executive System

Computation in the AGC is managed by a software executive system comprised of two groups of routines, Executive and Waitlist. This system controls two distinct types of computational units, jobs and tasks. In its normal operating mode, the computer processes jobs. These are scheduled by the Executive, according to a priority system. The Waitlist uses the TIME3 interrupt to schedule tasks for a specific time in the future. (Tasks originated by the other program interrupts take place independently of the software executive system.)

Most AGC computations are processed as jobs. Division of a program into discrete jobs is at the discretion of the programmer, who also assigns a priority to each job indicative of its importance. The Executive can manage up to seven jobs (eight in the LM program) simultaneously.

To schedule a job, the Executive places the job's priority and beginning location on a list, assigning the job a set of working storage locations called a core set. In addition, if a job requires a larger working storage, as in the use of interpretive language, a second area, called a VAC area, may be assigned. The Executive is capable of maintaining seven core sets (eight in the LM program) and five VAC areas as each is assigned to a job, and of redesignating them as available when the job is finished.

A job in process must periodically call Executive to scan the list of waiting jobs, thus determining if any scheduled job has a priority higher than itself. If so, the job currently active is suspended and the higher priority job is initiated. To permit suspension of a job and subsequent resumption at a point other than its beginning, the working storage associated with the job is saved when the job is suspended and restored when the job is reinstated. A suspended job is returned to the job list and is not reinstated until it has the highest priority on the list. Eventually, a given job will run to completion, at which time it is removed entirely from consideration. When all jobs on the list have run to completion, a "DUMMYJOB" with zero priority constantly checks to see if new jobs have appeared. (The computer also performs a self-check, as described in Section 2.1.4.)

The relative importance of a job may change for various reasons. When this is the case, Executive changes the priority list and rechecks the list for the job of highest priority. Many times it is desirable to purposely suspend the execution of a job, but not to terminate it completely. Temporary suspension is desirable to await an event such as the input or output of data, or for the availability of a nonreenterable subroutine currently in use. To accomplish temporary suspension, Executive saves the job's interrupted registers and sets its priority to a negative value. Because the interrupted job has a negative priority, DUMMYJOB has priority over it. As a result, the job is, in effect, suspended indefinitely. Eventually, Executive is called to restore the job, usually by the event for which the job is waiting. Executive restores the original priority and again checks the list for the highest priority job.

Waitlist allows any program to schedule a task to occur at a specified time in the future. The TIME3 clock interrupts the job in process at the correct time and initiates the task. (As mentioned before, tasks initiated by the other program interrupts are not controlled by the Executive.)

To schedule a new task, Waitlist requires the starting address of the task and the amount of time which must elapse before execution. Waitlist maintains a list of tasks waiting to run in the order in which they will be performed and a list of time differences between adjacent items on the task list. It determines when the new task will run in relation to others on the list, placing it appropriately in the list.

The TIME3 counter counts the time to the first item on the list. When this time arrives, the TIME3 program interrupt occurs. TIME3 is immediately set to overflow when the time has elapsed for the next task on the list, and all tasks and times move up one position on the list. The computer remains in interrupt mode until the task is completed. It is then free to process other interrupts or return to the original job.

Since TIME3 is a single precision AGC word (15 bits) that is incremented 100 times a second, Waitlist can process tasks up to 162.5 sec in the future. For longer delays, a routine called LONGCALL processes a single task—the repeated calling of Waitlist. LONGCALL can schedule tasks for as long as 745 hours in the future, a time span larger than an entire Apollo mission.

2.1.2.3 Sequence Control

In normal AGC operation, the Executive maintains a constant background of activity, while program interrupts break in for short, time-critical bursts. The execution of a job is subject to numerous interruptions. A counter interrupt may occur after the completion of any instruction. Program interrupts stop the job in process. While the computer is in interrupt mode, any further program interrupts are saved by the hardware and processed one at a time before returning to the job. Under control of the Executive, high-priority jobs also steal time from a job in process. This control system of interrupts and priorities ensures that in times of heavy load, the most critical computations for the mission will be processed first.

Normally, the CPU does not stop during periods of low activity. If no jobs or tasks are being executed, the CPU executes a short loop of instructions (DUMMYJOB) which continually looks for jobs to initiate. Periodically, TIME4 overflows, initiating a task to monitor various GN&C subsystems. If an autopilot is in operation, TIME5 triggers other interrupts for autopilot functions. In addition, periodic counter interrupts will occur as counter input is received and clock counters are updated. More extensive computer activity awaits action by the astronaut, as described in the following section.

2.1.3 Computer Interfaces

To perform its various functions, the AGC must interact with the other spacecraft systems, the astronaut, and the ground. External to the AGC are the various sensors and controls which provide inputs, and the spacecraft systems and displays which receive outputs. Figures 2.1-1 and 2.1-2 illustrate the signal interconnections between the computer and the external hardware for the CM and LM systems, respectively. This report will not, in general, discuss these external equipments, except as they apply to specific AGC programs. (See functional description treated in Part 1, Chapter II, and Part 2, Chapter I of this Apollo Final Report.)

Within the AGC, the actual transmission of data is accomplished through special registers known as counters and channels, as discussed below. Various AGC programs process the input and output data. A mission program such as rendezvous will interrogate selected counters and channels for the specific input data it requires. The program will, in turn, issue commands by means of these interfaces. The operation of the mission programs is discussed in Section 2.2 and in the appendices. In addition to the mission programs, there are also special programs designed to process input/output information for purposes of telemetry and communication with the astronauts. These intêrfaces are discussed in the present section.

2.1.3.1 Counters and Channels

All AGC input/output takes place through counters and channels. Counters are used for the transmission and reception of numeric data; channels are used for the communication of discrete* data.

Channels are solid-state registers in the CPU that do not form part of memory. They cannot be referenced by most machine-language instructions, but are read and in some cases written into by means of special channel instructions. Each channel can consist of up to 15 separate bits or discretes. For input channels, the

*The AGC has 15 input and output channels whose bits are individually distinct (i.e., discrete). Each bit either causes or indicates a change of state, e.g., liftoff, zero optics, SPS-engine on, RCS-jet on, etc.

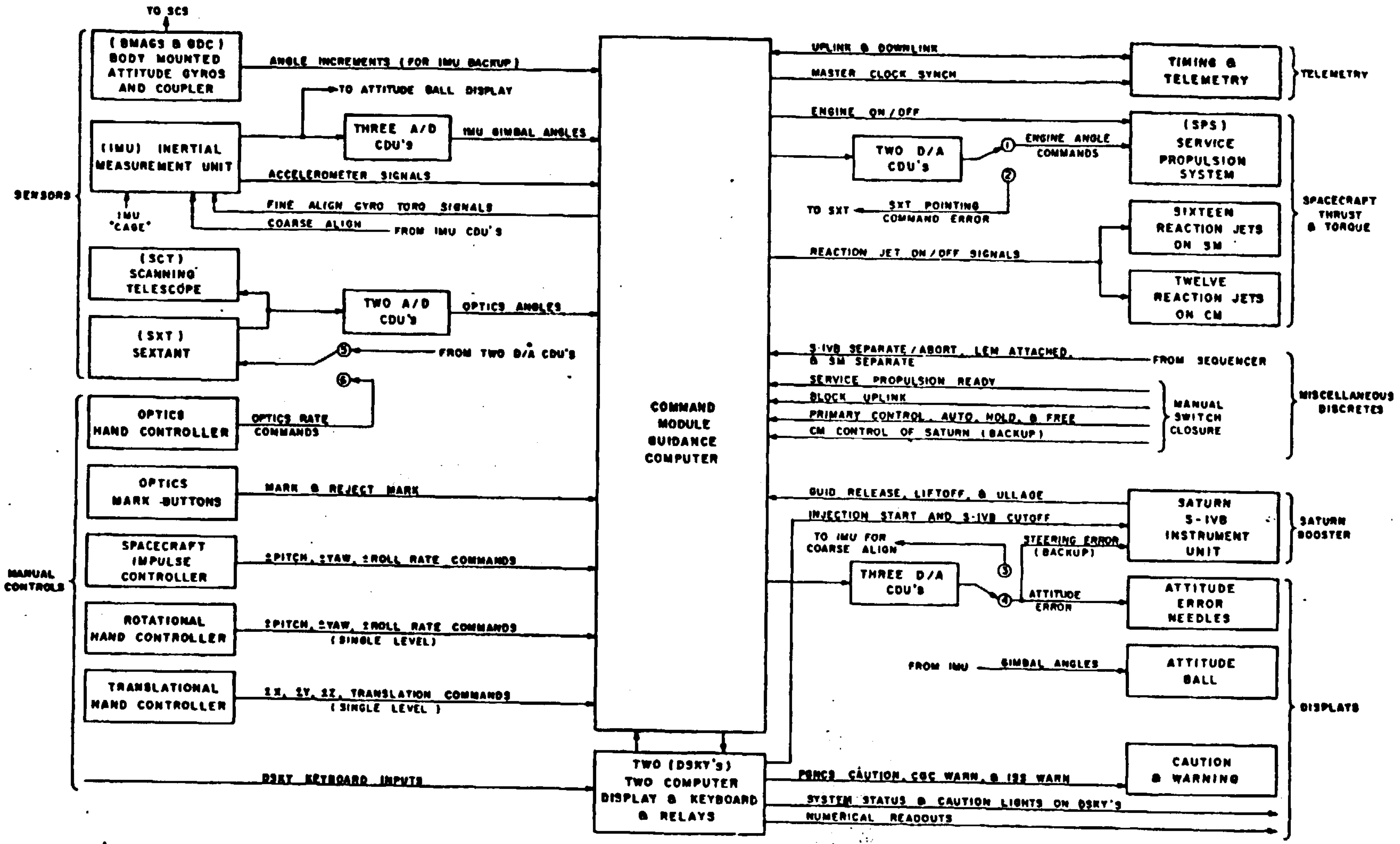

TO SCS
(BMAGS & GDC) BODY MOUNTED ATTITUDE GYROS AND COUPLER
ANGLE INCREMENTS (FOR IMU BACKUP)
TO ATTITUDE BALL DISPLAY
THREE A/D CDU'S
IMU GIMBAL ANGLES
(IMU) INERTIAL MEASUREMENT UNIT
ACCELEROMETER SIGNALS
FINE ALIGN GYRO TORQ SIGNALS
COARSE ALIGN
FROM IMU CDU'S
IMU "CAGE"
SENSORS
(SCT) SCANNING TELESCOPE
TWO A/D CDU'S
OPTICS ANGLES
(SXT) SEXTANT
5
FROM TWO D/A CDU'S
6
OPTICS HAND CONTROLLER
OPTICS RATE COMMANDS
OPTICS MARK BUTTONS
MARK & REJECT MARK
SPACECRAFT IMPULSE CONTROLLER
±PITCH, ±YAW, ±ROLL RATE COMMANDS
ROTATIONAL HAND CONTROLLER
±PITCH, ±YAW, ±ROLL RATE COMMANDS (SINGLE LEVEL)
TRANSLATIONAL HAND CONTROLLER
±X, ±Y, ±Z, TRANSLATION COMMANDS (SINGLE LEVEL)
MANUAL CONTROLS
DSKY KEYBOARD INPUTS
COMMAND MODULE GUIDANCE COMPUTER
TWO (DSKY'S) TWO COMPUTER DISPLAY & KEYBOARD & RELAYS
UPLINK & DOWNLINK
MASTER CLOCK SYNCH
TIMING & TELEMETRY
TELEMETRY
ENGINE ON/OFF
TWO D/A CDU'S
1
ENGINE ANGLE COMMANDS
2
(SPS) SERVICE PROPULSION SYSTEM
TO SXT
SXT POINTING COMMAND ERROR
SIXTEEN REACTION JETS ON SM
REACTION JET ON/OFF SIGNALS
TWELVE REACTION JETS ON CM
SPACECRAFT THRUST & TORQUE
S-IVB SEPARATE/ABORT, LEM ATTACHED, & SM SEPARATE
FROM SEQUENCER
SERVICE PROPULSION READY
BLOCK UPLINK
PRIMARY CONTROL, AUTO, HOLD, & FREE
CM CONTROL OF SATURN (BACKUP)
MANUAL SWITCH CLOSURE
MISCELLANEOUS DISCRETES
GUID RELEASE, LIFTOFF, & ULLAGE
INJECTION START AND S-IVB CUTOFF
SATURN S-IVB INSTRUMENT UNIT
SATURN BOOSTER
TO IMU FOR COARSE ALIGN
STEERING ERROR (BACKUP)
3
THREE D/A CDU'S
4
ATTITUDE ERROR
ATTITUDE ERROR NEEDLES
FROM IMU
GIMBAL ANGLES
ATTITUDE BALL
DISPLAYS
PGNCS CAUTION, CGC WARN, & ISS WARN
CAUTION & WARNING
SYSTEM STATUS & CAUTION LIGHTS ON DSKY'S
NUMERICAL READOUTS

Figure 2.1-1 Guidance, Navigation and Control Interconnections in the Command Module

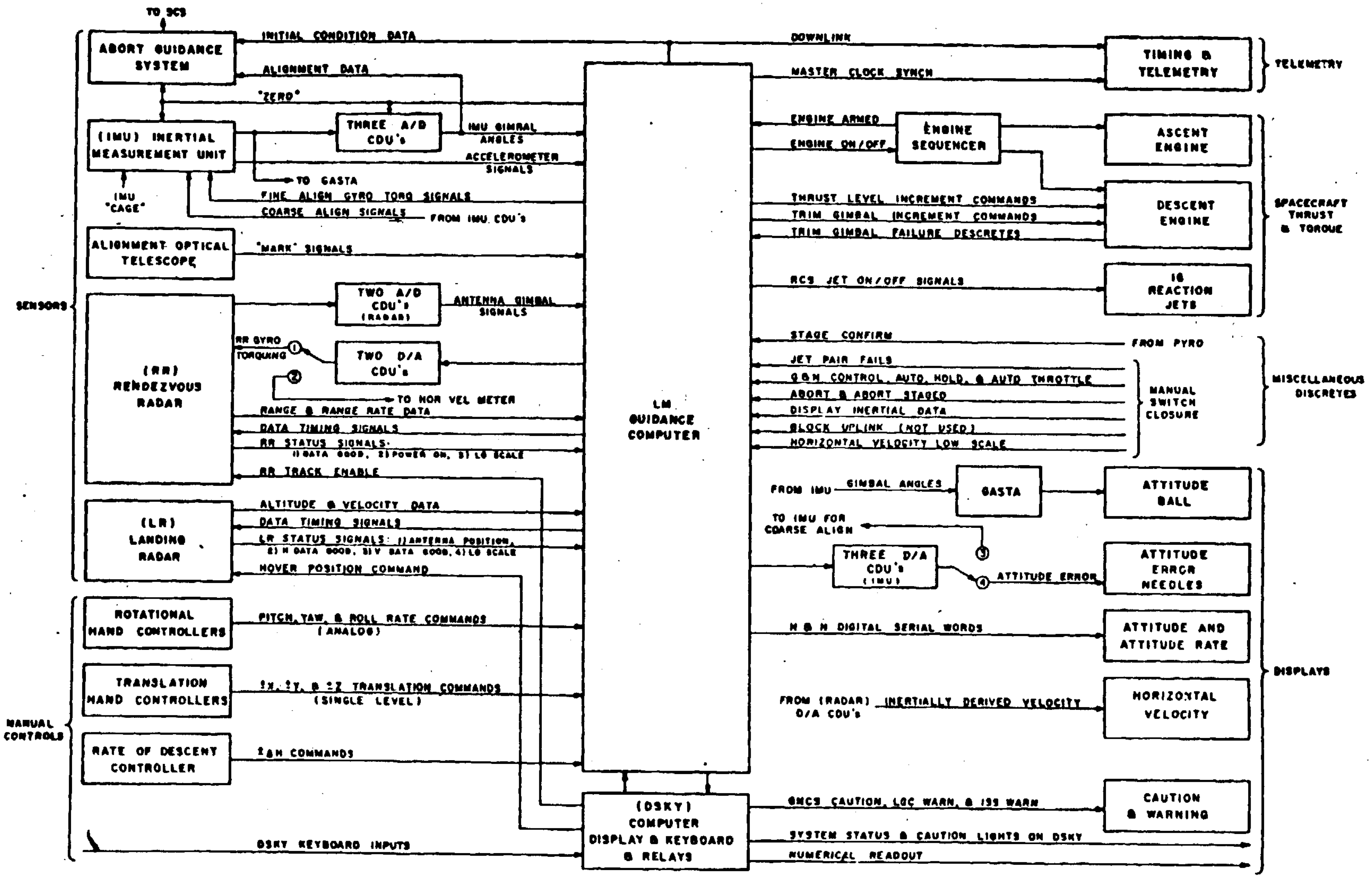


Figure 2.1-2 Guidance, Navigation and Control Interconnections in the Lunar Module

discretes are set by external G&N hardware and may be read by the computer. The input channels inform the computer of the state of the hardware, such as a hand controller out of detent, or the last key depressed on the DSKY. Output channels are written into by the computer to command external hardware functions, such as turning jets on or off, changing the DSKY display, or turning on panel lights. The AGC reads or writes into channels only when instructed to do so—either by the ongoing program or by a program interrupt. For example, pressing a key on the DSKY changes the information in channel 15; it also initiates the KEYRUPT1 program interrupt which causes the computer to read channel 15.

Counters are used for the input and output of numerical information. As described in Section 2.1.2, counters can be changed by programs as if they were ordinary erasable locations, but the counters also respond to counter interrupts which are not under program control.

For input, a typical operation requires that a counter first be set to zero under program control. The counter may then be incremented or decremented, one count at a time, via counter interrupts triggered by an external device. Thus, a counter is able to keep track of the state of the external device. An example of this kind of counter is that used with the Coupling Data Units (CDUs), the interfaces between the Inertial Measurement Unit and the AGC. For each 39.5-arcsec change in a particular gimbal angle, the CDU generates a signal to the AGC which causes a decrement or increment counter interrupt to the appropriate counter.

The output counters function in a similar way. The program sets the counter to an initial value which is later "enabled" via a channel discrete. Following the initialization, all action is automatic and not under program control. A series of counter interrupts decrement the counter toward a value of zero. For each interrupt, a signal of appropriate sign is sent to an external device. When the counter reaches zero, another signal is generated which stops the counting process. Thus, the number of signed pulses sent out is equal to the original contents of the counter. For example, signed pulses torque the gyros or control the optics shaft and trunnion drives.

For telemetry input, counter interrupts shift a pattern of bits into the counter. Selective use of two types of interrupts achieves the desired pattern after the counter has been cleared under program control.

2.1.3.2 Cockpit Displays and Controls

The Apollo Guidance, Navigation and Control System has been designed to utilize the best features of man and machine. Many mission tasks are best left to the computer, such as those that are extremely tedious or that require accurate response too rapid to lie within man's capabilities. However, man's judgment and adaptability, his decision-making capability in reacting to unanticipated situations, and his unique ability to recognize and evaluate patterns are all necessary for mission success. The Apollo displays and controls have therefore been designed to provide the crew with the most flexibility in monitoring and controlling the spacecraft. The astronaut can choose to be directly involved in the procedures, or to allow automatic operation which he can monitor.

Displays available to the crew in both the CM and LM are the attitude ball, attitude-error needles, attitude-rate needles, caution and warning lights, and a DSKY. The LM has additional displays which give the astronaut essential information during the descent to the lunar surface; these are the altitude/altitude-rate, horizontal-velocity, and thrust-level meters and the Landing Point Designator.

Several manual controllers enable the astronaut to become directly involved in spacecraft control. Both the CM and LM have rotational and translational hand controllers. The LM has a rate-of-descent controller. In the CM, additional controllers are used in conjunction with the optics; these are the minimum-impulse and optics hand controllers and the optics mark buttons. In the LM, a DSKY command can convert the rotational hand controller to a minimum impulse controller. All of these controllers make available to the astronaut a large repertoire of manual maneuvers.

The basic man/computer interface device is the DSKY (shown in Fig. 2.1-3). Through the DSKY the astronaut can initiate, monitor, or change programs being processed by the computer. He can request the display of specific data or enter new data. Communication with the DSKY is two-way; just as the astronaut can exercise command via the DSKY, the computer can request the astronaut to monitor, approve, or enter data when necessary. There are two DSKYs available in the CM and one in the LM. Each DSKY has a keyboard, several electroluminescent displays, and activity and alarm lights. The activity lights are for the computer and the

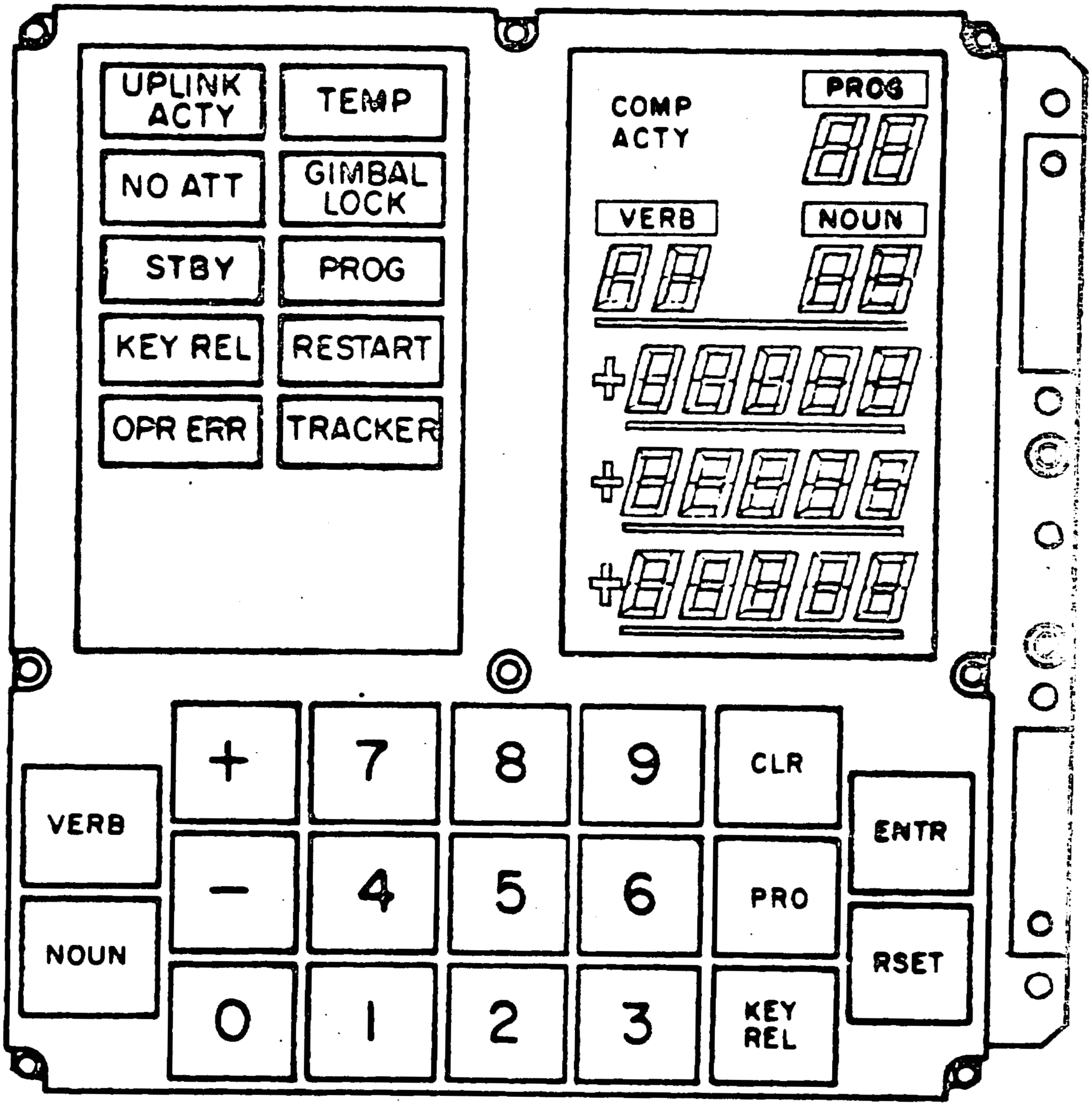


Figure 2.1-3 Display and Keyboard

telemetry uplink, and the alarm lights are for the computer and inertial subsystems. These aid the astronaut in monitoring the status of the G&N system. The alarm lights indicate equipment-failure and program alarms. There are two levels of program alarms. The more serious type of alarm either terminates all but the most necessary program activities or terminates all current program activities and requests astronaut action. The latter is accomplished by a preemptive flashing display of an error code indicating the cause of the alarm. The other type of program alarm is also indicated by the program-alarm light, but in this case the program in process continues without change. Should the astronaut wish to interrogate the cause of this alarm, he can key in a request to the computer to display the error code. The DSKY keyboard and displays are discussed in the next section.

2.1.3.3 PINBALL and DSKY Displays

The AGC program which responds to DSKY buttons and requests illumination of the DSKY lights is called PINBALL GAME BUTTONS AND LIGHTS—or PINBALL, for short. PINBALL is under Executive control and enables communication between the computer and the astronaut. As mentioned in the previous section, exchanges can be initiated by operator action or by an internal computer program. Four modes of operation are associated with PINBALL—internal data display, external data loading, systems-test usage, and initiation of large-scale mission phases. Internal data can be displayed once for verification (e.g., the ascent-injection parameters for lunar ascent) or periodically updated and displayed for monitoring (e.g., time-to-go to main-engine ignition). External data are displayed in the appropriate display-panel register as they are keyed into the DSKY. The data for the loading (external) and displaying (internal) modes can be presented in octal or decimal format; if internal data are presented in decimal format, the program supplies the appropriate scale factors for the display. PINBALL can also initiate a class of routines used for systems-test functions which might require operator interaction to determine whether to stop or continue the routine. The final mode of PINBALL is initiation of large-scale mission phases by operator action, i.e., by changing the mission program via the DSKY. (Fig. 2.1-4 lists the AGC programs for the CM and LM available during a lunar mission.)

The DSKY keyboard contains the following notations: VERB, NOUN, +, -, the numerical characters 0 through 9, CLR (clear), ENTR (enter), RSET (error reset),

Command Module AGC Programs

00	CMC[*] Idling
01	Prelaunch Initialization
02	Gyro Compassing
03	Verify Gyro Compassing
06	CMC Power Down
07	IMU Ground Test
11	Earth Orbit Insertion (EOI) Monitor
17	Transfer Phase Initiation (TPI) Search
20	Rendezvous Navigation
21	Ground Track Determination
22	Orbital Navigation
23	Cislunar Midcourse Navigation
27	CMC Update
30	External ΔV
31	Lambert Aimpoint Maneuver
32	Coelliptic Sequence Initiation (CSI)
33	Constant Delta Height (CDH)
34	Transfer Phase Initiation (TPI)
35	Transfer Phase Midcourse (TPM)
37	Return to Earth (RTE)
38	Stable Orbit Rendezvous (SOR)
39	Stable Orbit Midcourse (SOM)
40	SPS
41	RCS
47	Thrust Monitor
51	IMU Orientation Determination
52	IMU Realign
53	Backup IMU Orientation Determination
54	Backup IMU Realign
61	Maneuver to CM/SM Separatio titude
62	CM/SM Separation & Preentry ıeuver
63	Entry-Initialization
64	Entry-Post 0.05 g
65	Entry-Up Control
66	Entry-Ballistic
67	Entry-Final Phase
72	LM Coelliptic Sequence Initiation (CSI)
73	LM Constant Delta Height (CDH)
74	LM TPI Targeting
75	LM TPM Targeting
76	Target ΔV
77	LM TPI Search
78	LM SOR Targeting
79	LM SOM Targeting

Lunar Module AGC Programs

00	LGC[**] Idling
06	GNCS Power Down
12	Powered Ascent Guidance
20	Rendezvous Navigation
21	Ground Track Determination
22	Lunar Surface Navigation
25	Preferred Tracking Attitude
27	LGC Update
30	External ΔV
31	Lambert Aimpoint Maneuver
32	Coelliptic Sequence Initiation (CSI)
33	Constant Delta Height (CDH)
34	Transfer Phase Initiation (TPI)
35	Transfer Phase Midcourse (TPM)
38	Stable Orbit Rendezvous (SOR)
39	Stable Orbit Midcourse (SOM)
40	DPS
41	RCS
42	APS
47	Thrust Monitor
51	IMU Orientation Determination
52	IMU Realign
57	Lunar Surface Align
63	Braking Phase
64	Approach Phase
65	Landing Phase (Auto)
66	Landing Phase (ROD)
67	Landing Phase (Manual)
68	Landing Confirmation
70	DPS Abort
71	APS Abort
72	CSM CSI Targeting
73	CSM CDH Targeting
74	CSM TPI Targeting
75	CSM TPM Targeting
76	Target ΔV
78	CSM SOR Targeting
79	CSM SOM Targeting

[*] CMC is Command Module Computer (CM AGC)

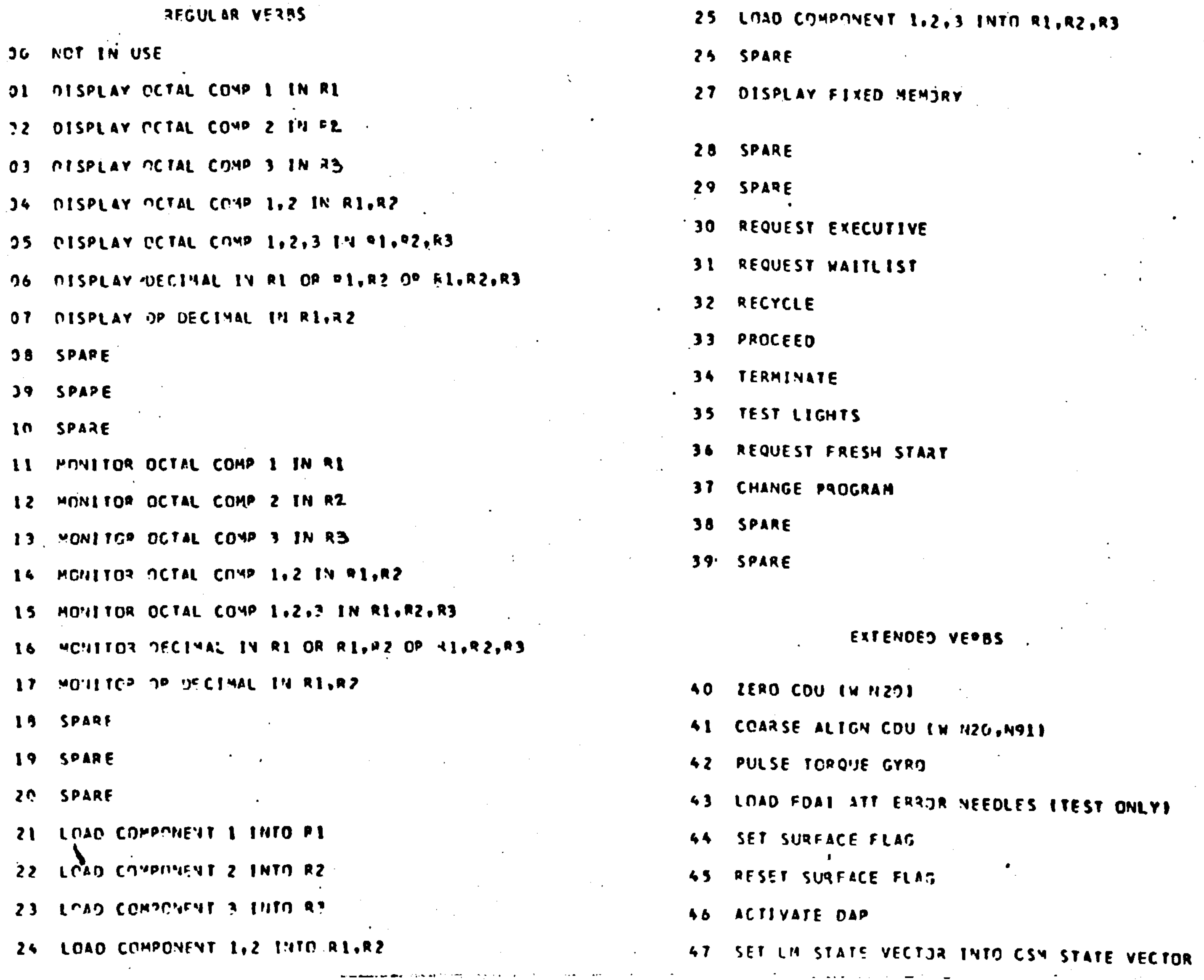
[**] LGC is Lunar Guidance Computer (LM AGC)

Figure 2.1-4 Programs for a Lunar-Landing Mission

PRO (proceed), and KEY REL (key release). Each of these notations is internally represented by a 5-bit binary code which is transmitted and recognized by the computer. When the operator depresses any one of these buttons on the keyboard, an interrupt program called KEYRUPT enters a request to the Executive for another program that decodes and stores the key code in an input register of the AGC.

The numeric sections of the DSKY panel form three data-display registers, R1, R2 and R3, which can contain up to five numerals each, and three control display registers, VERB, NOUN, and PROG (program), of two numerals each. Each of the three data display registers has a sign section which displays a plus sign, a minus sign or nothing at all (blank). The PROG register indicates the mission program currently operating; the VERB and NOUN registers indicate the display and load activity initiated by the operator or by the computer. All information necessary to operate the display panel on the DSKY is transmitted from the computer through an output register which activates two display characters at a time. The basic language used for communication between the operator and PINBALL is a pair of two-character numbers that represents a verb/noun combination. The verb code indicates the operation to be performed, while the noun code indicates the operand to which the operation (verb) applies. Typical of the verb codes used are those for displaying and loading data. Noun codes call up groups of erasable registers within computer memory. Figures 2.1-5 and 2.1-6 give a list of the verbs and nouns available in the AGC for the CSM program COLOSSUS. (The LM program, LUMINARY, has a similar list.)

In addition to the numeric buttons and verb/noun control buttons, PINBALL responds to the other control buttons found on the DSKY. The RSET button usually turns off the alarm lights on the panel. Should any of these alarm lights remain on after the RSET button is depressed, the condition causing the alarm persists. The ENTR button has two functions: it causes the AGC to execute the verb/noun combination appearing in the VERB and NOUN registers or to accept a newly-entered data word. The CLR button is used to blank R1, R2, or R3 during a data-loading sequence, thus allowing reloading of a data word. The KEY REL button allows internal programs to use the DSKY if the operator has not previously released the DSKY for such use. The KEY REL light is turned on when an internal program attempts to use the DSKY but finds that the astronaut has not released it for internal use; depressing the KEY REL button performs this release. Thus, the operator



REGULAR VERBS

00	NOT IN USE
01	DISPLAY OCTAL COMP 1 IN R1
02	DISPLAY OCTAL COMP 2 IN R2
03	DISPLAY OCTAL COMP 3 IN R3
04	DISPLAY OCTAL COMP 1,2 IN R1,R2
05	DISPLAY OCTAL COMP 1,2,3 IN R1,R2,R3
06	DISPLAY DECIMAL IN R1 OR R1,R2 OR R1,R2,R3
07	DISPLAY DP DECIMAL IN R1,R2
08	SPARE
09	SPARE
10	SPARE
11	MONITOR OCTAL COMP 1 IN R1
12	MONITOR OCTAL COMP 2 IN R2
13	MONITOR OCTAL COMP 3 IN R3
14	MONITOR OCTAL COMP 1,2 IN R1,R2
15	MONITOR OCTAL COMP 1,2,3 IN R1,R2,R3
16	MONITOR DECIMAL IN R1 OR R1,R2 OR R1,R2,R3
17	MONITOR DP DECIMAL IN R1,R2
18	SPARE
19	SPARE
20	SPARE
21	LOAD COMPONENT 1 INTO R1
22	LOAD COMPONENT 2 INTO R2
23	LOAD COMPONENT 3 INTO R3
24	LOAD COMPONENT 1,2 INTO R1,R2
25	LOAD COMPONENT 1,2,3 INTO R1,R2,R3
26	SPARE
27	DISPLAY FIXED MEMORY
28	SPARE
29	SPARE
30	REQUEST EXECUTIVE
31	REQUEST WAITLIST
32	RECYCLE
33	PROCEED
34	TERMINATE
35	TEST LIGHTS
36	REQUEST FRESH START
37	CHANGE PROGRAM
38	SPARE
39	SPARE

EXTENDED VERBS

40	ZERO CDU (W N20)
41	COARSE ALIGN CDU (W N20,N91)
42	PULSE TORQUE GYRO
43	LOAD FDAI ATT ERROR NEEDLES (TEST ONLY)
44	SET SURFACE FLAG
45	RESET SURFACE FLAG
46	ACTIVATE DAP
47	SET LM STATE VECTOR INTO CSM STATE VECTOR

Figure 2.1-5 Verbs Used in Program COLOSSUS

```
48  LOAD DAP DATA (R03)
49  START CREW DEFINED MANEUVER (R62)
50  PLEASE PERFORM
51  PLEASE MARK
52  MARKED ON OFFSET LANDING SITE
53  PLEASE MARK ALTERNATE LOS
54  START REND BACK UP SIGHTING MARK (R23)
55  INCREMENT CMC TIME (DECIMAL)
56  TERMINATE TRACKING
57  START REND SIGHTING MARK (R21)
58  RESET STICK FLAG AND SET V50N18 FLAG
59  PLEASE MARK (OPTICS CALIBRATION)
60  SET ATTITUDE ERROR REFERENCE TO PRESENT ATTITUDE.
61  SELECT MODE 1 (DISPLAY DAP ATTITUDE ERROR)
62  SELECT MODE 2 (DISPLAY TOTAL ATTITUDE ERROR (N22-N20))
63  SELECT MODE 3 (DISPLAY TOTAL ASTRONAUT ATTITUDE ERROR (N17-N20))
64  START S-BAND ANT CALC (R05)
65  START OPTICAL VERIFICATION OF PRELAUNCH ALIGNMENT (P03)
66  SET CSM STATE VECTOR INTO LM STATE VECTOR
67  START W-MATRIX RMS ERROR DISPLAY
68  CSM STROKE TEST ON
69  RESTART
70  UPDATE LIFTOFF TIME (P27)
71  UNIVERSAL UPDATE-BLOCK ADR (P27)
72  UNIVERSAL UPDATE-SINGLE ADR (P27)
73  UPDATE CMC TIME (OCTAL) (P27)

74  INITIALIZE ERASABLE DUMP VIA DOWNLINK
75  BACKUP LIFTOFF
76  SET PREFERRED ATTITUDE FLAG
77  RESET PREFERRED ATTITUDE FLAG
78  UPDATE PRELAUNCH AZIMUTH
79  START BARBECUE MODE ROUTINE (R64)
80  UPDATE LM STATE VECTOR
81  UPDATE CSM STATE VECTOR
82  REQUEST ORBIT PARAM DISPLAY (R30)
83  REQUEST REND PARAM DISPLAY #1 (R31)
84  SPARE
85  REQUEST REND PARAM DISPLAY #2 (R34)
86  REJECT REND BACK UP SIGHTING MARK
87  SET VHF RANGE FLAG
88  RESET VHF RANGE FLAG
89  START REND FINAL ATTITUDE MANEUVER (R63)
90  REQUEST REND OUT OF PLANE DISPLAY (R36)
91  BANKSUM
92  START IMU PERFORMANCE TEST (P07)
93  ENABLE W MATRIX INITIALIZATION
94  ENABLE CISLUNAR TRACKING RECYCLE
95  SPARE
96  TERMINATE INTEGRATION AND GO TO P00
97  THRUST FAIL DISPLAY
98  SPARE
99  ENABLE ENGINE IGNITION
```

Figure 2.1-5 (cont.)   Verbs Used in Program COLOSSUS

00	NOT IN USE	
01	SPECIFY ADDRESS (FRAC)	.XXXXX FRAC .XXXXX FRAC .XXXXX FRAC
02	SPECIFY ADDRESS (WHOLE)	XXXXX. INTEG XXXXX. INTEG XXXXX. INTEG
03	SPECIFY ADDRESS (DEGREE)	XXX.XX DEG XXX.XX DEG XXX.XX DEG
04	SPARE	
05	ANGULAR ERROR/DIFFERENCE	XXX.XX DEG
06	OPTION CODE	OCT OCT
07	CHANNEL/FLAGWORD/ERASABLE OPERATOR	OCT OCT OCT
08	ALARM DATA	OCT OCT OCT
09	ALARM CODES	OCT OCT OCT
10	CHANNEL TO BE SPECIFIED	OCT
11	TIG OF CSI	00XXX. HRS 000XX. MIN 0XX.XX SEC
12	OPTION CODE	OCT OCT
13	TIG OF CDH	00XXX. HRS 000XX. MIN 0XX.XX SEC
14	SPARE	

15	INCREMENT ADDRESS	OCT
16	TIME OF EVENT (USED BY EXT VERB ONLY)	00XXX. HRS 000XX. MIN 0XX.XX SEC
17	ASTRONAUT TOTAL ATTITUDE (USED IN MODE 3 NEEDLES (V62))	XXX.XX DEG XXX.XX DEG XXX.XX DEG
18	BALL ANGLES AUTO MANEUVER	R XXX.XX DEG P XXX.XX DEG Y XXX.XX DEG
19	SPARE	
20	PRESENT ICDU ANGLES	R XXX.XX DEG P XXX.XX DEG Y XXX.XX DEG
21	PIPAS	X XXXXX. PULSES Y XXXXX. PULSES Z XXXXX. PULSES
22	NEW ICDU ANGLES	R XXX.XX DEG P XXX.XX DEG Y XXX.XX DEG
23	SPARE	
24	DELTA TIME FOR CMC CLOCK	00XXX. HRS 000XX. MIN 0XX.XX SEC
25	CHECKLIST (USED WITH V50)	XXXXX.
26	PRIO/DELAY,ADRES,BBCON	OCT OCT OCT
27	SELF TEST ON/OFF SWITCH	XXXXX.
28	SPARE	
29	XSM LAUNCH AZ	XXX.XX DEG
30	TARGET CODE (GYROCOMPASSING VERIFICATION)	XXXXX. XXXXX. XXXXX.

Figure 2.1-6 Nouns Used in Program COLOSSUS

31	*SPARE*	
32	TIME FROM PERIGEE	XXX. HRS
		000XX. MIN
		0XX.XX SEC
33	GETI	00XXX. HRS
		000XX. MIN
		0XX.XX SEC
34	TIME OF EVENT	00XXX. HRS
		000XX. MIN
		0XX.XX SEC
35	TIME FROM EVENT	00XXX. HRS
		000XX. MIN
		0XX.XX SEC
36	TIME OF CMC CLOCK	00XXX. HRS
		000XX. MIN
		0XX.XX SEC
37	GETI(TPI)	00XXX. HRS
		000XX. MIN
		0XX.XX SEC
38	TIME OF STATE VECTOR	00XXX. HRS
		000XX. MIN
		0XX.XX SEC
39	DELTA TIME FOR TRANSFER	00XXX. HRS
		000XX. MIN
		0XX.XX SEC
40	TFI/TFC	XXBXX M-S
	VG	XXXX.X FPS
	DELTA V (ACCUMULATED)	XXXX.X FPS
41	TARGET AZIMUTH	XXX.XX DEG
	TARGET ELEVATION	XX.XXX DEG
	TARGET IDENTIFIER (PASTE FROM N30)	0000X
42	APO ALT	XXXX.X NM
	PER ALT	XXXX.X NM
	DELTA V (REQUIRED)	XXXX.X FPS
43	LATITUDE (+ NORTH)	XXX.XX DEG
	LONGITUDE (+ EAST)	XXX.XX DEG
	ALTITUDE	XXXX.X NM
44	APO ALT	XXXX.X NM
	PER ALT	XXXX.X NM
	TFF	XXBXX M-S
45	MARKS (VHF-OPTICS)	XXBXX
	TFI (NEXT BURN)	XXBXX M-S
	MGA	XXX.XX DEG
46	DAP CONFIG	OCT
		OCT
47	THIS VEHICLE WEIGHT	XXXXX. LBS
	OTHER VEHICLE WEIGHT	XXXXX. LBS
48	GIMBAL PITCH TRIM	XXX.XX DEG
	GIMBAL YAW TRIM	XXX.XX DEG
49	DELTA R	XXXX.X NM
	DELTA V	XXXX.X FPS
	SOURCE CODE	0000X.
50	SPLERROR	XXXX.X NM
	PERIGEE	XXXX.X NM
	TFF	XXBXX M-S
51	RHO	XXX.XX DEG
	GAMMA	XXX.XX DEG
52	CENTRAL ANGLE OF ACTIVE VEHICLE	XXX.XX DEG
53	RANGE	XXX.XX NM
	RANGE RATE	XXXX.X FPS
	PHI	XXX.XX DEG
54	RANGE	XXX.XX NM
	RANGE RATE	XXXX.X FPS
	THETA	XXX.XX DEG
55	PER CODE	XXXXX.
	ELEVATION ANGLE (E)	XXX.XX DEG
	CENTRAL ANGLE OF PASSIVE VEHICLE	XXX.XX DEG
56	REENTRY ANGLE	XXX.XX DEG
	DELTA V	XXXXX. FPS
57	DELTA R (SOI) (+ PASS VEH LEADS)	XXXX.X NM

Figure 2.1-6 (cont.) Nouns Used in Program COLOSSUS

58	PER ALT (POST TPI OR SOR)	XXXX.X NM
	DELTA V (TPI OR SOR)	XXXX.X FPS
	DELTA V (TPF OR SOR FINAL)	XXXX.X FPS
59	DELTA V LOS 1	XXXX.X FPS
	DELTA V LOS 2	XXXX.X FPS
	DELTA V LOS 3	XXXX.X FPS
60	G MAX	XXX.XX G
	V PRED	XXXXX. FPS
	GAMMA EI (+ UP)	XXX.XX DEG
61	IMPACT LATITUDE	XXX.XX DEG
	IMPACT LONGITUDE	XXX.XX DEG
	HEADS UP/DOWN (+ UP)	+000X.
62	VI ,INERTIAL VEL MAG	XXXXX. FPS
	HDOT ,ALT RATE	XXXXX. FPS
	H ,ALT ABOVE PAD RADIUS OF LANDING SITE	XXXX.X NM
63	RTGO ,RNG FROM E.I. TO SPLASH	XXXX.X NM
	VIO ,PREDICTED INERT VEL	XXXXX. FPS
	TFE ,TIME FROM FROM E.I.	XXBXX M-S
64	G, DRAG ACCELERATION	XXX.XX G
	VI ,INERTIAL VELOCITY	XXXXX. FPS
	R TO GO (+ OVSHT)	XXXX.X NM
65	SAMPLED CMC TIME (FETCHED IN INTERRUPT)	00XXX. HRS
		000XX. MIN
		0XX.XX SEC
66	BETA, CMD BANK ANGLE	XXX.XX DEG
	CROSS RANGE ERROR (+ TGT RT)	XXXX.X NM
	DOWN RANGE ERROR (+ OVSHT)	XXXX.X NM
67	R TO GO (+ OVSHT)	XXXX.X NM
	LAT ,PRESENT POSITION (+ NORTH)	XXX.XX DEG
	LONG ,PRESENT POSITION (+ EAST)	XXX.XX DEG
68	BETA, CMD BANK ANGLE	XXX.XX DEG
	VI ,INERTIAL VELOCITY	XXXXX. FPS
	HDOT ,ALT RATE	XXXXX. FPS
69	BETA	XXX.XX DEG
	DL	XXX.XX G
	VL	XXXXX. FPS
70	CELESTIAL BODY CODE (BEFORE MARK)	OCT
	LANDMARK DATA	OCT
	HORIZON DATA	OCT
71	CELESTIAL BODY CODE (AFTER MARK)	OCT
	LANDMARK DATA	OCT
	HORIZON DATA	OCT
72	DELT ANG (+ ACT VEH LEADS)	XXX.XX DEG
	DELT ALT (+ PASS VEH ABOVE)	XXXX.X NM
	SEARCH OPTION	XXXXX.
73	ALTITUDE	XXXXXB. NM
	VELOCITY	XXXXX. FPS
	FLIGHT PATH ANGLE	XXX.XX DEG
74	BETA, CMD BANK ANGLE	XXX.XX DEG
	VI , INERTIAL VELOCITY	XXXXX. FPS
	G , DRAG ACCELERATION	XXX.XX G
75	DELTA ALTITUDE CDH	XXXX.X NM
	DELTA TIME (CDH-CSI OR TPI-CDH)	XXBXX MIN/SEC
	DELTA TIME (TPI-CDH OR TPI-NOMTPI)	XXBXX MIN/SEC
76	SPARE	
77	SPARE	
78	SPARE	
79	RATE (+ INCREASING CDU)	X.XXXX DEG/SEC
	DEADBAND	XXX.XX DEG
	AXIS CODE	XXXXX.
80	TFI/TFC	XXBXX M-S
	VG	XXXXX. FPS
	DELTA V (ACCUMULATED)	XXXXX. FPS
81	DELTA VX (LV)	XXXX.X FPS
	DELTA VY (LV)	XXXX.X FPS
	DELTA VZ (LV)	XXXX.X FPS
82	DELTA VX (LV)	XXXX.X FPS
	DELTA VY (LV)	XXXX.X FPS
	DELTA VZ (LV)	XXXX.X FPS
83	DELTA VX(CONT)	XXXX.X FPS
	DELTA VY(CONT)	XXXX.X FPS
	DELTA VZ(CONT)	XXXX.X FPS
84	DELTA VX(O VEH)	XXXX.X FPS
	DELTA VY(O' VEH)	XXXX.X FPS
	DELTA VZ(O.VEH)	XXXX.X FPS

Figure 2.1-6 (cont.) Nouns Used in Program COLOSSUS

```
85  VGX (CONT)                                   XXXX.X FPS
    VGY (CONT)                                   XXXX.X FPS
    VGZ (CONT)                                   XXXX.X FPS

86  DELTA VX (LV)                                XXXXX. FPS
    DELTA VY (LV)                                XXXXX. FPS
    DELTA VZ (LV)                                XXXXX. FPS

87  MARK DATA: OPTICS SHAFT ANGLE                XXX.XX DEG
               OPTICS TRUNNION ANGLE             XX.XXX DEG

88  PLANET UNIT POSITION VECTOR X                .XXXXX
                                Y                .XXXXX
                                Z                .XXXXX

89  LANDMARK LATITUDE (+ NORTH)                  XX.XXX DEG
    LANDMARK LONGITUDE/2 (+ EAST)                XX.XXX DEG
    LANDMARK ALTITUDE                            XXX.XX NM

90  REND OUT OF PLANE PARAMETERS Y               XXX.XX NM
                                 Y DOT           XXXX.X FPS
                                 PSI             XXX.XX DEG

91  PRESENT OCDU ANGLES - SHAFT                  XXX.XX DEG
                        - TRUN                   XX.XXX DEG

92  NEW OCDU ANGLES-  SHAFT                      XXX.XX DEG
                      TRUN                       XX.XXX DEG

93  DELTA GYRO ANGLES                          X XX.XXX DEG
                                               Y XX.XXX DEG
                                               Z XX.XXX DEG

94  ALTERNATE LOS - SHAFT                        XXX.XX DEG
                    TRUN                         XX.XXX DEG

95  PREF ATT FDAI ANGLES                       R XXX.XX DEG
                                               P XXX.XX DEG
                                               Y XXX.XX DEG

96  +X AXIS ATT FDAI ANGLES                    R XXX.XX DEG
                                               P XXX.XX DEG
                                               Y XXX.XX DEG

97  SYSTEM TEST INPUTS                           XXXXX.
                                                 XXXXX.
                                                 XXXXX.

98  SYSTEM TEST RESULTS                          XXXXX.
                AND INPUTS                       .XXXXX
                                                 XXXXX.

99  RMS VALUE OF POSITION ERROR                  XXXXX. FT
    RMS VALUE OF VELOCITY ERROR                  XXXX.X FPS
    OPTION CODE                                  XXXXX.
```

Figure 2.1-6 (cont.)   Nouns Used in Program COLOSSUS

has control over the displays he wishes to observe, without being interrupted by an internal request. As will be shown in a discussion which follows on multi-level displays, the KEY REL button can also be used to reestablish displays which have been temporarily suspended.

While the astronaut communicates with the computer by entering information in the DSKY, the computer communicates with the astronaut by a flashing or nonflashing verb/noun display. The loading of data registers provides an example of two-way communication. To load three registers of data, the astronaut selects VERB 25 NOUN XX ENTR, where NOUN XX describes the data involved. He then depresses the ENTR button and the computer responds by flashing VERB 21, telling him to load register R1, which has been blanked. After the astronaut keys in the initial data, he keys ENTR. The computer responds with a flashing VERB 22, indicating that it is ready to accept data in the second register. The process is then repeated for the third register. Since PINBALL is able to distinguish between two modes of the ENTR button (execute verb/noun or enter data), data are not processed until the final component is loaded and the ENTR button is depressed. At this time, the data entered are scaled for each component and stored in the proper location in memory.

When a sign button is depressed before data are entered into each register, numeric information is treated as decimal; otherwise, PINBALL considers the data to be octal. If the operator depresses the 8 or 9 button on the DSKY while loading octal data, the OPR ERR (operator error) light is illuminated, which he can turn off by depressing the RSET button.

PINBALL was first developed to exercise systems-test and operations programs in an early version of the AGC. At that time only one level of priority was provided. Consequently, two internal jobs requiring displays could not run simultaneously. (This was satisfactory then and even for later unmanned flights during which the Boost Monitor Display—a constantly updated sequence of trajectory parameters —was continuously displayed on the DSKY.) But procedures like rendezvous-radar navigation marktaking could not run in the background behind a targeting computation and communicate updated data through the normal display activity in the foreground. With the advent of manned flights, it became clear that the computer would have to communicate with the astronaut on several levels; consequently, development of cause the next lower-level display to reappear. This feature gives the astronaut the flexibility of using five levels of displays at a time.

2.1.3.4 Uplink and Downlink

Uplink is the digital telemetry system which enables ground control to load data or issue instructions to the AGC in the same manner employed by the astronaut using the DSKY keyboard. All information received by the AGC via uplink is in the form of keyboard characters. Each character is assigned an identifying code number called its character code. The AGC picks up the transmitted codes (these codes are the same as key codes) and enters a request to the Executive for the program which decodes and accepts them. The PINBALL program which decodes and accepts the transmitted code makes no distinction between inputs from the keyboard or from uplink, and any ground-command sequence normally transmitted via uplink may be duplicated by the astronaut using the keyboard.

The astronaut can choose to reject uplink from ground control by setting a toggle switch on the cockpit control panel to the blocked position.

A Universal Update Program exists in the AGC which facilitates updating the erasable memory and can be called by a number of extended verbs. To protect against the ingestion of erroneous information, the Update Program temporarily stores all new inputs in a buffer and transmits its contents back to ground control via downlink (see below) for verification. Furthermore, storage of state-vector updates (position and velocity) with their associated sphere-of-influence (earth or lunar) are delayed until current state-vector integration is finished.

The Update Program accepts four types of erasable-memory updates:

1. Contiguous Block Update provides ground-control capability to update up to 18 consecutive erasable-memory registers in the same erasable-memory bank.
2. Scatter Update provides ground-control capability to update from 1 to 9 nonconsecutive erasable-memory registers in the same or different erasable banks.
3. Octal-Clock Increment provides ground-control capability to increment or decrement the AGC clock with a double-precision octal-time value.

has control over the displays he wishes to observe, without being interrupted by an internal request. As will be shown in a discussion which follows on multi-level displays, the KEY REL button can also be used to reestablish displays which have been temporarily suspended.

While the astronaut communicates with the computer by entering information in the DSKY, the computer communicates with the astronaut by a flashing or nonflashing verb/noun display. The loading of data registers provides an example of two-way communication. To load three registers of data, the astronaut selects VERB 25 NOUN XX ENTR, where NOUN XX describes the data involved. He then depresses the ENTR button and the computer responds by flashing VERB 21, telling him to load register R1, which has been blanked. After the astronaut keys in the initial data, he keys ENTR. The computer responds with a flashing VERB 22, indicating that it is ready to accept data in the second register. The process is then repeated for the third register. Since PINBALL is able to distinguish between two modes of the ENTR button (execute verb/noun or enter data), data are not processed until the final component is loaded and the ENTR button is depressed. At this time, the data entered are scaled for each component and stored in the proper location in memory.

When a sign button is depressed before data are entered into each register, numeric information is treated as decimal; otherwise, PINBALL considers the data to be octal. If the operator depresses the 8 or 9 button on the DSKY while loading octal data, the OPR ERR (operator error) light is illuminated, which he can turn off by depressing the RSET button.

PINBALL was first developed to exercise systems-test and operations programs in an early version of the AGC. At that time only one level of priority was provided. Consequently, two internal jobs requiring displays could not run simultaneously. (This was satisfactory then and even for later unmanned flights during which the Boost Monitor Display—a constantly updated sequence of trajectory parameters —was continuously displayed on the DSKY.) But procedures like rendezvous-radar navigation marktaking could not run in the background behind a targeting computation and communicate updated data through the normal display activity in the foreground. With the advent of manned flights, it became clear that the computer would have to communicate with the astronaut on several levels; consequently, development of display-interface software and a hierarchy of priority interrupts was begun. Boost Monitor Display programs in SUNSPOT were the initial components of the complete G&N astronaut/AGC interface software that was further developed in SUNDISK and ultimately refined for COLOSSUS and LUMINARY.

The initial display-interface routine, GOFLASH, was created to save coding for the four or five calls to PINBALL by the Boost Monitor programs. The subroutine approach saved 12 instructions of the 18 otherwise required each time the AGC initiated an information transfer through PINBALL to the DSKY. In a recent COLOSSUS program, there are 45 calls to GOFLASH, which accomplishes a net saving of 540 instructions.

A second level of displays which was added carried a higher priority than normal program displays. These so-called Extended-Verb displays permitted an information request to be keyed in—even though another normal-priority program might be in progress—and to attract the crew's attention via a flashing display, effectively preempting the normal program's DSKY activity. An Extended Verb usually takes the form of an information request which differs from a regular verb in that it cannot be satisfied by simply displaying already available information stored in an erasable-memory location. An Extended Verb requires some data manipulation and ordinarily involves one or more subroutine calls. While the Extended Verb is running, the normal display is held in abeyance. Since sufficient information has to be saved to restore an interrupted display after the interrupt, display points became natural restart points. And because displays are usually natural breakpoints in an extended computation, they provide excellent demarcation points for program phase changes. A special restart mechanism therefore was created to permit "restarts" to pick up at the most recent display. A more comprehensive description of restarts follows in Section 2.1.4.

At about the time the need for Extended-Verb displays was recognized, a similar requirement was recognized for mark displays. During rendezvous, the astronaut is very busy with three four-part operational cycles (navigation, targeting, maneuver, and burn) in succession to be accomplished during brief spans of time. It therefore became virtually mandatory that the Range Radar (LM) and VHF (CM) navigation marktaking be performed automatically without astronaut supervision, but with provision for astronaut intervention if anomalous mark data were obtained. The same priority-interrupt technique implemented in the Extended Verb feature was also implemented to permit navigation marks to be taken while a targeting routine was in progress, and—when they satisfied certain threshold-acceptance criteria—to be incorporated automatically. Only marks that violate accept/reject criteria need be presented for the astronaut's consideration explicitly via the display-interrupt software interface. Since Extended Verbs and marking-program displays shared the same priority level, a restriction was necessarily imposed that no Extended Verb using displays could be imposed during marktaking.

A second higher level of priority-interrupt displays was required both to display anomalous mark data which exceeded the threshold for acceptance and to permit alarm-type displays to override the first two levels. Since targeting programs or Extended Verbs run during the rendezvous programs, a third priority level was needed for alarm conditions and for marks that exceeded the auto-accept threshold. The three-level display hierarchy thus consists of normal displays, which are the lowest level and can be overriden by Extended Verb or mark displays, and third-level priority displays (alarm conditions, excessive updates) which can interrupt displays in both of the lower priority levels.

In addition to the three internally-generated priority-display levels described, the astronaut can key in two higher levels called external monitor request and non-monitor request. Altogether, five levels of display information are provided. After keying in a non-monitor request over an external-monitor request which in turn has overridden the three levels of internal priority display, an astronaut can return to the fourth external-monitor level from the fifth non-monitor level by keying KEY REL, and from external monitor to the third (priority) level via another KEY REL. He can then respond to the priority display and obtain the second and normal display levels, in turn, by keying appropriate responses to each succeeding display level. Thus, while monitoring a program computation and simultaneously taking navigation marks, the astronaut may be notified of an emergency-alarm condition by a priority display and may then initiate two levels of monitor-interrupt displays to discover the cause of the alarm condition before taking appropriate action.

The most significant effect of the additional display routines was that it became possible to have three levels of programs—with displays—running simultaneously. Response by the astronaut to any of the higher level displays would automatically cause the next lower-level display to reappear. This feature gives the astronaut the flexibility of using five levels of displays at a time.

2.1.3.4 Uplink and Downlink

Uplink is the digital telemetry system which enables ground control to load data or issue instructions to the AGC in the same manner employed by the astronaut using the DSKY keyboard. All information received by the AGC via uplink is in the form of keyboard characters. Each character is assigned an identifying code number called its character code. The AGC picks up the transmitted codes (these codes are the same as key codes) and enters a request to the Executive for the program which decodes and accepts them. The PINBALL program which decodes and accepts the transmitted code makes no distinction between inputs from the keyboard or from uplink, and any ground-command sequence normally transmitted via uplink may be duplicated by the astronaut using the keyboard.

The astronaut can choose to reject uplink from ground control by setting a toggle switch on the cockpit control panel to the blocked position.

A Universal Update Program exists in the AGC which facilitates updating the erasable memory and can be called by a number of extended verbs. To protect against the ingestion of erroneous information, the Update Program temporarily stores all new inputs in a buffer and transmits its contents back to ground control via downlink (see below) for verification. Furthermore, storage of state-vector updates (position and velocity) with their associated sphere-of-influence (earth or lunar) are delayed until current state-vector integration is finished.

The Update Program accepts four types of erasable-memory updates:

1. Contiguous Block Update provides ground-control capability to update up to 18 consecutive erasable-memory registers in the same erasable-memory bank.
2. Scatter Update provides ground-control capability to update from 1 to 9 nonconsecutive erasable-memory registers in the same or different erasable banks.
3. Octal-Clock Increment provides ground-control capability to increment or decrement the AGC clock with a double-precision octal-time value.

4. Liftoff-Time Increment provides ground-control capability to increment or decrement the AGC time, LM and CSM state-vector times and ephemeris time with a double-precision octal-time value.

This Universal Update Program capability has been available since SUNDISK (Apollo 7).

Downlink is the digital telemetry system which automatically selects lists (downlists) of internal AGC data for transmission to the ground downlink. Each downlist contains data pertinent to specific mission phases. COLOSSUS has five standard downlists: Powered, Coast and Align, Rendezvous and Prethrust, Entry and Update, and P22 (Orbital Navigation Program). LUMINARY has six standard downlists: Orbital Maneuvers, Coast and Align, Rendezvous and Prethrust, Descent and Ascent, Lunar Surface Align, and Initialization and Update of the Abort Guidance System (AGS). Whenever a new program is entered, a request for its list is made by placing the appropriate code into a downlink register. The downlink program then transmits the complement of this code as an identifier and uses it to select the appropriate list. The complete list is transmitted even if the program is changed during its transmission.

The standard AGC downlist contains 100 words (200 AGC registers). The AGC digital downlink is transmitted at a high rate of 50 words/sec or at a low rate of 10 words/sec. Thus, transmission of one downlist requires two sec at the high rate and ten sec at the low rate.

Certain data on the standard downlists are meaningful only when considered in multiregister arrays. Since the programs which compute these arrays are not synchronized with the downlink program, a "snapshot" is taken of these words so that changes in their values will not occur while these arrays are being transmitted to the ground. When a "snapshot" is taken, several words are stored at the time the first word is transmitted. The other words in the downlist are read at the time of transmission.

There is a special mode of downlink, called Erasable-Memory Dump, which can preempt the standard downlist being transmitted. The transmission consists of all of the erasable banks being transmitted sequentially. One complete pass through erasable requires 20.8 sec. The computer makes two passes through the complete erasable memory before returning to the standard downlist for the current mission phase. Since normal processing continues during the transmission of the Erasable-Memory Dump, some of the registers transmitted could have different contents on the second pass because they may have been recalculated during the transmission time.

This erasable-dump capability can be initiated using an Extended Verb and was developed to support postflight analysis; it can, however, be used whenever information not on a standard downlist is desired.

2.1.4 Error-Detection and Self-Check Features

Considerable effort has been expended over the years to uncover and correct for a number of hardware- or software-initiated problems. These problems can vary from a hardware power failure to the software getting caught in a loop. Both the hardware and software are designed to catch these problems, and the software procedures used to reinitialize (restart) the computer have become relatively standard.

The function of the hardware- and software-restart logic is to restore the current program with a minimum of disturbance to the mission. Fundamentally, this requires that certain specified tasks be called at the end of the correct time intervals (from a suitable base time), and that the specified jobs be reestablished with the proper priorities. In some cases, the proper "restarting" addresses for the jobs and/or tasks should not be at their beginning, but instead at some intermediate location or even at a special location entered only if a restart is encountered. These locations (restart points) are chosen to fall between computations such that when a restart occurs, the program resumes at a point in the program which precedes the place where the problem arose.

To accomplish the required restart functions, the various activities performed by the program software, in essentially independent computations, are divided into "restart groups"; there is provision in the restart software for six groups. One group, for example, might be concerned with the periodic powered-flight navigation cycling; another with orbital integration (perhaps required with powered flight to generate relative CSM/LM display data); a third with the timing of events leading to engine ignition; a fourth with generation of a time display on the DSKY; a fifth with computation of required velocity information for a rendezvous maneuver; and a sixth with a special computation performed shortly before engine ignition (to estimate the length of the burn). All six of these functions could be part of the complete program's computational load (as jobs or tasks) at one time and be in various stages of completion; and, consequently, they could be associated with separate restart groups. Not all computational activity in the program is restart-protected in this fashion; for example, should a restart occur while data are being loaded via the DSKY, the loading sequence must be reinitiated.

A restart group, therefore, can generally be considered to be associated with a particular functional software activity. Each group, in turn, is conventionally divided into a number of "phases" indicating just where the computations should be reinitiated in the event of a restart. The phase information for a given group is retained in both true and complemented form in the erasable memory, giving a total of 12 cells for the six pairs of cells associated with the six restart groups. When the restart software is entered, a check is made to ensure that all six pairs of cells have the proper internal complement relationship. If not, it is concluded that suspect information prevents the satisfactory resumption of computations, and the attempt to perform the restart is abandoned in favor of a FRESH START. FRESH START, which reinitializes the complete guidance system and essentially leaves it in an "idling" configuration, is discussed in Section 2.1.4.2.) The complement relationship could be destroyed if the erasable memory were modified by whatever caused the restart action, such as a power transient, or should the restart occur during certain portions of the programs that change restart-phase information.

Should the restart software conclude that adequate phase information is available (on the evidence of a proper complement relationship for the six pairs of phase data), the RESTART routine can be entered for each restart group that is "active" (a group is made "inactive" by setting the phase of that group to +0, indicating that none of its computations are restart-protected). The RESTART routine, depending on the value of the phase associated with that group, can cause jobs to be established and/or Waitlist tasks to be called at appropriate times via LONGCALL or the normal waitlist routines. The value of the phase information also determines whether one or two such jobs and/or tasks are to be reinitiated, and, additionally, whether the parameters associated with the reinitiation are to be obtained from fixed or erasable memory.

The value of the phase for a particular restart group, properly interpreted, is used to select an appropriate table entry in fixed and/or erasable memory. The table entries, separated by groups, are stored so that memory capacity is not wasted should there be more fixed-memory tables of one type than the other. The polarity with which information is stored in the tables is used to determine whether the table information pertains to a job, a Waitlist task, or a LONGCALL task, and, additionally, to determine which of several available options for defining the reinitiation parameters is to be employed.

During the course of the computations, it is necessary to update the phase value associated with the appropriate group. This can be done directly by loading new phase information into the appropriate group's phase cells or through use of one of several available phase-changing subroutines. The three most commonly used phase-changing subroutines are NEWPHASE, PHASCHNG and 2 PHSCHNG, all of which have a variety of options, depending upon the details of the calling sequence. Each one of these subroutines identifies the nature of the restart desired—fixed-memory table only, fixed and erasable tables, or erasable-memory table only.

The AGC restart mechanism provides great flexibility for restarting with optimal configuration of important computations, at almost no cost in erasable memory and little cost in execution time.

The significance of this restart protection can be appreciated more fully if one considers the consequences of the accidental knockdown of an unprotected engine-on bit during a burn. The following two sections describe remedies for hardware- and software-discovered difficulties and illustrate how self-check procedures contribute to the integrity of the mission program.

2.1.4.1 Hardware Restarts

One kind of program interrupt—a hardware restart—differs markedly from those described in Section 2.1.2.1. This special kind of program interrupt does not result in "normal" resumption of the program; it takes absolute priority over other program interrupts; it cannot be inhibited; and it can even interrupt an interrupt. As part of its generation, a special involuntary-interrupt instruction is produced, causing the hardware to generate a master-clear signal which knocks down all of the outbits.

A hardware restart can be triggered by such hardware problems as power failure, computer-oscillator failure, or parity failure. If the failure is transitory, the restart logic will resume the program flow.

A parity failure indicates possible malfunctions in a fixed or erasable register, in a sense line or in an amplifier. The AGC-stored word length consists of a sign bit, 14 magnitude bits of information and a parity bit. Whenever a register is addressed, odd parity must be observed or a hardware restart will occur. Should the parity error be detected in an erasable-memory register, it will be reinitialized and thus reset by the software-recovery logic. However, should a parity failure occur in a fixed-memory register, either a more serious physical problem exists or the astronaut has accidentally addressed an empty (unused) register.

Hardware restarts can occur upon the software-detection of a program-interrupt failure (RUPT LOCK) revealed if a program interrupt is continuously in effect for a specified period of time or if no program interrupt takes place within an equally long interval. Similarly, a transfer-control failure (TC TRAP) can be discovered. In addition, a special procedure called NIGHT WATCHMAN reveals the failure to address one specific memory location with a certain frequency, thus detecting the inadvertent entrapment in a large program loop.

Several lesser problems are indicated by warning lights and do not cause a restart: Counter Fail, which arises if counter increments occur too frequently or fail to occur following an increment request; PIPA Fail, which arises if no pulses arrive from the Pulsed Integrating Pendulous Accelerometers during a specified period, or if both positive and negative pulses occur simultaneously, or if too long a time were to elapse without at least one positive pulse and at least one negative pulse arriving; and Uplink Too Fast and Downlink Too Fast.

2.1.4.2 Software Restarts

Software restarts are programmed branches into the software-recovery logic. They use much of the same coding as the hardware restarts and, in fact, execute the actual restart in an identical fashion.

Software-restart logic is frequently useful to perform nonproblem functions such as stopping certain computations while allowing others to continue. It is also used when a new mission program is selected via V37. In this case, current processing is stopped; all scheduled jobs, tasks and interrupts are cleared out; all restart groups except the one used by the background-tracking program (if in progress) become inactive; the new program is set up in a restart group; and then the restart is executed to initiate the new program. Restart logic is used similarly in an abort from lunar descent, but in this case, the new program selected would be the abort program. Software restart procedures can also be initiated by such software-detected difficulties as too many tasks in the Waitlist system or a negative input to the square-root subroutine.

Two of the more important alarms which cause software restarts are BAILOUT and POODOO. A BAILOUT initiates a software restart for a problem from which recovery is expected, such as the overflow of job-register sets. A POODOO initiates a software restart for a problem from which a simple recovery is not expected, such as an attempt to take the square root of a negative number. Such a problem can happen if erroneous parameters have been loaded; consequently, a reinitialization of these same parameters will continuously yield the same alarm. In this case, normal computation flow is terminated and a flashing V37 (Change Program) comes up on the DSKY.

A FRESH START reinitializes the complete guidance system and essentially leaves it in an "idling" configuration with all of the output channels (outbits) and pending interrupts knocked down; at this point the program checks to see if the engine-on bit should be restored and if the IMU is in gimbal lock, and it takes whatever protective measures are necessary. FRESH START is the most radical reinitialization available for recovery.

A software program called BANKSUM Check, initiated by an Extended Verb to check all fixed and erasable memory for parity failures, is used principally for systems-test purposes. This routine sums the contents of the addresses within each fixed bank—halting temporarily when the last memory cell is reached. At this point a memory-cell summing routine included in the self-check portion of the fixed memory checks to ensure that the magnitude-of-the-sum is equal to the bank number and provides a DSKY display of the sum for operator review. The feat of having the magnitude-of-the-sum equal to the bank number is accomplished in the assembly process simply by adding an appropriate constant stored at the end of each bank to the correct value of each BANKSUM's magnitude.

As mentioned in Section 2.1.2.2, when no mission functions are being performed, an idling job (DUMMYJOB) is run to check for new jobs while checking fixed and/or erasable memory, depending on the option last selected by the astronaut.

Lastly, should the astronaut want to check the DSKY lamps, they can all be illuminated.

2.2 Major Mission Tasks Accomplished with the Computer Software

2.2.1 Early Approach to Navigation, Targeting, Guidance and Control

The navigation, targeting, guidance and control software specifies and manages the various spacecraft motions required to accomplish each mission phase. Functions of concern include the onboard measurement of rotational and translational motion, the processing of these measurements for display to the crew and ground control, the acceptance from the crew or ground control of desired spacecraft-maneuver instructions, and the execution of the defined maneuvers to change the spacecraft motion by modulating the firing of the various rocket-propulsion systems. In this context, navigation, targeting, guidance and control are defined as follows:

Navigation is the measurement and computation necessary to determine the present spacecraft position and velocity.

Targeting is the computation of the maneuver required to continue on to the next step in the mission.

Guidance is the continuous measurement and computation during accelerated flight to generate steering signals necessary to assure that the position and velocity changes of the maneuver will be those required by navigation measurements and targeting computations.

Control is the management of spacecraft-attitude motion—the rotation to and the stable maintenance of the desired spacecraft attitude during free-fall coasting flight and powered accelerated flight.

The appendices to this report present a functional description of these major program capabilities. Their design and development represent a significant portion of the Apollo software effort. The integration of these guidance, navigation and control programs with mission-oriented programs into a flight rope requires the comprehensive testing and verification effort described in Section III.

The early studies of the major program capabilities began, in most cases, well before the Apollo mission plan was finalized, since most of their concepts were fundamental to the overall task to be performed. For example, rendezvous procedures would be essential to both the earth-orbit and lunar-orbit rendezvous plans.

As a first step in MIT's software efforts, the basic organization of AGC computation and control had to be decided upon and implemented. PINBALL was developed to enable communication between the astronaut and the AGC. Guidance, navigation and control techniques had to be developed for every phase of the Apollo mission—from earth-orbit insertion to soft landing on the lunar surface to reentry into the earth's atmosphere. Similarly, abort procedures had to be developed for every phase. Studies determined the effect of the earth's luminous exponential atmosphere upon space navigation. Star- and horizon-sighting techniques had to be developed. Lunar-orbit determination using star-occultation measurements and the NASA Manned Space Flight Network were investigated. The effects of retrorocket exhaust velocity on visibility were ascertained. Development proceeded on a universal powered-flight guidance program tailored specifically to exploit the powers of an onboard digital computer. In addition, powered-flight steering of a spacecraft using a time-shared digital computer was studied, considering, of course, such factors as performance, response time and fuel conservation. And operating procedures had to be defined for the entire Apollo mission.

These are but a fraction of the many tasks which were studied and implemented before a mission-oriented rope could be integrated. These tasks continue. Flight experience frequently indicates the desirability of improvements or refinements. An example of such ongoing design work is the automation of the rendezvous sequence. Another is the restoration of the GN&C System Saturn-Take-over program as a backup system. With these exceptions and at this advanced date in the program, most changes are of a relatively minor nature.

The lunar-landing objective of the Apollo mission was finally achieved after many preliminary flights, each of which evolved from its predecessor (see Section 1.2). Each flight rope contained not only the programs necessary for the completion of its stated mission, but also many programs which were not of immediate application. In this fashion, existing flight ropes also served to bench-test programs which would be utilized in future flights. For example, the lunar-operations sequence was present in its entirety in SUNDANCE, the rope developed for a manned earth-orbital flight. But SUNDANCE provided the unique opportunity to exercise the lunar sequences in the comparatively safe earth-orbital environment. To prepare the actual lunar-landing sequence, however, those programs still had to be adapted to the conditions expected to prevail at the time of the lunar landing.

2.2.2 The G&N Mission Phases

For tractability the Apollo mission was divided into a number of discrete phases. Although each phase will be discussed somewhat independently, it is essential to note that all phases lead logically and efficiently from one to another in a stepwise fashion.

The lunar-landing mission, Apollo 11, contained all of the completed software programs. While many detailed variations can exist in future missions, the guidance, navigation and control functions remain essentially the same. A synopsis of a typical Apollo lunar-landing mission follows to aid in understanding the comprehensive task which the G&N software performs.

As stated above, the overall Apollo mission trajectory can be divided into several linked phases. Figure 2.2-1 illustrates thirteen such phases. The following paragraphs discuss each of these phases, along with lunar-surface operations.

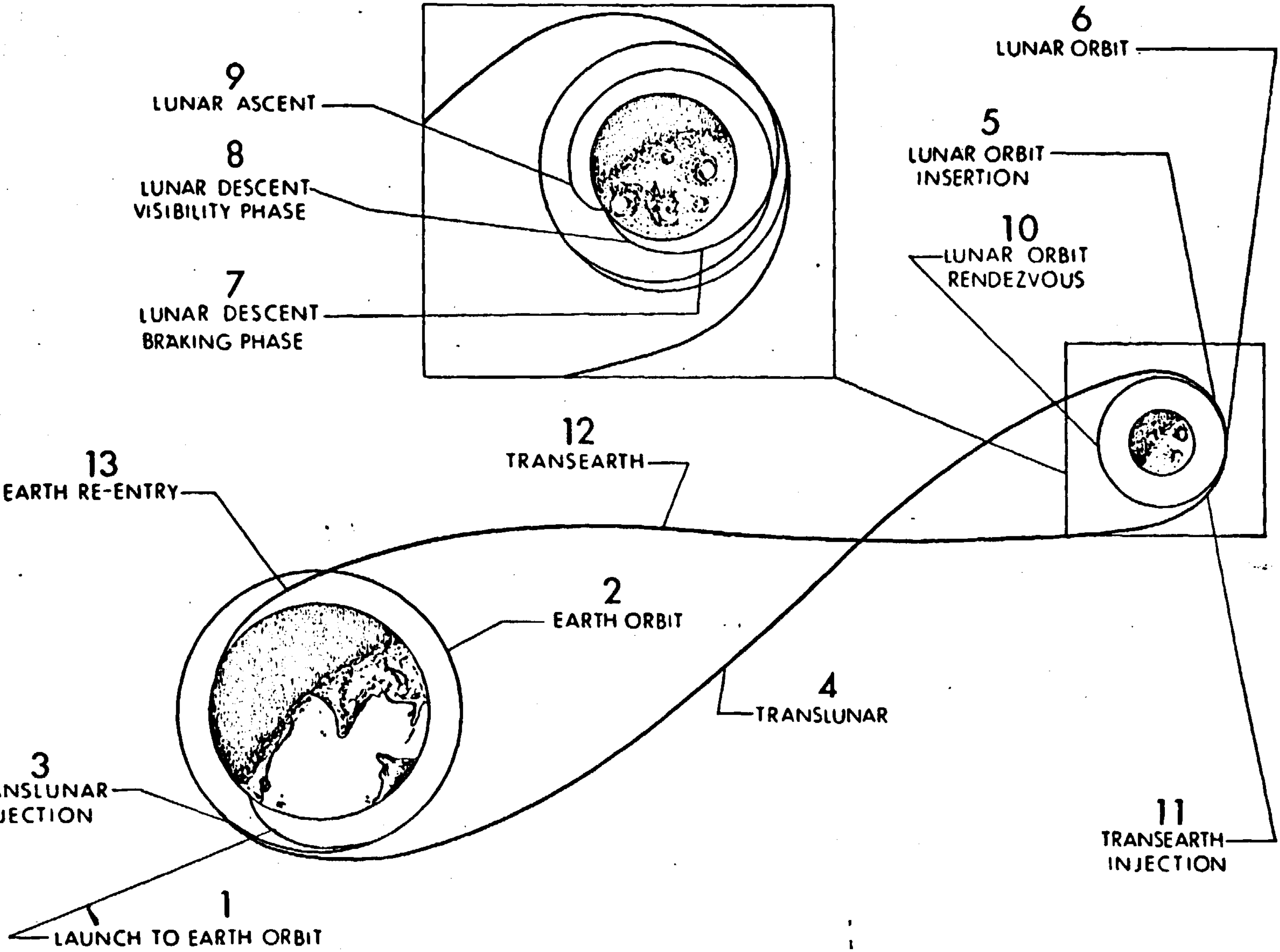


Figure 2.2-1 G&N Mission-Phase Summary

2.2.2.1 Launch to Earth Orbit

Prior to launch there is an intensive and intricate schedule of activity. Automatic programmed checkout equipment performs exhaustive tests of the major subassemblies in two major sequences: countdown demonstration and the actual countdown. Two operating sets of guidance equipment are prepared for the launch. The Saturn guidance equipment in the Saturn Instrument Unit controls the launch vehicle, while the Apollo guidance equipment in the Command Module provides a monitor of Saturn guidance during launch. The Lunar Module GN&C System, after prelaunch testing, is normally powered down for the launch phase of the mission.

Both sets of inertial guidance sensors, Saturn and Command Module, are aligned to a common vertical and launch-azimuth reference. During countdown, both systems are gyro-compassed to an earth-frame reference. Near liftoff, both systems respond to discrete signals to switch over from the earth reference to the nonrotating inertial reference used during boost.

During first-stage flight, the Saturn guidance system controls the vehicle by swiveling the outer four rocket engines. During the initial vertical flight, the vehicle is rolled from its launch azimuth to the flight-path azimuth. The Saturn guidance then controls the vehicle in an open-loop preprogrammed pitch maneuver designed to pass safely thorugh the period of high aerodynamic loading.

Both the Saturn and Command Module guidance systems continuously measure vehicle motion and compute position and velocity. In addition, the GN&C System compares the actual motion history with that expected from the Saturn control equation to generate an error display for the crew.

Shortly after the initial fuel-settling ullage and the second-stage thrust, the aerodynamic pressure approaches zero, the launch escape tower is jettisoned, and the vehicle passes out of the atmosphere. Any required abort, now, would normally be accomplished using the Service Module propulsion to accelerate the module away from the rest of the vehicle.

Since the problems of aerodynamic structure loading are no longer important, the Saturn guidance system now steers the vehicle toward the desired orbital-insertion conditions using propellant-optimizing guidance equations. Thrust-vector control is achieved by swiveling the outer four engines of the second stage.

During second-stage flight, the GN&C System continues to compute vehicle position and velocity, as well as several other flight parameters which can be displayed to the crew. The free-fall time to atmospheric entry, the apocenter altitude and pericenter altitude are the primary displays at this time.

The third Saturn stage (SIVB) has a single main propulsion engine gimballed for thrust-vector control. Roll control is achieved using the small SIVB roll attitude-control thrusters. The Saturn guidance system continues to steer the vehicle to orbital altitude and speed. When orbit is achieved, the main SIVB propulsion is shut down; this usually occurs at about 12 minutes after liftoff on a 100-mile circular orbit.

During the second- and third-stage boost flight, the Command Module is configured to allow the crew to take over the SIVB steering function manually, should the Saturn guidance system indicate failure. Should this switchover occur, presumably the mission could be continued. More drastic failures would require an abort using the Service Module propulsion system.

2.2.2.2 Earth Orbit

The Apollo spacecraft remains attached to the Saturn SIVB in earth orbit. The Saturn system controls attitude by commands to the small SIVB reaction-control thrusters for pitch, yaw and roll.

Ground-tracking navigation data telemetered from the Manned Space Flight Network (MSFN) stations are available to correct the position and velocity of the Saturn navigation system and to provide navigation data for the GN&C System via uplink telemetry. The inertial-subsystem alignment in the Command Module may also be updated by star sightings with the optical subsystem. For these measurements, the crew exercises manual control of vehicle attitude through the Saturn attitude-control system.

Typically, the earth-orbital phase lasts less than three hours for systems checkout before the MSFN-computed signals are transmitted to the Saturn system to initiate the translunar-injection maneuver.

2.2.2.3 Translunar Injection

Translunar injection is performed using a second burn of the Saturn SIVB propulsion system. Saturn guidance and control systems again provide the necessary steering and thrust-vector control to the near-parabolic velocity that puts the vehicle on a so-called "free return" trajectory to the moon. This trajectory is constrained ideally to pass in back of the moon and to return to earth-entry conditions without additional propulsion.

As before, the GN&C System independently generates appropriate parameters for display to the crew for monitoring purposes. Should the Saturn guidance system indicate failure, steering takeover by the crew is possible. The typical translunar-injection thrusting maneuver continues for slightly over five minutes' duration before the SIVB is commanded its final shutdown.

2.2.2.4 Translunar

The spacecraft configuration injected into the translunar free-fall must be reassembled for the remaining operations. An adapter in front of the SIVB houses the LM until this phase of the flight. The astronauts separate the Command and Service Modules from the SIVB and then turn the CSM around for docking with the Lunar Module. To accomplish this, the pilot has a three-axis left-hand translation controller and a three-axis right-hand rotational controller. Output signals from these controllers are processed in the Command Module computer to modulate the firing of the 16 low-thrust reaction-control jets for the maneuver. The normal response from the translation controller is proportional vehicle acceleration in the indicated direction. The normal response from the rotational controller is proportional vehicle angular velocity about the indicated axis.

During the separation and turnaround maneuver, the SIVB control system holds the Lunar Module attitude stationary; this allows for a simple docking maneuver of the Command Module to the Lunar Module docking hatch. The SIVB, Saturn Instrument

Unit, and Lunar Module adapter are staged to leave the Apollo spacecraft in the translunar flight configuration. A further short maneuver puts the SIVB on a separate trajectory which will not interface with the Apollo spacecraft.

Very soon after injection into the translunar free-fall coast phase, MSFN-computed navigation measurements are examined to determine the acceptability of the trajectory. These data indicate whether there is a need for an early midcourse maneuver to correct any error in the flight path which might propagate with time to a larger value, thus avoiding a needless waste of correction-maneuver fuel. This first correction is made—perhaps a few hours from injection—only if it is needed. Ground-tracking data can be telemetered to the spacecraft anytime they are available. Using these ground data or horizon-to-star angle measurements obtained from the onboard sextant, the onboard computer can correct the knowledge of the spacecraft state vector—position and velocity.

Mission control on the ground periodically examines the ground-based radar data for uncertainty in position and velocity and the estimate of indicated velocity correction required to improve the present trajectory. If the indicated position and velocity uncertainties are suitably small and the indicated correction is large enough to be worth the effort, the crew may execute the telemetered midcourse correction. Each midcourse velocity correction requires, first, an initial spacecraft orientation which aligns the estimated direction of the thrust axis along the desired acceleration direction. Once the thrust direction is aligned, the rocket is ignited and controlled by the GN&C System.

Typical midcourse corrections are of the order of 30 ft/sec or less. If a required correction happens to be very small, it is made with the small reaction-control thrusters. Larger corrections require a short burn of the service-propulsion rocket. The direction and magnitude of each burn adjust the trajectory so that the moon is finally approached near the plane and pericynthion altitude that provide for satisfactory conditions for the lunar-orbit insertion and lunar landing.

During the translunar phase, mission control periodically transmits blocks of data via voicelink to the crew to permit safe return in the event of loss of communications. These data include state-vector updates to be loaded by the crew at the appropriate time into the AGC. The data are sent as a precaution against the contingency that telemetry and/or voice communication fail prior to the next scheduled update. These updates occur at about ten-hour intervals.

2.2.2.5 Lunar-Orbit Insertion

Prior to lunar-orbit insertion maneuvers, as with all normal thrusting with the Service Propulsion System, the inertial subsystem is realigned using star sightings. Then the GN&C System generates initial conditions and steering parameters based upon targeting parameters telemetered from the ground. The guidance programs initiate engine turn-on, control the direction of the acceleration appropriately, and shut the engine down when the maneuver is complete. Lunar-orbit insertion maneuvers are the two burns typically intended to put the spacecraft in an orbit of approximately 60 nmi altitude. The first thrusting maneuver, behind the moon, slows the spacecraft so that it will be "captured" by lunar gravity into a highly elliptical orbit and not pass on free-return to earth. Then, the second burn, at perilune behind the moon, circularizes the orbit. The plane of the orbit is selected to pass over the preplanned landing region.

2.2.2.6 Lunar Orbit

In lunar orbit, navigation measurements may be made to update the knowledge of the actual orbital motions. A particularly important sighting—that to the intended landing target—provides data for the site's precise location in the lunar navigation coordinate frame. Sufficient measurements must be made and combined with ground-tracking data to provide accurate initial conditions to the Lunar Module guidance system for the LM's controlled descent to the lunar surface.

2.2.2.7 Lunar Descent

During lunar orbits, before separation, the Lunar Module GN&C System is turned on and receives a checkout and its initial conditions, and the rendezvous radar (RR) is self-tested. Before initiation of the Lunar Module descent-injection maneuver, the vehicles are separated; the Lunar Module inertial subsystem receives final realignment from star sightings; the directional tracking and ranging operation of the RR is checked against the radar transponder on the CM; and the maneuver attitude is assumed. The maneuver is made using Lunar Module descent-stage propulsion under control of the module's GN&C System.

During free-fall phases of the Lunar Module descent, the Command Module can make optical tracking and VHF range-only measurements of the Lunar Module for confirmation of its relative orbit. For that part of the trajectory in front of the moon, earth-based tracking provides an independent check. The RR continues to track the CM transponder throughout free-fall for additional trajectory corroboration. At lower altitudes, the Lunar Module landing radar on the descent stage is self-tested prior to powered descent-insertion. Alignment updating of the Lunar Module inertial subsystem is also performed.

2.2.2.7.1 Braking Phase

Powered-descent braking begins when the descent engine is reignited; the velocity- and altitude-reducing maneuver is controlled via the Lunar Module inertial subsystem and autopilot calculations in the computer.

The descent-stage engine can be throttled over the range necessary to provide initial braking and to provide controlled hover above the lunar surface. Engine-throttle setting is automatically commanded by the guidance system to achieve proper path control, although the crew can override this signal with several alternative control modes, if desired.

Thrust-vector control of the descent stage is achieved by a combination of body-fixed reaction jets and limited gimballing of the engine. The engine gimbal angles follow guidance commands in a slow loop (fixed rate command of approximately 0.2 deg/sec), thus causing the thrust direction to pass through the vehicle center of gravity—and minimizing the need for continuous fuel-wasting torques from the reaction jets.

During all phases of the descent, the operations of the various systems are monitored from onboard and earth-based radar. The landing can be retargeted by uplink telemetry or the mission could be aborted for a number of reasons. If the GN&C System performing the descent control is still operating satisfactorily, it would control the abort back to rendezvous with the Command Module. If the primary guidance system has failed, the independent backup Abort Guidance System could steer the vehicle back to orbital conditions for rendezvous.

2.2.2.7.2 Visibility Phase

One significant feature of this phase is that the controlled trajectory is selected to provide the Lunar Module crew with visibility of the landing surface. The vehicle attitude, descent rate, and direction of flight are all essentially constant, so the landing point being controlled by the guidance appears fixed, relative to the window. A simple reticle pattern in the window indicates this landing point in line with a number denoted by computer display. Should the astronaut observe that the landing point is in an area of unsatisfactory surface features relative to other areas nearby, he can select a new landing site for the computer-controlled landing. Alternately, the astronaut has the option of taking manual control of this landing maneuver at any time.

Automatic guidance control during the braking and visibility phases uses weighted combinations of inertial-sensing and landing-radar data, with the weighting dependent upon expected uncertainties in the measurements. The landing radar includes altitude measurement and a three-beam Doppler measurement of three components of Lunar Module velocity with respect to the lunar surface.

At any point in the landing, the astronaut can elect to assume partial or complete control of the vehicle. For instance, one logical mixed mode of operation would have the rate-of-descent controlled automatically by modulation of the thrust magnitude and astronaut manual control of attitude for horizontal maneuvering.

Near the lunar surface, the spacecraft enters a hover phase which may have a variety of conditions, depending upon mission ground rules, crew option and computer program. Descent-stage fuel allowance provides for hovering before touchdown. If hovering is not accomplished, an abort is initiated on the ascent stage. The crew makes final selection of the landing point and maneuvers to it either by tilting the vehicle or by operating the reaction jets for translation acceleration. The inertial-subsystem altitude and velocity computation is updated by the landing radar so that, as touchdown is approached, good data are available from the inertial sensors, since the flying dust and debris caused by the rocket exhaust degrade radar and visual information. Touchdown is made with the spacecraft near vertical and with a downward velocity of less than 4 ft/sec.

2.2.2.8 Lunar-Surface Operations

The period on the moon includes considerable activity in exploration, equipment deployment, experimentation, and sample gatherings. Also during this time, spacecraft systems are checked and prepared for the return. For example, the ephemeris of the Command Module in orbit is periodically updated, and the information is relayed to the Lunar Module crew and computer. The Lunar Module rendezvous radar can also track the Command Module as it passes overhead to provide further data upon which to base the ascent-guidance maneuvers. The inertial subsystem receives final alignment from optical star or planet sightings prior to the start of ascent or, as a backup, the vertical components of this alignment can be achieved by accelerometer sensing of lunar gravity in a vertical-erection loop. Still another backup mode involves using computer-stored knowledge of the spacecraft's inertial alignment at touchdown. Liftoff must be timed to achieve the desired trajectory for rendezvous with the Command Module.

2.2.2.9 Lunar Ascent

Launches from the lunar surface leave the descent stage of the Lunar Module behind, and can be initiated over a range of time by entering a holding orbit at low altitude until the phasing is proper for transfer to the Command Module. A desirable constraint on all ascent-powered maneuvers, as well as abort maneuvers during the landing, is that the following coasting trajectory have sufficient altitude to avoid intersection with the lunar surface. This is a safety consideration which allows for the possibility of failure of the engine to reignite. If the Lunar Module engine thus fails, the spacecraft could then safely coast until a rescue maneuver by the Command Module is accomplished. That is, the Command Module could execute "mirror images" of those thrusting maneuvers that the Lunar Module would have normally performed. Thus, the Lunar Module can be the passive vehicle in the rendezvous exercise.

The initial part of the ascent trajectory is a vertical rise followed by pitchover, as commanded by the guidance equations. The ascent-engine maneuvers are under the control of the GN&C System. The ascent engine is fixed-mounted and nonthrottleable; consequently, thrust-vector control is achieved by complementing the engine thrust with that of the 16 reaction-control jets mounted on the ascent stage. Required commands from guidance terminate thrusting when a suitable rendezvous coast trajectory is achieved.

2.2.2.10 Lunar-Orbit Rendezvous

This phase starts from the low holding orbit achieved by the ascent burn of the previous phase. From this orbit, the RR makes direction and range measurements to the Command Module for refinement of the navigation data in the Lunar Module computer. The phasing of motion between the two vehicles eventually reaches a specific point at which a standard transfer burn puts the Lunar Module on an ascending trajectory to intercept the orbiting Command Module. During this period, radar measurements provide data for the Lunar Module computer's small velocity corrections needed to establish a more accurate intercept trajectory. Coasting continues between and during these corrections until the range to the Command Module is reduced to a few miles.

A series of braking maneuvers under control of the Lunar Module GN&C System and the astronaut is required during the terminal rendezvous phase. During this phase, data from the inertial sensors and the rendezvous radar are utilized. The Command Module pilot can monitor progress with the sextant, with VHF ranging, and with the computer-contained rendezvous program. This operation reduces Lunar Module velocity relative to the Command Module to zero at close range, leaving the Lunar Module pilot in a position to initiate a manual docking maneuver with the translation and rotation control of the reaction jets. These maneuvers are normally done with the Lunar Module, although propulsion or control problems could require the Command Module to take the active role. After final docking, the Lunar Module crew transfers into the Command Module. The remaining ascent stage of the Lunar Module is then jettisoned.

2.2.2.11 Transearth Injection

Navigation measurements made while in lunar orbit determine the proper initial conditions for transearth injection. These measurements are performed as before, using available onboard and earth-based tracking data.

The guided transearth injection, which of necessity is performed behind the moon, is normally made under the control of the GN&C System. Targeting for this maneuver is normally provided by uplink telemetry before the spacecraft passes behind the moon. Several backup means are available to cover possible failures in the primary system. The injection maneuver is controlled to put the spacecraft on a free-fall coast which will attain satisfactory entry conditions near earth.

2.2.2.12 Transearth

The transearth phase is very similar to the translunar phase. During the long coasting phases going to and from the moon, the systems and crew must control the spacecraft orientation. Typical midcourse orientation constraints include ensuring that the high-gain communication antenna can point to earth while remaining within its gimbal limits; that the proper omnidirectional antenna is selected by the crew; and that the spacecraft attitude is not held fixed relative to the sun for too long a period, thus minimizing the effect of local heating. Consequently, a passive thermal-control mode (barbecue) is normally used via the GN&C System to change spacecraft attitude slowly, relative to the sun line-of-sight.

Onboard and ground-based navigation measurements nominally lead to a series of three midcourse correction maneuvers during the transearth flight. Very accurate transearth injection has made it probable that one or more of these maneuvers may be deleted. The aimpoint of these corrections is the center of the safe earth-entry corridor suitable for the desired landing area. This safe corridor is expressed as a variation in flight-path angle of -6.5 deg ±0.05 deg, measured with respect to the local horizontal. A too-high entry could lead to a skipout from the atmosphere; a too-low entry could lead to atmospheric drag decelerations exceeding the crew tolerance.

After safe entry conditions are confirmed by navigation, the inertial platform is aligned or realigned, the Service Module is jettisoned, and the initial entry attitude of the Command Module is achieved.

2.2.2.13 Reentry'

Initial control of entry attitude is achieved by GN&C System commands to the 12 reaction jets on the Command Module. As the atmosphere is entered, aerodynamic forces create torques determined by the shape and center of mass. These torques are in a direction toward a stable trim orientation, with the heat shield forward and the flight path nearly parallel to one edge of the Command Module's conical surface. The entry digital autopilot in the GN&C System now operates the reaction jets to damp out any oscillation about this trim orientation. The resulting angle of attack of the entry shape causes an aerodynamic lift; this force is used for entry path control by rolling the vehicle about its wind axis under control of the GN&C System. Range control is achieved by rolling, so that an appropriate component of the lift vector is either up or down, as required. Cross-range control involves rolling the spacecraft so that the lift vector points right or left of the flight path, as required.

Safe reduction of high velocity to suborbital conditions through the energy-dissipation effect of the atmospheric drag forces is the first concern of the entry guidance. At lower velocity, controlling to the earth-recovery landing area is included in the automatic guidance; manual entry maneuvers can also be used as a backup mode. Velocity continues to decrease until deployment of the drogue parachutes. Final letdown is normally by three parachutes to a water landing.

2.2.3 Rope Design Philosophy and Problems Encountered

The principal flight software efforts which, when integrated together, allow such a complicated mission to succeed are coasting flight navigation, targeting, powered-flight guidance and navigation, and digital autopilots. The philosophy which guided the design, development and integration of each of these tasks is presented in this section, and a functional description of each is presented in the appendices.

Early in MIT's Apollo software effort, the engineer who designed a mission program was also responsible for the coding and testing of that program. Because early programs were to fly in unmanned, fixed-sequence flights, mission programs were arranged in a fixed, predefined sequence. AGC memory capacity seemed ample, and programming and verification were relatively simple and straightforward.

With each successive rope, the software task became decidedly more complicated. With the arrival of manned flights, provision for astronaut interaction brought about a requirement for nonfixed program sequences with interfacing routines. The necessity arose for several programs to run simultaneously. Memory requirements began to grow at a staggering rate*. Finally, the mission programs themselves became so complex that it became virtually impossible for an individual design engineer to accomplish all the design, programming and verification tasks by himself. Clearly, the need for a formal design philosophy was at hand.

Mission programs were apportioned into standardized computational, service and interfacing routines. Furthermore, nearly every program was modularized so that there were no assumptions concerning program sequence, except where mandatory. Consequently, the program became tractable,.allowing the allocation of analysis, programming and verification to expert programming individuals—each of whom was to become a specialist in his own area.

With this modularization of the programs, it became apparent that many could run in parallel. (The CM AGC Executive allows up to seven to run in parallel, and the LM AGC Executive allows up to eight.) Parallel operation would create DSKY display conflicts, however, because PINBALL originally restricted to one the number of programs which might have access to the DSKY at any one time. But these conflicts had been anticipated, since the multiple-level, DSKY-display capability was being developed concurrently. Furthermore, the DSKY-display capability provided a standard display interface for all programs and established a useful mechanism for restarting programs (see Sections 2.1.3.3. and 2.1.4). The modularization of the programs, together with the multiple-level DSKY displays, allowed the flexibility and program manageability needed to accomplish the Apollo mission.

Problems were attendant throughout the development, however.

Great care had to be exercised in the allocation of erasable memory, since the demand exceeded the available registers; the sharing of erasables wherever possible became standard. With the enlarged staff of programmers, careful control was more critical than earlier. Each individual programmer concentrated on a particular aspect of the program, and frequently was unfamiliar with areas other

*Even the relatively simple Apollo 4 program had required no less than 87 percent of the Block I computer memory.

than his own. Considerable effort* was expended in the allocation of erasable storage and in the prevention or correction of erasable-memory conflicts.

Difficulty in "shoehorning" erasable storage** and the ever-attendant problems of erasable-memory conflict were not the only vexations imposed by the meager AGC erasable memory (2048 words): erasable sharing brought on external restraints, causing programs to become less flexible—they had to be programmed to conserve erasable memory even at the cost of simplicity and execution time; and many basic subroutines could not be made reentrant.

From the beginning, restart protection has been provided for all the ropes—at a cost in fixed memory, execution time, and complexity (complexity because a restart could occur anywhere in the program). One school of thought felt such protection was unnecessary; it was unlikely, this viewpoint held, that such a restart would occur in flight at all, and any that did occur would probably be during an unimportant part of the program. However, a more conservative philosophy prevailed, providing safe error recovery—a sobering factor, since little or no redundancy was provided for fault tolerance in the hardware. Simpler, more obvious programming techniques, which might have averted some of the problems encountered, were not used if it were felt that they might restrict the scope and usefulness of the program.

Gradually, provision has been included in the software to check against astronaut procedural errors and to back up hardware failures with alternative software processing; several software procedures have been implemented to ensure that failures of critical switches and indicators can be overcome by special provisions within the program.

*At one point an Erasable Committee, consisting of the Assembly Supervisor and representative experts from each of the major areas, would adjudicate every request for an erasable word or bit.

**COLOSSUS 237 (Apollo 8) flew with only 15 unused erasable words, and LUMINARY 69 (Apollo 10) with only 5.

SECTION III

TESTING, VERIFICATION, AND MISSION SUPPORT

For each flight a new assembly of onboard computer programs is integrated and tested. Improvements over the previous flight are included and parameters improvements over the previous flight are included, and parameters are changed to meet specific flight objectives. As mentioned in Section I, this completed assembly of hard-wired and erasable memory is known as a "rope", a name taken from the weaving process by which the fixed memory is manufactured. The present section of this report describes MIT's continuing effort in the qualification and support of each new rope. The support effort is varied in nature. Before release for manufacture, the rope undergoes a vigorous testing and verification program. Specification change procedures provide NASA with control over the software system. Documentation is generated for training and information purposes, as well as for specification control. MIT also supports the Apollo missions by training crews, flight controllers and others, by providing support personnel to NASA, and by actively monitoring each flight.

3.1 Testing and Verification

3.1.1 Testing Philosophy

Because the lives of astronauts are at stake, all components of the Apollo system must undergo exceptionally stringent testing. Schedules have been tight and launches frequent; thus, timely, well-managed testing programs have been necessary. The testing program for Apollo software was designed under additional constraints, because the software is subject to constant change. Improvements are continually suggested by the astronauts, NASA and MIT—even up to the time of launch. The fixed memory, however, must be tested and released for manufacture three to four months prior to flight, to allow for manufacturing time and for integrated testing of the complete vehicle. Thus, an obvious conflict arises between the desire for improvements and the need for testing. As a result, MIT must perform a large amount of work in a short period of time—and with very high accuracy.

In general, the MIT testing program encompasses two major areas—computation and logic. The mathematical portions of the program are tested for computational accuracy, both to discover programming errors and to identify degradation in accuracy resulting from such factors as truncation and roundoff. It is also necessary to test the entire program sequentially to ensure that the proper logical sequences occur.

The first step in the testing program is the preparation of comprehensive test plans. A test plan specifies the objective of the test, the broad initial conditions and the sequence of program operation, and it identifies the criteria (test points) upon which the results are to be judged. Preparation of test plans requires the cooperation of the designers and programmers who are intimately acquainted with the particular coding being tested, as well as coordination by those familiar with the overall program structure. Test plans thus serve to organize, control and evaluate the testing program.

After preparation of the test plan, the second step in the testing procedure is to generate specific initial-condition data and a detailed operating sequence, including astronaut operations when applicable. The third step is to perform the test on the All-Digital or Hybrid Simulators and to collect the test-point data from on-line printouts and post-run edits. Comparison data are collected from other simulations. The fourth step is to compare the test-point data from the various sources and to make a judgment concerning the future course of the test. It is not unusual for a test to go through the second, third and fourth steps repeatedly before being judged successful. The final step is the documentation of the test.

Testing procedures developed along with the programs. In the early conceptual and engineering stages, MAC* programs were written by the designers to test their ideas before AGC coding was started. When small pieces of AGC coding were completed, they were individually tested to see that all logical branches were correct and that they yielded the desired arithmetic outputs. As these pieces of coding were integrated to form larger blocks, interfaces were tested to verify that the

*As explained in Section I, MAC is a high-level programming language for general-purpose computers, developed at MIT for scientific applications. It is not to be confused with MIT's Project MAC. The latter was named independently, some years later, and is unrelated to the MAC language.

pieces of coding which were tested independently would also work together. Much insight and planning were necessary to ensure that sufficient representative tests were run on combinations of programs, since the length and complexity of the integrated system software made it impossible to test every conceivable sequence of events. However, all of those sequences which could reasonably occur for a particular mission were vigorously tested.

As work progressed from subroutines to the major-program level, testing emphasis shifted from the individual bits and branches to the overall performance, computational accuracy, scaling problems, and major logic flow. It was important to determine whether the design was adequate to perform the required functions.

As it reached completion, the integrated flight rope required performance and stress testing. Typical mission sequences, such as navigation, targeting and powered flight, were simulated. Testing was also designed to ensure that the computer could accomplish all the required tasks in real time. (If the AGC is asked to do too many things at once, a restart will occur, and valuable time will be lost.) It was also important to test the effects of off-nominal procedures and data upon computer functioning.

After the early missions were flown and the testing program became well defined, it became unnecessary to duplicate the above testing for each new mission. The program worked—only the changes and additions needed exhaustive testing. As a flight approached, the testing emphasis shifted towards those program sequences and combinations which were anticipated for the mission.

3.1.2 Levels of Testing

Formally, the testing effort has been subdivided into six levels:

Level 1 testing was part of the early design effort. As a particular set of specifications was created, design engineers coded the equations in MAC and performed various test cases to identify possible computational and logical difficulties, such as loss of acceptable accuracy and range of variables.

Level 2 testing began when a block of AGC coding was completed for the above specifications. The programmer would test the coding on the All-Digital Simulator. Only those factors directly influencing the block of coding were included in the simulation. Results of Levels 1 and 2 testing were compared and distributed among MIT personnel.

Eventually, Levels 1 and 2 were combined by building edit programs in the All-Digital Simulator which processed the data through both MAC and AGC equations, and printed comparisons. As the overall programs became well developed, new design changes would thus undergo "unit testing", which took the place of Levels 1 and 2.

Level 3 testing was done by the programmers to verify the operation of complete programs or routines. Digital and hybrid simulations were used to ensure that the smaller blocks of coding fit together logically. As each logical path of the coding was tested, it was traced on a master copy of GSOP, Section 4, including test number and date. (Section 4 is the NASA-approved specification document for software-logic flow, as discussed in Section 3.2.1 of this report.)

Level 4 testing required the cooperation of designers and programmers, using both digital and hybrid simulations. Sequences of several programs were tested, corresponding to possible mission usage. These tests verified the proper communication from program to program and investigated conflicts in such areas as erasable-memory usage and time sharing, between the major programs. Test points were compared with the edit programs in the All-Digital Simulator, or, if edit programs were unavailable, with an engineering simulation. Completion of Level 4 testing corresponded to release of a program for manufacture.

The programs underwent continual change during Levels 3 and 4 testing due to new specifications, as well as problems uncovered by the testing program. Level 5 testing repeated all the Levels 3 and 4 tests on the final rope which was released for manufacture, and thus verified the continual validity of these earlier tests.

Level 6 testing, which took place after the rope was released for manufacture, made use of the All-Digital Simulator to establish performance specifications, and the Hybrid Simulator to reveal program anomalies. Level 6 testing on the All-Digital

Simulator was oriented toward the particular flight; these tests used the expected timeline, operational trajectories, procedures and erasable data. Expected one-sigma and three-sigma errors in equipment and in state vectors were employed to give a broad range of performance data. Results of the tests were analyzed by the designers and programmers, and presented to NASA as predictions of the Guidance, Navigation and Control System's performance.

3.1.3 Testing Tools

Software designers and programmers used various simulations in the development and testing of the flight programs. The All-Digital Simulator bore the largest brunt of the testing effort. It afforded the most precise and repeatable simulation of the AGC and its environment. The Hybrid Simulator permitted the tester to interface directly with a program by means of a DSKY and to make on-the-spot changes if necessary. The Engineering Simulator provided quick turnaround, thus permitting multiple runs with changes in many parameters. The following sections briefly describe each of these simulations, with emphasis on those aspects pertinent to their use in the testing and verification program.

3.1.3.1 All-Digital Simulator

The All-Digital Simulator has been the most powerful tool in the verification program. It exists entirely as coding on a general-purpose digital computer, and is composed of two logically independent sections, linked by an interface routine. The AGC Instruction Simulator simulates the operation of the Apollo Guidance Computer, both in storage layout and in detailed arithmetic and logical operation. The Environment, made up of a number of MAC-coded subroutines, simulates all relevant aspects of the hardware and flight environment within which the AGC operates. This environment includes effects of the engine, spacecraft dynamics, optics, IMU, radar, astronaut interactions, atmospheric and gravity effects, and celestial-body motion. Almost every aspect of the environment which can conceivably interact with the flight program is included.

During a simulated sequence, the Instruction Simulator advances through the AGC program, instruction by instruction, simulating the detailed operations performed by the AGC in executing each instruction. After each instruction cycle, the state of the simulated computer, including such factors as instruction sequencing, contents of erasable storage, interrupt activity and clock incrementation, is identical to the state of an actual AGC executing the same program; in addition, truncation, round-off, overflow and timing exhibit the same behavior on the simulated AGC as they do in the real one.

In the course of advancing through the AGC program, the Instruction Simulator encounters instructions which refer to input or output operations, such as the reading of an input counter or the setting of an output discrete. A program known as the Communicator examines all such input/output references and determines whether immediate interaction with the Environment simulation is required by the specific action of the AGC. When input data are required by the Instruction-Simulator, the Communicator tries to provide this information by extrapolation from the previous Environment state. If this can be done, control returns immediately to the Instruction Simulator. Should the Communicator not have a valid extrapolation formula, there will be a full Environment update. In general, the Communicator updates the Environment over the longest possible time interval consistent with maintaining simulation accuracy.

By maintaining a high degree of similarity between the simulated and the real AGC-Environment interface, the simulated AGC can be subjected to computational loads and dynamic situations which closely approximate the conditions of a real mission. Precision in the simulated AGC performance is degraded primarily by inaccuracies in AGC or Environment models. These inaccuracies may be deliberate, representing a compromise between fidelity and computational speed, or may be of unknown cause and difficult to evaluate; however, the inaccuracies are all within the precision needed to test the programs vigorously.

In addition to providing the Instruction Simulator with all the necessary inputs for the simulation to run, the Environment serves as a standard against which flight software performance can be judged. This is because many of the tasks required of the AGC involve measurement and computation of factors in the external surroundings, such as spacecraft attitude and trajectory, the effects of gravity, and sensor errors. Inaccuracies can arise in these AGC computations for a number of reasons: information from the sensors may be imperfect; the measurements available may have to be processed before the information required can be obtained; space and time limitations in the AGC, with its short, 15-bit single-precision accuracy, also introduce errors; finally, programming errors can lead to subtle or gross miscalculations. All of these error sources are represented in the All-Digital Simulator. However, the Environment portion of the simulator has available or can generate the "true" value of the quantity being measured and the "true" value of the quantity being computed. Although the "true" quantities in the Environment simulation are obtained from finite precision mathematical models, the 64-bit accuracy of the MAC-coded environment is far greater than the AGC provides, and the models are more comprehensive than those used in the flight programs. For example, the Environment can compute the "true" altitude of the Lunar Module above the simulated lunar surface. This altitude can serve as a standard by which to judge the AGC-computed altitude. Post-run edits permit the user to make this type of comparison on any pertinent section of the software.

The Digital Simulator provides the user with numerous output options, traces, dumps and edits, which permit detailed analysis of AGC performance. Before processing each instruction, the Instruction Simulator checks whether there is a user-interrupt attached to that instruction. These interrupts can be initiated by accessing a memory location, or can be made conditional upon various parameters of the computer state or upon the number of accesses to a location. Thus, the user can interrupt the program to dump onto magnetic tape any portion of the AGC memory or the Environment. He can flag the time an instruction occurs, change any register, or even terminate the run. The user may periodically dump a "snapshot" of the entire simulator from which a subsequent simulation can be initiated. This feature, commonly called "rollback", is extremely valuable when many hours have been invested in a simulation run that has terminated for one reason or another. The results may be examined, changes made to the AGC program or the Environment,. and the run continued in a deterministic manner. Since the simulation is entirely digital, it has bit-by-bit repeatability, and any changes between runs can be attributed to modifications by the user. As previously mentioned, the editing capability causes information to be stored and then analyzed at the end of the simulation by a MAC program.

Generally, debugging of AGC programs proceeds by testing individual elements of programs on the Digital Simulator separately, and then gradually merging the elements into a working rope. The AGC programmer uses the simulation in early stages of development to debug preliminary coding. The program under test is executed in a simulation, and, by using the various diagnostic tools, the programmer can determine where errors exist.

In later stages of rope development, the Digital Simulator can be used to verify the adequacy of the various guidance, navigation and control programs to perform required tasks in a flight environment. The implementation of specific guidance, navigation or control laws on the AGC often leads to problems with scaling, job sequencing, or timing. These problems may be uncovered in simulation and result in redesign of some of the control algorithms. The closed-loop simulation of the AGC interacting with the vehicle is able to test the adequacy of the steering and autopilot design in many ways that are not possible through analysis alone. In the final stages of program development, the simulator may be used to generate long verification runs which demonstrate the full mission capability of the rope.

The All-Digital Simulator plays the largest part in the testing and verification program, Among its advantages are the exact reproducibility of tests, and the availability of many user options. One disadvantage is that the user cannot interface directly with the program. All required environment and astronaut actions must be decided upon before the test, and changes cannot be made until computer printout is returned to the user. Another disadvantage is that, in a few circumstances, the simulation may be forced to run much slower than real time, as when high-frequency bending is being simulated, and the Instruction Simulator has to wait while digital approximations are being calculated in the Environment. For these cases the Hybrid Simulator is the more appropriate testing tool, and complements the capabilities of the Digital Simulator.

3.1.3.2 Hybrid Simulator

The Hybrid Simulator combines analog and digital computers with various pieces of G&N hardware to provide a real-time simulation of the flight programs. By interfacing with the simulation through a DSKY and various hand controllers and switches, the user can control the flow of the program in process and can make on-line modifications if necessary. This capability is especially pertinent, since the Apollo system involves such a high degree of man/machine interaction. The user may be a designer testing a new design, a programmer verifying his coding, a human-factors engineer evaluating crew procedures, or an astronaut familiarizing himself with the system. Two complete simulators exist, one for the Command Module and one for the Lunar Module. Mockups of the CM and LM cockpits are interfaced with each of the hybrid computers to provide an environment for realistic replication of crew functions associated with the G&N system.

Analog and digital computers are both necessary to provide real-time simulations. Such high-frequency effects in the environment as bending and actuator dynamics are simulated by analog computers, since a digital computer cannot respond in real time with the accuracy needed. Repetitive mathematical and data-processing functions, however, are best performed by the digital computer.

In the Hybrid Simulator, actual Apollo LM and CM computers are used; however, Core Rope Simulators replace all of the AGC memory with erasable memories, thus facilitating conversion from one rope assembly to another. Core Rope Simulators also provide many useful features to aid in program analysis, such as the ability to monitor and change memory locations, and to stop and single-step either computer. Actual Coupling Data Units interface with the AGCs, but the remaining G&N hardware, as well as spacecraft dynamics and the external environment, are simulated. The cockpits feature planetarium displays and television for use by the optics equipment and for simulated lunar landing.

Operation of the Hybrid Simulator requires the participation of an AGC user and a computer operator. An XDS 9300 computer controls the simulation. It initializes, checks and modes the analog computers. It loads the Core Rope Simulator with an AGC program, sets up the values of variables, uplinks erasable-load values to the AGC, and turns the entire simulation on. At this point, the AGC user will call up on the DSKY the AGC program to be verified. This can be done either from the DSKY in the hybrid laboratory or from the one in the cockpit mockup.

During operation, data are taken from the AGC every two seconds in two ways: the cockpit displays, the DSKY and the Core Rope Simulator provide visual data displays; and the telemetry simulator transfers the AGC downlists directly to an XDS 9300 program which records each downlist, together with a selected "snapshot" of pertinent simulation parameters, onto magnetic tape. Following a simulation, the downlink tape is run through an Edit program to produce an arrayed, scaled and labeled line-printer output in a format convenient for comparing AGC and XDS 9300 quantities. Strip-chart recorders are used for recording simulated variables from the analog computers and also for some digital-computer variables after digital-to-analog conversion.

A disadvantage of the Hybrid Simulator is that the results are not exactly repeatable. The output of the analog computers can vary slightly with time, thus preventing a microscopic quantitative analysis and comparison of results. However, qualitative analysis and the checking of logical branches are facilitated by the fast turnaround time, and the ease with which AGC assemblies can be loaded into the Core Rope Simulator and changed as necessary.

3.1.3.3 Engineering Simulator

The Engineering Simulator was designed to aid in early analysis and Level 1 testing. The software logic specified in the GSOP was coded directly in the MAC language and run with a greatly simplified environment. The engineering simulation was also used to help evaluate AGC-coded performance on the All-Digital Simulator. As the Edit capabilities of the All-Digital Simulator were developed and improved, the Engineering Simulator became less important. However, the high operating speed and simple environments of the engineering simulations made them especially suited to statistical analysis of various techniques, such as rendezvous. The user could run many trials with changes in parameters, thus forming a large data base for statistical judgments. It would have been extremely costly and time-consuming to perform such runs on the All-Digital Simulator.

3.1.3.4 Systems Test Laboratory

The Systems Test Laboratory contains two complete G&N hardware systems —one each for the LM and CM. Although used principally to check out the hardware and hardware/software interfaces, the systems provide a software test and verification capability not present in any of the other simulators, since they include actual radars, optics and Inertial Measurement Units. This hardware complement allows the meticulous checking of radar and optics programs and further provides real hardware/software interfaces, with all of their inherent random characteristics. It is these characteristics that can never be duplicated on any simulator.

In the course of checking out the hardware/software, the operators have ofttimes uncovered bugs which otherwise would not have been discovered, since on a simulator all of the possible vagaries of an actual hardware/software union might not have been simulated.

Most problems which occur during flight can be readily explained, but it remains to be proven in the Systems Test Laboratory if that explanation is indeed correct. For example, during the Apollo 11 lunar descent several alarms came up on the DSKY indicating that the computer was saturating without apparent reason. A suspicion that the rendezvous-radar power switch was in the wrong position was confirmed via voicelink to the crew, thus erasing initial doubts about equipment failure. This explanation for the troubles encountered during lunar descent was later verified in the Systems Test Laboratory when the lunar-descent programs were run with the hardware in the incorrect switch configuration.

From a software-testing point of view, one disadvantage the Systems Test Laboratory has is that it makes no provision for spacecraft dynamics, but this is of little consequence since the Hybrid Simulator does. The Hybrid Simulator serves as the tool for the great bulk of those tests which require an astronaut/software/hardware interface. However, those programs which utilize interfaces with the optics and radar are tested in the Systems Test Laboratory. In a real sense, therefore, these facilities complement one another.

3.2 Software Sṕecification Control

The Guidance System Operations Plan (GSOP) is the NASA-approved specification document for each new rope. Before release for manufacture, the coding should fulfill all of the performance requirements and logic specified in the GSOP. Changes in this specification from one flight to the nextmust be approved by the NASA Software Control Board (SCB) in the form of a Program Change Request or Program Change Notice. There are, however, many points in the coding which are "below" the GSOP level of specification. Changes to coding not covered by the GSOP may be made without NASA approval, but require internal MIT review in the form of MIT Assembly Control Board (ACB) approval. After the rope is released for manufacture, an Anomaly form is used to report detected deviations from the specification.

The GSOP is by definition an incomplete specification, in that it does not accurately reflect such program factors as timing, flag setting, restarting, display of data, jobs and tasks, or erasable structure. Changes in coding below the specification level of the GSOP do not require NASA approval. Thus, there is a certain amount of freedom of implementation available to MIT. However, MIT performs internal change control by requiring Assembly Control Board approval of all changes not specifically covered by other documentation. ACB requests are used primarily to conserve coding and improve program efficiency.

Various meetings with NASA serve to define and control software implementation. The Software Development Plan Meeting is held regularly at MIT to review the status of the software effort and plan future development. Three meetings have been used to mark official NASA acceptance of a rope. (See Section 3.2.3.) The First Article Configuration Inspection (FACI) followed Level 4 testing, and was the preliminary approval to release a rope for manufacture. Upon completion of Level 5 testing, the Customer Acceptance Readiness Review (CARR) marked approval of the complete functioning of the rope. About one month before flight, the Flight Software Readiness Review (FSRR) approved the rope for the particular flight details and uses. Since Apollo 8, the FACI and CARR have not been used.

3.2.1 The Guidance System Operations Plan (GSOP)

As discussed briefly in Section 1.3.2, the Guidance System Operations Plan is the specification document for the software effort. It is published separately for the Lunar Module and the Command Module. The GSOP is updated with each new program release, thus providing NASA with ready and accurate control over the software and system operations. In addition to its role as a specification document, it has served as a working document within MIT to coordinate the inputs of the various groups, and as a testing foundation for simulator personnel. It has also served MSC personnel and contractors as a G&N description and as a crew-training aid.

The GSOP is published in six sections, each a separate volume. Section 1, Prelaunch, contains prelaunch calibration and test operations. Section 2, Data Links, describes programs and data for digital uplink and downlink between the onboard computer and the ground. Section 3, Digital Autopilots, describes the autopilot design and function. Section 4, Operational Modes, specifies the logic flow of the software coding for most programs and routines. (Since Section 4 does not specify the coding itself, programmers are relatively free to use the most convenient method of coding for a particular situation.) Section 5, Guidance Equations, is an engineering-oriented view of the guidance and navigation computations as used by the logic described in Section 4. Section 6, Control Data, is a summary of the data used in the All-Digital and Hybrid Simulators to verify the flight programs.

3.2.2 Change Control Procedures

All changes to program specifications must be submitted for NASA approval as either a Program Change Request or Notice. The Program Change Request (PCR) is a request for a change, originating either at NASA or MIT. It is given a preliminary review for technical content by the MIT program engineer and by the NASA Flight Software Branch, then held for Software Control Board action. Composed of representatives of various branches of NASA, the SCB may disapprove a change, order a more detailed evaluation from MIT, or order MIT to implement the change. This decision involves overall mission considerations and scheduling, as well as the particular software considerations.

Although a Program Change Notice (PCN) follows the same approval procedure as a PCR, it is a notification by MIT that a change is being made, rather than a request for a change. The PCN is used for clerical corrections to the GSOP, or for changes which must be made for program development to continue. The use of PCNs to authorize changes has some risk, in that formal SCB action may disapprove the PCN, requiring the undoing of the change.

An Anomaly is a failure of the program to perform to the specification. Anomaly reports result from testing and inspection after rope release. They may be originated by NASA, MIT, or the other contractors to report program irregularities or deviations in expected performance. The Anomaly form submitted to the Flight Software Branch contains a detailed description of the Anomaly, including its cause, how the Anomaly is recognized, its effect on the mission, avoidance procedures, recovery procedures, and suggested program corrections. Since Anomalies occur late in the preparation for a mission, after a rope has been manufactured, the disposition is usually to write a "program note" for the present mission, and correct the problem in a future release. Sometimes, however, as of result Anomalies, new PCRs, or problems discovered in testing, it is necessary to re-release a rope. When the decision is made to fix an Anomaly, authorization may be given in one of two ways. If the Anomaly has no effect on the GSOP, a routing slip is attached to the Anomaly with direction to fix the problem. If the Anomaly has GSOP impact, a PCR or PCN is prepared and processed in the normal manner. Approval by MSC of the PCR is the authorization to fix the GSOP and the program.

Program and Operational Notes are prepared by the NASA Flight Software Branch and reviewed by MIT personnel with the crews in attendance before each flight. The purpose of Program Notes is to advertise to the crew and flight controllers known subtleties and Anomalies in a rope, and to provide workaround procedures.

3.2.3 Software Control Meetings

Various meetings among NASA, MIT, and the other contractors serve to disseminate information about software status, to control changes in specification, and to mark formal acceptance of the released flight rope by NASA.

The Software Development Plan Meeting is held biweekly at MIT, with NASA represented by the Flight Software Branch. Reports are presented by MIT on the programs in development, and problems are discussed at the programming level. These are working meetings, long and detailed, where many policy decisions are made, and misunderstandings ironed out.

Periodically the Software Development Plan Meeting is expanded to include the Chairman of the Software Control Board, thus forming the Joint Development Plan Meeting. More formal presentations are included, and crucial decisions made.

Following each meeting, the Software Development Plan group issues a plan to organize and control schedules, personnel assignments, and other internal requirements. The plan presents the status of PCRs and Anomalies, and includes detailed milestones of program development, testing, verification and documentation.

In accepting a rope for a specific flight, NASA's original concept was to conduct three milestone meetings:

1. First Article Configuration Inspection (FACI)
2. Customer Acceptance Readiness Review (CARR)
3. Flight Software Readiness Review (FSRR).

The FACI was to culminate MIT's testing program through Level 4 and to provide a "B" release which could be used for training purposes. In a joint MIT/MSC meeting, working groups would review the results obtained from Levels 3 and 4 testing to ascertain whether the rope was ready to undergo Configuration Control. The review at the FACI was directed towards assuring that the program reflected the GSOP specifications, and that the testing program was sufficient and proper. The FACI would approve manufacture of a "B" rope, and authorize MIT to conduct formal Level 5 tests on the rope, using a NASA-approved "Qualification Test Plan".

The CARR was conducted to review the results of Level 5 testing and authorize the manufacture of an "A" release to be used on the mission. All aspects of the program were to be approved, not only those expected for the forthcoming mission.

Following the CARR, MIT conducted Level 6 testing, oriented toward the particular flight. NASA and other contractors also tested the rope, using the expected data and trajectories. Anomalies were reported and documented. The FSRR was then conducted four to six weeks prior to launch, to review program performance under actual mission requirements. This was done to determine whether additional testing or workaround procedures were necessary, and to formally accept the rope for use on the flight.

In actuality, no rope has been accepted according to this plan. The "B" release which follows Level 4 testing has been flown in every mission since Apollo 8, and has often been manufactured prior to FACI. The FACI and CARR Meetings have fallen into disuse, since Software Development Plan Meetings and telephone conferences have provided NASA with a more efficient working format for information and control of rope development. The FSRR is left as the only official meeting for the analysis and acceptance of a rope.

3.3 Documentation Generation and Review

MIT software documentation is necessary for specification control as well as mission support, general communication and training purposes.

The GSOP (described above in Section 3.2.1) is the NASA-approved specification document for each rope, but also provides general information about the software system. Sections 3 and 5 of the GSOP include much of the engineering analysis underlying the control systems and guidance equations. The logic flow of the programs given in Section 4 provides a convenient format for individuals to develop an operational understanding of the guidance and navigation functions on the spacecraft, without having to delve into the actual computer coding. Early versions of Section 4 included the crew-abbreviated and expanded G&N checklists, linking the operational details with the software logic. The checklist format was a DSKY display/crew response sequence. It included pertinent options, and those systems operations which interfaced with the G&N. Later, to expedite document reproduction, the checklist was separated from the GSOP and included in the Functional Description document. It was integrated by MSC into the complete onboard checklist, with format and content essentially unchanged.

The Functional Description document was created to provide an operationally-oriented description of interfaces between the G&N hardware and software, and between the G&N and the backup systems. This document also served in the training and familiarization of crew and crew-support personnel. It was the first MIT document to include a detailed description of all G&N hardware, as well as telemetry outputs and complete backup and malfunction-detection procedures. It detailed those steps the crew would perform to determine where a failure had occurred if one or more symptoms of subsystem malfunction appeared. These procedures were presented in flowchart format, and have been incorporated into the contingency checklist section of the onboard flight-crew data file under the direction of the Astronaut Office and Flight Crew Support Division of MSC. The Functional Description document was updated with each mission. Outside critique by other subcontractors through MSC helped MIT to maintain a high degree of accuracy. The document was last updated for Apollo 12 and has been discontinued, since hardware design and operations have stabilized. Many of the software aspects of the Functional Description document will be fulfilled by a Users' Guide, described below.

The computer listing of the AGC rope also serves a documentation role. This listing is a printout, line by line, of each instruction and location in the rope. However, "remarks" have been liberally added to the listing to aid the user in following and understanding the various programs. A program may include a general description, a list of calling programs, explanations of various branches, and other aids to understanding the logic flow, depending on the individual programmer. There is also a general section of remarks, including lists of verbs, nouns, alarm codes and flagwords. A symbol table provides a cross-reference for symbols used in the various programs and gives their definitions and uses.

A document flowcharting the computer programs has evolved from a series of blue-line charts to the present, bound, mission-specific volumes. These flowcharts are distinct from those of GSOP Section 4, in that they follow in detail how the AGC coding has been implemented. The flowcharts are produced by a documentation group separate from the programmers, which not only makes for standardization, but can serve as an independent check on the validity of the coding. Whenever possible, the flowchart is keyed to the equations of GSOP Section 5. Comments are freely used to clarify a program's function and to define for the benefit of the reader such terms as variables, units and scale factors. Thus, the flowcharts can replace the computer listing as a reference source for many purposes, and can provide a commentary and guide for those cases where the listing must be consulted as the primary source.

The above documents, as well as the Apollo Operations Handbook (published by NASA), have been used for crew training purposes. However, they have generally appeared to be too detailed and inclusive for easy assimilation of information by flight crews. For this reason, the Users' Guide to Apollo GN&CS Major Modes and Routines is being written. The Users' Guide presents the basic operation of the onboard system for use by crew members and flight controllers who have no prior G&N experience. The objective is to comprise all programs, routines, and extended verbs defined by the GSOP, describing their operation, theory and interrelationships. in sufficient detail for a crew member to gain the prerequisite understanding on which to base a more rigorous study of specific, flight-particular details and procedures. The Users' Guide is not mission oriented, but is updated periodically to reflect major software changes.

In addition to maintaining the above documents, MIT reviews various NASA publications. The Apollo Operations Handbook (AOH) Volume 2, for the CM and for the LM, is reviewed for accuracy and conformity with each successive onboard

Program. All current operational Anomalies and program notes are incorporated. Changes to the AOH are submitted on a Proposed Operational Procedures Change (POPC) form to the Apollo Program Control Office and indicate the recommended wording for each desired change. MIT also reviews POPCs submitted by other contractors. The AOH is kept up to date, and the final version is released one month before launch. This document, however, is not designed to be mission oriented; its primary function is to specify the physical characteristics of a given spacecraft —the spacecraft's role during a given mission is treated only peripherally.

Flight Plans and Mission Rules documents are reviewed upon receipt by the current MIT Mission Program Engineer. In addition, the following documents (in preliminary and final editions), issued by the MSC Data Priority Coordination group, have been reviewed by a large number of MIT design and flight-support personnel.

a. Abort Summary Document
b. CSM Rendezvous Procedures Document
c. LM Rendezvous Procedures Document
d. Reentry Procedures Document
e. LM Descent/Phasing Summary Document
f. Lunar Surface Operations Document

These volumes have been reviewed and commented upon within a three-week response period. Communication is mainly in the form of informal comments submitted to MSC Data Priority personnel through the MIT Mission Program Engineer responsible for that flight and vehicle covered in the particular document.

3.4 Mission Support

MIT's tasks in mission support are varied. Crews, flight controllers, and others are given formal and informal briefings, as well as simulator training. MIT personnel are assigned to NASA for flight support; and, via telephone link from Cambridge, MIT plays an important role in real-time support during missions.

3.4.1 Crew Support

A series of flight-crew classroom briefings on the G&N system were developed by MIT personnel to meet several objectives. These briefings sought to define fundamental problems in guidance and navigation, and to show the solutions as mechanized in Apollo. They described the G&N system capabilities and limitations, with emphasis on the reasons for particular programming, and they introduced the Apollo flight crews to detailed G&N procedures, operational control, software moding, and onboard program logic. Flight crews of every mission from AS-204 through Apollo 14, as well as NASA ground flight controllers and representatives of other NASA subcontractors, have been briefed by MIT in these classroom sessions. The training sessions have evolvéd into a format emphasizing system mechanization, rather than fundamental problems behind the techniques. This change in emphasis was a natural result of greater crew sophistication in understanding the nature of the G&N system. Time limitations also forced strict adherence to matters of immediate mission success.

The great majority of presentations were prepared and presented by the engineers who designed, built and analyzed the G&N system. Other presentations were prepared by simulation verification personnel and the respective Mission Program Engineers. Direct contact between the flight crews and MIT personnel benefited both parties and added a depth of appreciation for each other's problems and goals.

Each training session related to a particular mission and onboard program. Although particularly beneficial from the crew's point of view, this policy placed a sizable burden on MIT engineers at those times when crew briefing conflicted with program release. A possible alternative would have been to have two or three persons devoting their energies to understanding the entire G&N system, solely for crew-training purposes. However, the extremely rapid change and development of onboard programs made this a virtually impossible task.

In addition to the formal classroom briefings, there were periodic special crew briefings with MSC personnel to resolve issues of primary importance. MIT also monitored crew training on the Command Module and Lunar Module Simulators at Kennedy Space Center, and helped in troubleshooting possible system or simulation Anomalies. MIT personnel were thus available to explain G&N operations to flight crews and simulation personnel. These less formal approaches are considered to have been as essential to efficient crew use of the G&N system as the more formal classroom sessions.

To expedite crew procedures, MIT investigated short sequences of crew interactions with parts of the G&N system. These "part-task" evaluations sought to determine the limits of a human's ability to perform various G&N tasks, identifying those environmental factors most significant to performance.

A prime tool for these studies was the MIT Sextant Simulator. This device duplicated full optical motion of the sextant and provided optical images of landmarks, horizons and stars. It was used to verify marking accuracy during navigation-sighting tasks performed under a variety of environmental constraints. These tasks included star/landmark, star/horizon, star/reticle, flashing LM beacon, and simulated Apollo Optical Telescope star sightings. Tests were also performed on KC-135 zero-g flights to verify marking accuracies, and to determine the necessity for tethering the crew members during task performance. The CSM and LM cockpit mockups of the Hybrid Simulator also were used to evaluate crew tasks, such as attitude maneuvers, landing-point redesignations, and IMU-alignment sightings, as well as end-to-end flight sequences.

3.4.2 Flight Support

MIT's role in flight support has undergone considerable change over the course of the program. Early, during the development of the GN&C System, MIT was asked to support the Flight Control Division by sending personnel to be trained as flight controllers. In response to this request MIT assigned several people to the Manned Spacecraft Center in Houston. As it was, those MIT personnel who supported the Flight Control Division at MSC soon lost touch with the new developments at the Laboratory during those early days of rapid change of both hardware and software.

Before any missions were flown, MIT's real-time support underwent a change. As a result, the first four Apollo flights, all of which were unmanned and of less than one day's duration, were supported by the software specialist (dubbed "rope mother") responsible for the development of the onboard computer program and by a representative of the hardware division.

With Apollo 7, the first manned flight, two significant developments occurred. First, the program had become too large and complex for one, seemingly omniscent rope mother to oversee, thus requiring the overall responsibilities to be delegated to a large number of persons (see Section 2.2.3), one of whom was designated Mission Program Engineer. His responsibility included monitoring of the flight programs from the end of Level 6 testing through the real-time mission support. He represented MIT at Mission Control Center, Houston, and participated in any real-time decision making. Second, MIT was asked to provide software specialists to support the Flight Software Branch. These individuals were assigned to Houston on a 6- to 12-month basis and reported directly to the Flight Software Branch.

From Apollo 7 on, MIT has had the availability of a console in the Flight Dynamics Staff Support Room at MSC; and since Apollo 10, this has become a permanemt assignment. The MIT console is concerned not only with software aspects, but with the operation of the GN&C System as a whole; thus, it complements the adjacent Flight Software Branch console.

Since mission support is generally accomplished on a person-to-person basis, it has been advantageous to use a constant small group of people to represent MIT during the missions. Thus, the flight controllers develop confidence in the capability of particular individuals to respond to any mission-critical situation.

During each flight since AS-202, MIT in Cambridge has maintained direct contact with Mission Control Center, Houston through a Scheduling, Conferencing and Monitoring Arrangement (SCAMA). This consists of three dedicated telephone lines, one for two way phone conversations, one for "listen only" air-to-ground communications between the spacecraft crew and mission control, and the last for "receive only" teletype transmissions of Guidance, Navigation and Control parameters stripped from raw telemetry data.

Beginning with Apollo 7, SCAMA facilities were moved into a large room, a digital clock was added to keep track of ground elapsed time, and an input to the XDS 9300 computer was added in parallel with the teletype. This last addition allows a computer-editing process to take place on the telemetry information. Teletype messages and edited data are stored on magnetic tape for recall if required. The edited data are also printed for immediate verification by G&N specialists.

The minimum manpower required for flight support at MIT, Cambridge is three persons per shift, three shifts per day for round-the-clock coverage throughout the mission. These three people are a communicator, a software specialist and a hardware specialist. It is the communicator's responsibility to coordinate SCAMA phone conversations, to maintain a chronological events log and an action-item file, and to call in appropriate experts as required. The software specialist is cognizant of the entire program code and is expert in a particular section of coding current in the flight program. The hardware specialist is cognizant of all operational aspects of the G&N hardware and investigates any variances observed in the telemetry data.

Available for use as troubleshooting tools, procedural verifiers or mission phase predictors are two operational G&N Systems in the Systems Test Laboratory (one for each vehicle) and two Hybrid Simulators (one for each vehicle). These are all loaded with the appropriate flight programs prior to lift-off and maintained in operational readiness throughout the mission.

APPENDIX A

MAJOR PROGRAM CAPABILITIES—
Coasting-Flight Navigation

The navigation function of the Apollo spacecraft GN&CS is conducted during all phases of the Apollo lunar mission. As mentioned in Section 2.2, the mission phases for the spacecraft GN&CS are:

1. Launch to earth-orbit monitor
2. Earth-orbit navigation monitor
3. Translunar-injection maneuver monitor
4. Earth-moon (translunar) midcourse navigation and guidance
5. Lunar-orbit insertion maneuver
6. Lunar-orbit landing-site sightings
7. Descent-orbit injection maneuver
8. Lunar-landing maneuver
9. Lunar-ascent maneuver
10. Lunar-orbit rendezvous navigation and control
11. Transearth-injection maneuver
12. Moon-earth (transearth) midcourse navigation and guidance
13. Earth-reentry and landing

The navigation function during many of these mission phases is pure inertial navigation using the IMU and the computer. Typical maneuver phases of this type are the translunar injection, lunar-orbit insertion, lunar ascent, transearth injection, and earth reentry. These mission phases are characterized by large acceleration forces due to the spacecraft engines or atmospheric entry.

During all free-fall or coasting phases of the Apollo mission—cislunar, orbital and rendezvous—the onboard system employs the same navigation concept, a recursive formulation of the optimum linear estimator originally devised by R.E. Kalman. This concept incorporates measurement data sequentially without recourse to the batch-processing techniques common to other methods. Matrix inversion is avoided by regarding all measurement data as single-dimensional or scalar, with the measurement characterized by a geometry vector. These features allow a navigation formulation compatible with the complexity and computational limitations of the onboard computer. A further important feature of this concept is that, within the framework of a single computational algorithm, estimates of quantities such as rendezvous-radar biases (in addition to position and velocity) can be included by the simple expedient of increasing the dimension of the state vector. This appendix has been restricted to these three mission phases which utilize recursive navigation techniques.

A.1 Cislunar Navigation

The cislunar phases of the Apollo mission are the translunar trajectory between earth orbit and the moon, and the transearth trajectory from lunar orbit to the reentry into the earth's atmosphere. These two cislunar trajectories are illustrated in Fig. A.1-1, along with typical navigation sighting periods for the Command Module GN&CS. The primary mode of navigation for the Apollo cislunar phases is the Manned Space Flight Network (MSFN), a system of earth-based tracking stations. Within this system, ground-based radar-tracking data are processed in the Real Time Computation Center of the NASA Manned Spacecraft Center to determine the spacecraft state vector, and to compute required midcourse correction maneuvers. These are telemetered to the spacecraft for targeting the translunar and transearth trajectories to their desired terminal conditions. The onboard spacecraft GN&C System acts in a backup navigation capacity during these two phases. During the cislunar phases the GN&CS provides the self-contained capability to determine the spacecraft's state vector, using onboard measurements, so that the spacecraft can establish and target a safe-return trajectory to the earth if communications from the earth were lost. The sighting schedule illustrated in Fig. A.1-1 on the outbound translunar trajectory is the schedule used to checkout and calibrate the spacecraft navigation-sighting system under nominal conditions when the trajectory is being determined by the ground-tracking stations. Shown on the return transearth trajectory is a schedule which would be typical in the case where communication to the spacecraft were lost in the vicinity of the moon, necessitating that the system onboard the spacecraft navigate and control the return trajectory to earth. In this abort case, the objective of the spacecraft GN&CS is to determine and control the transearth trajectory such that the required earth-entry corridor conditions are achieved for

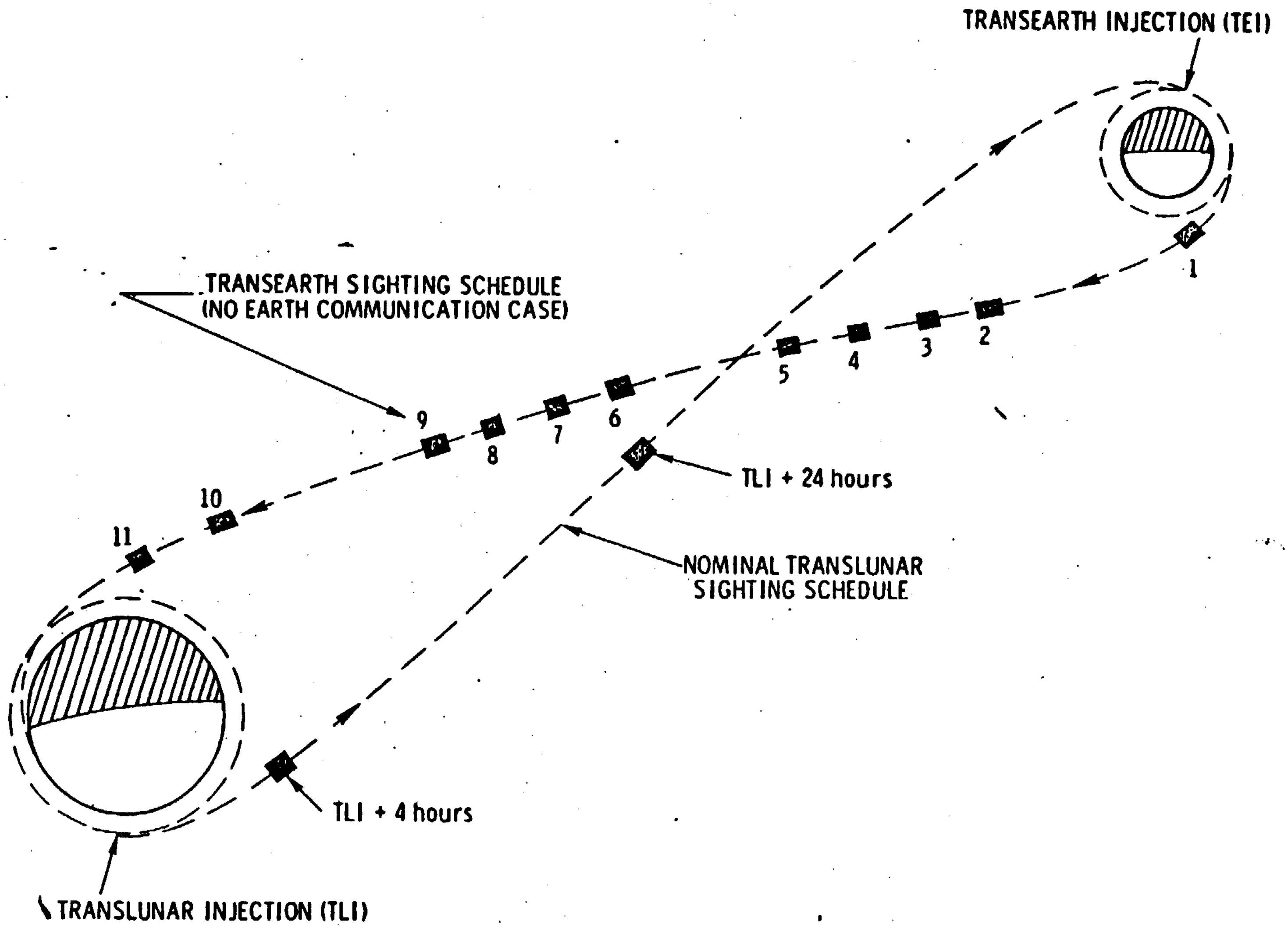

TRANSEARTH INJECTION (TEI)
TRANSEARTH SIGHTING SCHEDULE
(NO EARTH COMMUNICATION CASE)
1
2
3
4
5
6
7
8
9
10
11
TLI + 24 hours
NOMINAL TRANSLUNAR
SIGHTING SCHEDULE
TLI + 4 hours
TRANSLUNAR INJECTION (TLI)

Figure A.1-1 Apollo Cislunar Navigation Phases

a safe return. The navigation measurement used to achieve this objective is an optical star/horizon or star/landmark measurement made with the CM sextant.

For a cislunar navigation sighting, either the astronaut or the midcourse navigation program in the AGC points the sextant's two lines-of-sight at a specified reference star and landmark or horizon. It is then the navigator's task to center the superimposed star-image onto the landmark, if landmarks are being used, or onto the substellar point of the horizon, if the horizon target is being used. A sextant view of a typical star/horizon measurement is illustrated in Fig. A.1-2 at the moment when the navigator signals the computer to record the sextant trunnion angle and time of mark. This illustration is typical of the star/horizon view in the sextant during the first sighting interval shown on the translunar phase in Fig. A.1-1 when the spacecraft is approximately 30,000 nmi from the earth. In the Apollo lunar missions to date, the sunlit horizon of the earth has provided a more consistent and useful target for cislunar navigation sightings than landmarks, due to cloud cover and limited sunlit surfaces over major portions of cislunar trajectories. For navigation sightings using the moon, landmarks are preferred over horizons for their greater accuracy. Either the near or far substellar point of a horizon can be used in a star/horizon measurement, as shown in Fig. A.1-3. In this type of a measurement, it is important to superimpose the star-image on the sunlit horizon as close to the substellar point as possible and minimize the measurement plane misalignment error illustrated in Fig. A.1-3.

As previously stated, a single navigation concept is used in the Apollo spacecraft G&N systems for all coasting phases of the mission. A simplified functional diagram of the cislunar-navigation concept is shown in Fig. A.1-4. In this case, free-fall equations of motion extrapolate a six-dimensional state vector (position and velocity), along with the error-transition matrix, to the time at which a navigation measurement is to be made. After the reference star and planet landmark or horizon have been selected by the navigator, an estimate of the angle (A_{EST}) between this star and target is computed, based upon the extrapolated state vector, the reference star and the planet target. A measurement geometry vector ($\underline{b}$) is also determined, based upon the estimated vehicle state vector, reference star, planet target, and the type of measurement being made. For cislunar-navigation measurements, this geometry vector ($\underline{b}$) lies in the reference star/target planet plane and is normal to the planet line-of-sight, as illustrated in Fig. A.1-5. When the navig superimposes

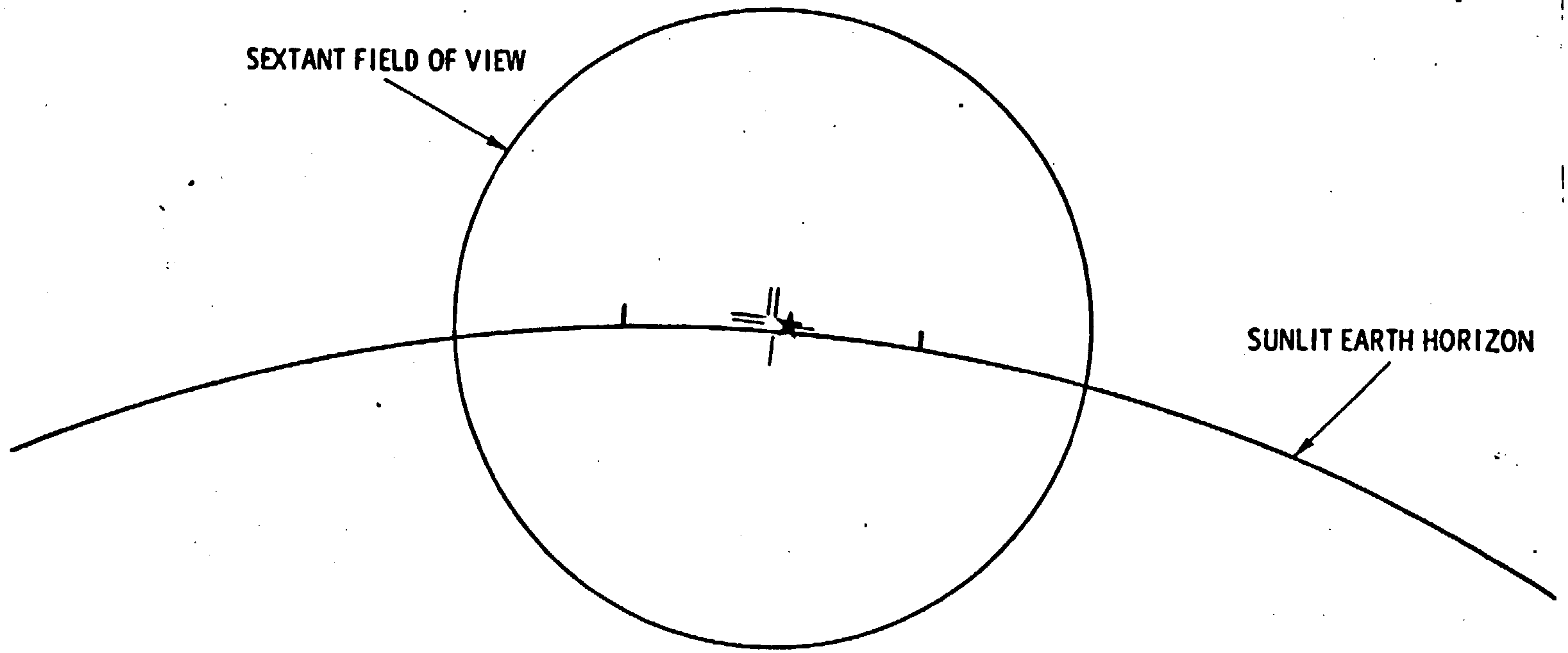


Figure A.1-2   Sextant View of a Typical Star/Horizon Measurement for First Cislunar Sighting Interval

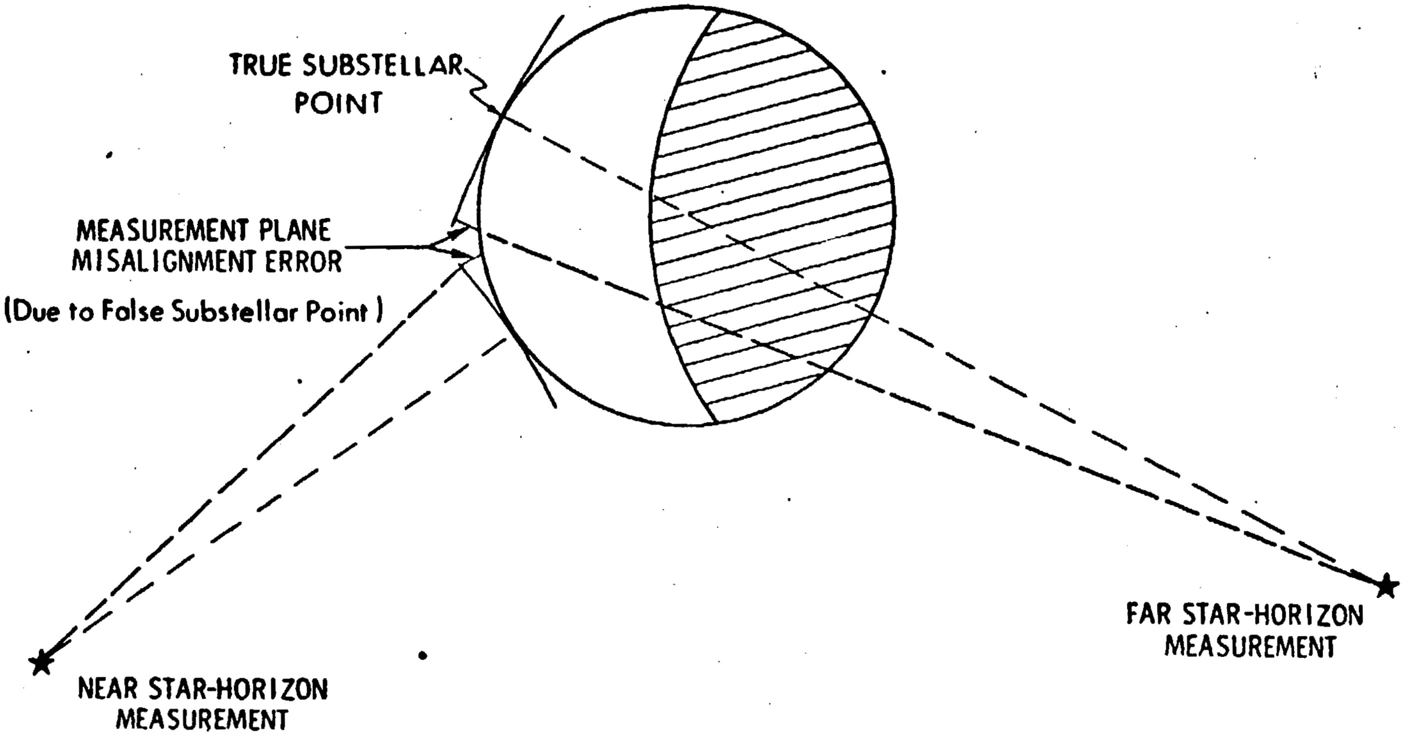


Figure A.1-3 Sextant Star/Horizon Measurements

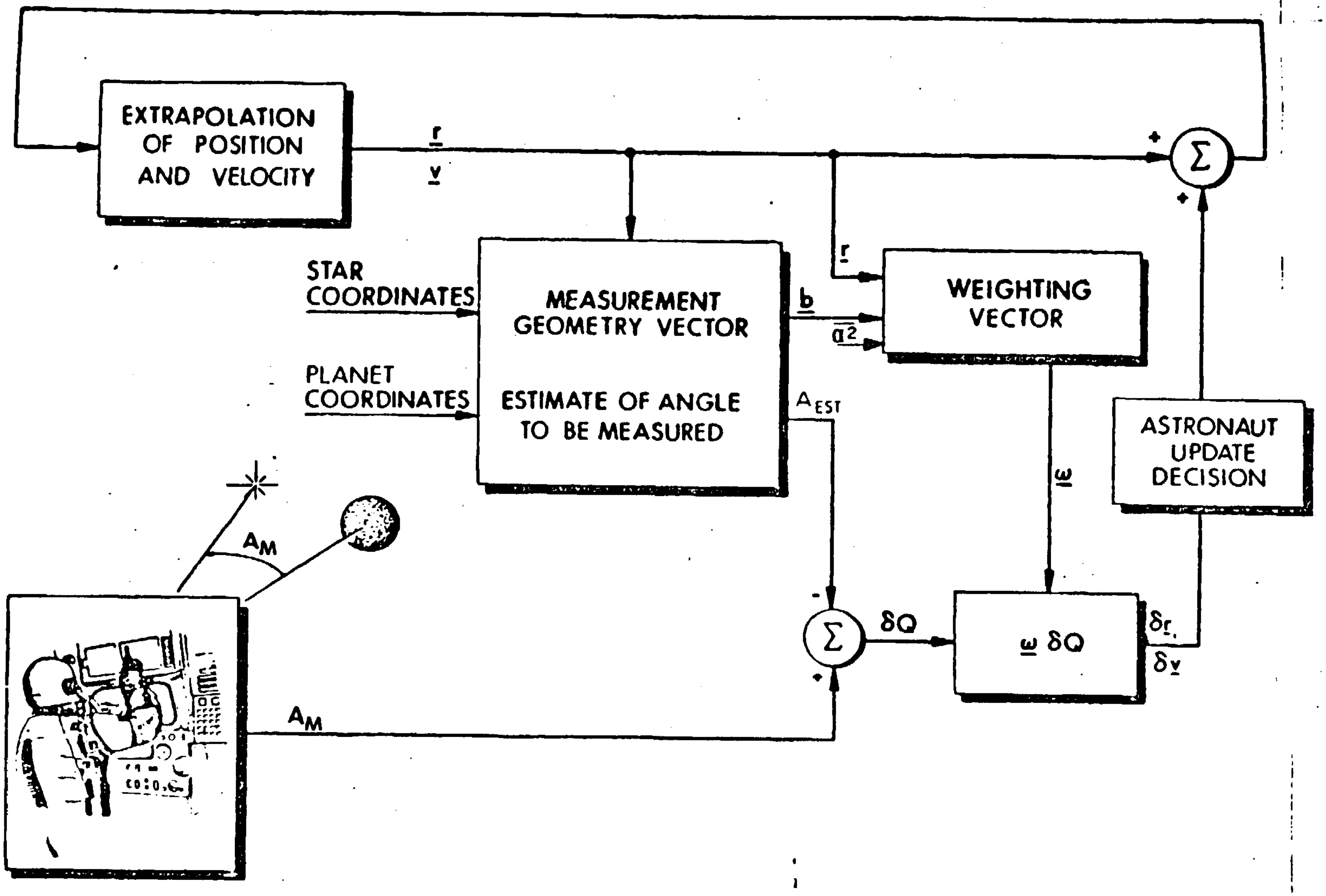

EXTRAPOLATION OF POSITION AND VELOCITY
r
v
+
Σ
+
STAR COORDINATES
MEASUREMENT GEOMETRY VECTOR
r
WEIGHTING VECTOR
b
a²
PLANET COORDINATES
ESTIMATE OF ANGLE TO BE MEASURED
A_EST
ω
ASTRONAUT UPDATE DECISION
A_M
-
Σ
+
δQ
ω δQ
δr
δv
A_M

Figure A.1-4 Simplified Functional Diagram of Cislunar Navigation

the images of the reference star on the planet target in the sextant field-of-view, the computer compares the measured angle, A_M, with the estimated angle, A_{EST}. With reference to Fig. A.1-4, the following are algebraically combined to form a weighting vector, $\underline{\omega}$: the measurement geometry vector, $\underline{b}$, the error-transition matrix, W, which is extrapolated to the measurement time (and updated, to take into account incorporation of a measurement for use with subsequent measurements), and the a priori mean-squared measurement error, $\overline{\alpha^2}$. A statistically optimum state-vector update, ($\delta\underline{r}$, $\delta\underline{v}$), is then computed from the difference in the estimated and measurement angles, δQ, and the weighting vector, $\underline{\omega}$. The geometry vector, $\underline{b}$, used to determine the weighting vector, $\underline{\omega}$, represents to a first-order approximation the variation in the measured quantity (A_M in this case) resulting in variations in the components of the state vector. This concept is illustrated in simplified form in Fig. A.1-5, depicting a single cislunar star/horizon navigation measurement and position update. The estimated position of the spacecraft is updated in this simplified example along the measurement geometry vector, $\underline{b}$, by an amount $\delta\underline{r}$, such that A_{EST} equals A_M. This example is simplified in two major respects: first, the magnitude of the update, δr, would be a function of the statistics of the sextant-measured errors and the extrapolated error-transition matrix, W, and would seldom make the measured and estimated angles exactly agree; second, the update shown in Fig. A.1-5 is entirely along the measurement geometry vector, $\underline{b}$, which might be valid for the first navigation measurement taken, but on subsequent measurements the weighting vector, $\underline{\omega}$, will rotate $\underline{b}$ by the correlation represented in W, such that the update $\delta\underline{r}$ will not be along the $\underline{b}$ vector. This correlation feature is central to the navigation concept. It should be recognized, however, that even though the cislunar star/horizon measurement directly updates the vehicle state vector in only one direction, $\underline{b}$; the other position and velocity components are also updated to a lesser, but still significant extent, through correlation. To achieve the greatest accuracy in cislunar navigation, sequential star/horizon measurements are ideally chosen so that the measurement planes of sequential sightings are separated by about 90 deg.

An important point to be noted in the cislunar-navigation functional diagram of Fig. A.1-4 is that, after the state-vector update has been computed by the AGC, this update is displayed to the navigator for his review, and he personally decides whether to accept or reject the update and navigation measurement. If the state-vector update computed from the first navigation sighting taken after several hours without navigation sightings exceeds a predetermined threshold, or if the update is fairly

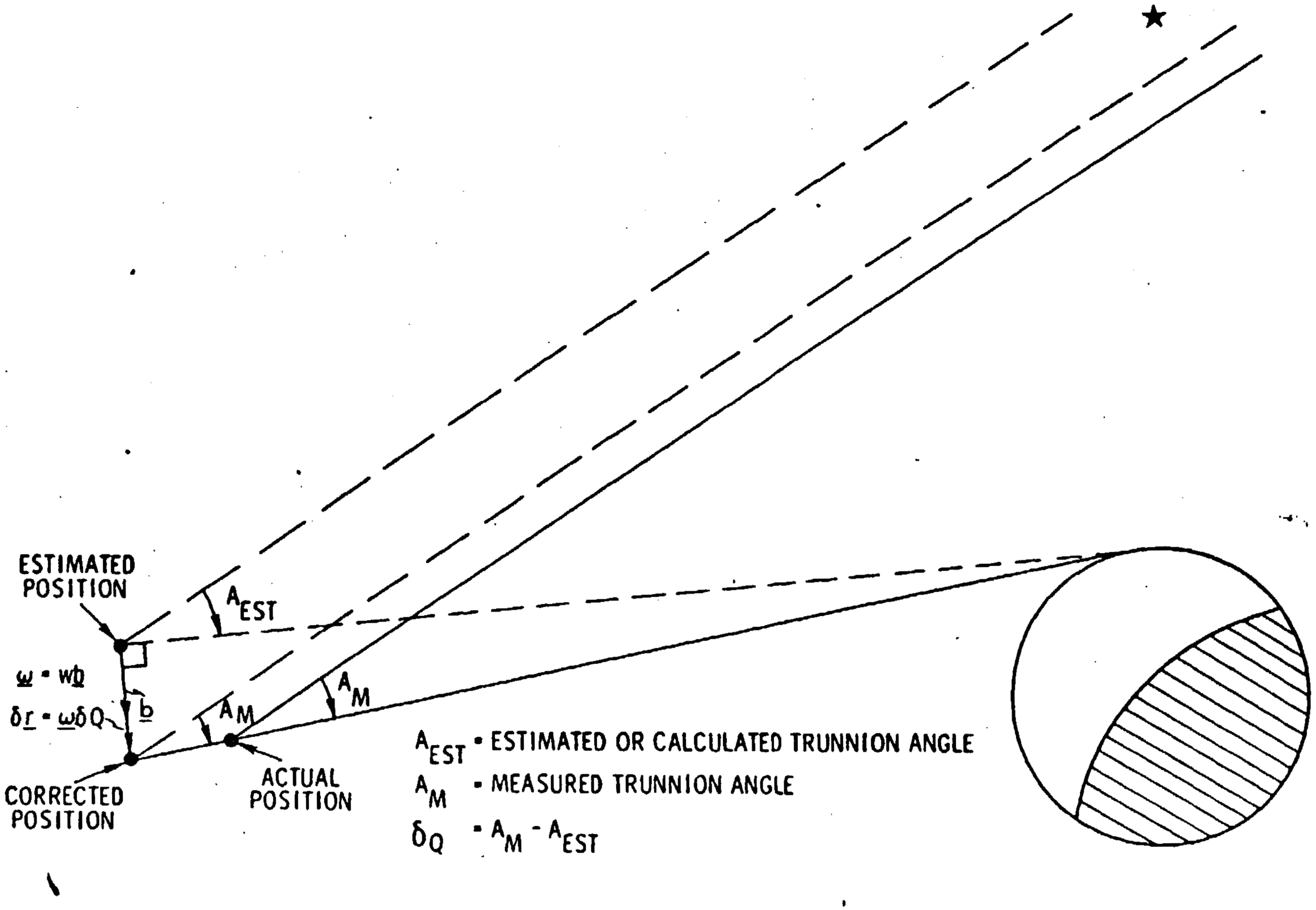

ESTIMATED POSITION
A_{EST}
$\underline{\omega} = w\underline{b}$
$\underline{b}$
$\delta \underline{r} = \underline{\omega}\delta Q$
A_M
A_M
CORRECTED POSITION
ACTUAL POSITION
A_{EST} = ESTIMATED OR CALCULATED TRUNNION ANGLE
A_M = MEASURED TRUNNION ANGLE
$\delta_Q = A_M - A_{EST}$

Figure A.1-5 Simplified Star/Horizon Navigation Updating

close to the threshold and the navigator is uncertain as to the sighting accuracy or identification of the target, he would reject the update. Upon rejection he repeats the navigation sighting. If the state-vector update is essentially identical to the previous (unincorporated) update, the navigator is logically obliged to accept the update and incorporate it into the state vector. Normally, after the first few navigation sightings and updates in a sighting period, all subsequent updates will fall below the preselected threshold and are routinely accepted.

Spacecraft cislunar-navigation accuracy is primarily limited by (a) unmeasured or unaccounted-for perturbing forces on the vehicle, (b) computational precision and computer-word length, and (c) optical measurement errors. In the Apollo GN&CS the optical measurement errors are the most serious. These measurement errors arise from:

a. Planet-lighting limitations

b. Sextant optical-design limitations

c. Horizon-phenomena uncertainties

d. Astronaut-sighting inability to determine the substellar point on the horizon, and to superimpose the star/horizon images during the presence of spacecraft attitude motion.

In the initial prototype Apollo spacecraft-sextant design a blue-sensitive photometer was included for horizon detection, but this was subsequently removed from the production systems since it had been decided that earth-based radar tracking would be the primary source of cislunar navigation. Without the photometer the navigator must select an altitude point (horizon locator) that can be consistently repeated from one navigation sighting to the next. It is believed, based upon simulation and flight experience, that the higher altitudes of the sunlit horizon provide the most consistent reference for navigation sightings where atmospheric phenomena are less likely to cause perceptual uncertainties. This reference altitude is approximately 32 km above the earth. Figure A.1-6 is a further illustration of how atmospheric weather conditions, such as clouds, can change the apparent horizon altitude in the lower atmosphere, and why a higher altitude reference was chosen. Each Apollo navigator must choose his own particular horizon altitude and try to maintain this reference throughout the cislunar phases. From post-flight analysis data of five Apollo lunar missions, this reference altitude has varied between 17 km

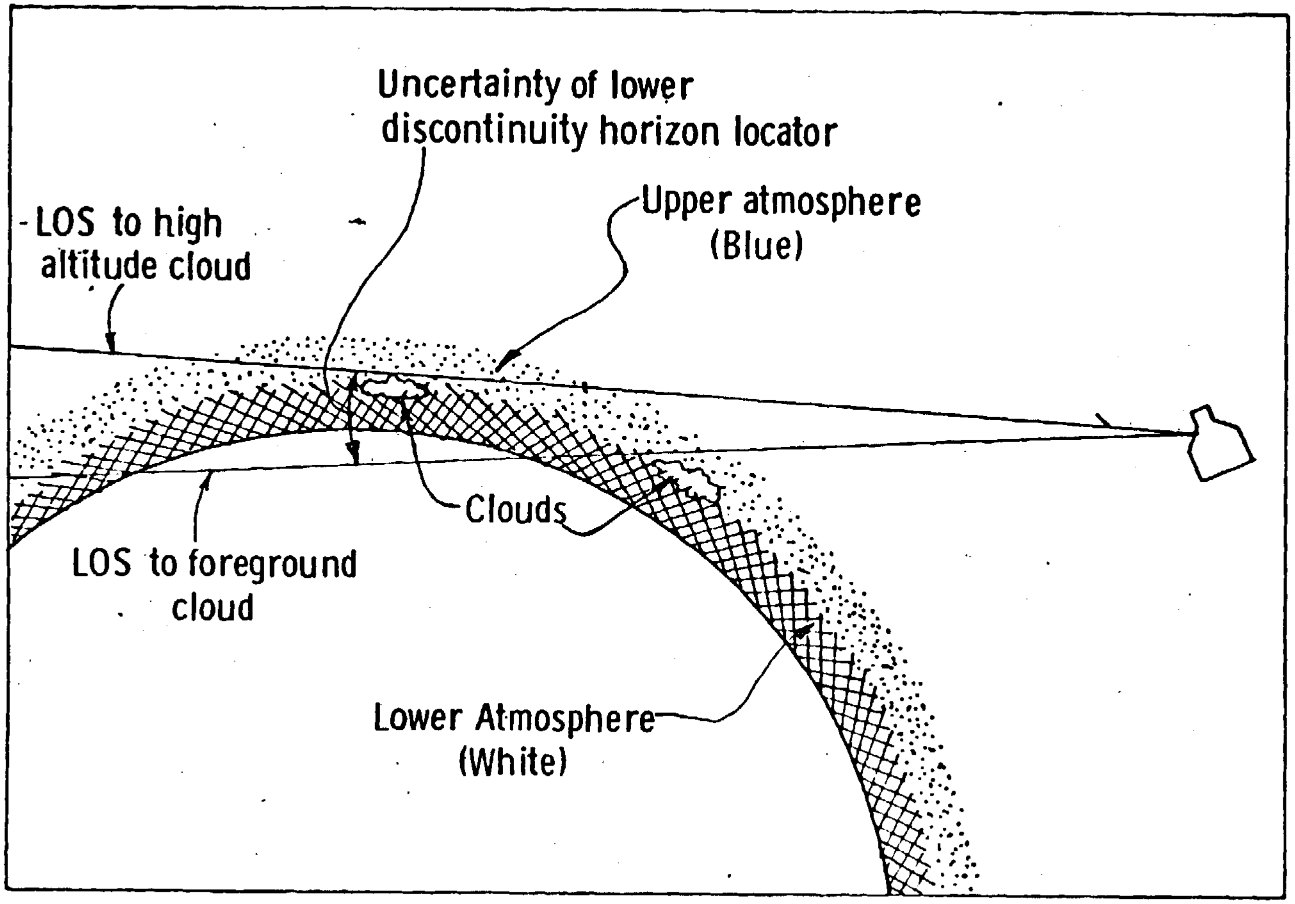
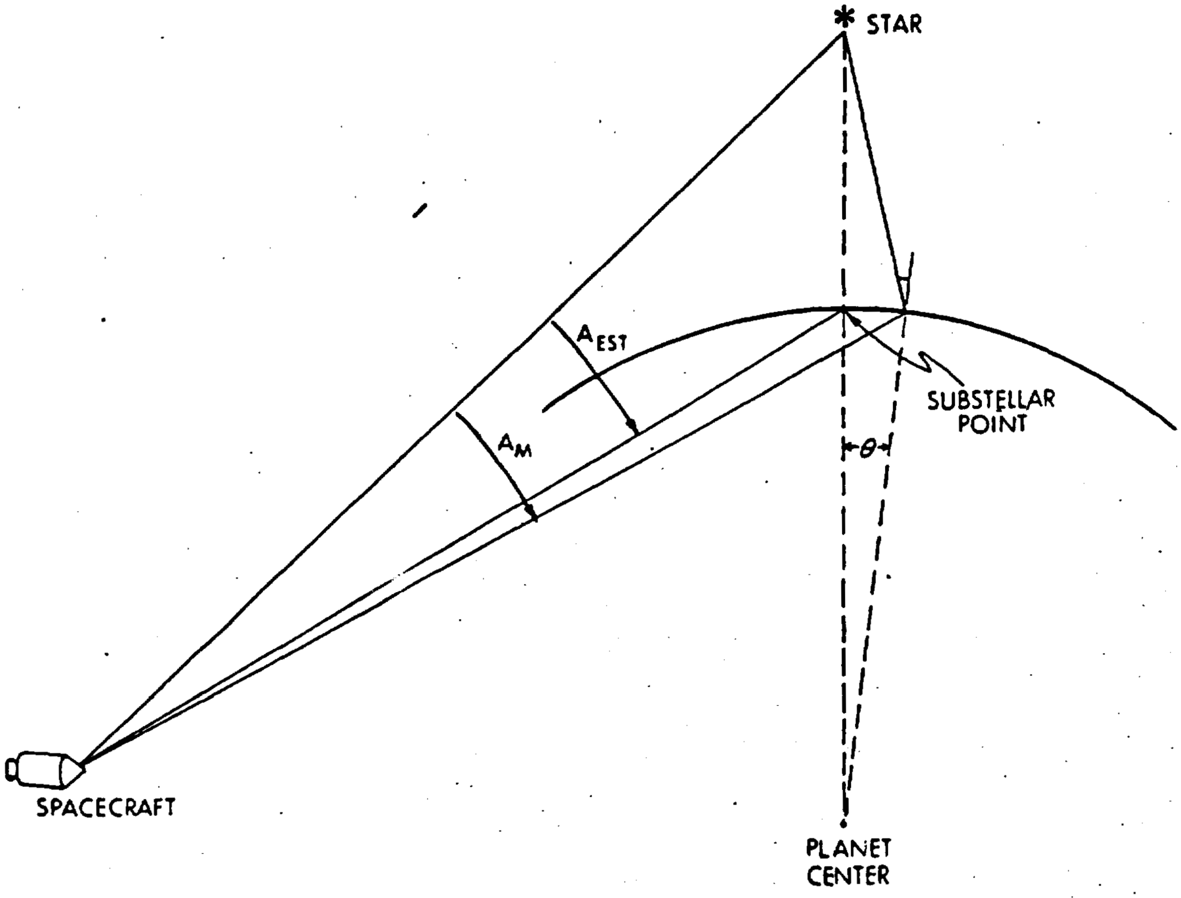


Figure A.1-6 Illustration of Cloud-Top Problem with the Use of Apparent Horizon as a Locator



θ *MEASUREMENT PLANE MISALIGNMENT ANGLE*

A_M *MEASURED TRUNNION ANGLE*

A_C *ESTIMATED TRUNNION ANGLE*

Figure A.1-7 Measurement-Plane Misalignment

to 44 km, but the consistency of an individual navigator once he has chosen this altitude reference is the most important feature with respect to navigation accuracy rather than the absolute value of this reference altitude. Since it is difficult to determine the absolute-reference altitude a given navigator will use before a mission, the first translunar-sighting period shown in Fig. A.1-1 at TLI+4 hours is used to calibrate the sextant and determine the reference altitude of the sunlit horizon the navigator will use for that flight by letting him sight on the horizon and then check this initial sighting against the predicted sighting angle using telemetered angle data. These initial horizon sightings along with other translunar sightings, are used to update the stored reference altitude in the guidance computer. Navigation sightings using the lunar horizons are naturally not complicated by the atmospheric effects encountered for the earth horizon. As a result these are more straightforward —the biggest problem being the roughness of the lunar terrain itself.

The third factor previously listed that affects navigation accuracy is the astronaut's ability to correctly superimpose the reference star on the horizon at the substellar point. This point is contained in the measurement plane defined by the spacecraft, star, and center of the planet at the point of tangency of the line-of-sight from the spacecraft to the horizon. Measurement plane misalignment is illustrated in Fig. A.1-7. In general, the star is not placed at the substellar point, but slightly to one side or the other, due to the dynamic nature of the measurement, since small attitude changes continually take place, or due to insufficient range and resulting curvature of the horizon resulting in a perceptual limitation to the accurate determination of the substellar point. This type of measurement error causes the sextant trunnion angle to be too large for a near-horizon (Fig. A.1-3) and too small for a far-horizon measurement.

The Command Module G&N system for cislunar navigation is basically a computer-aided manual operation. The navigator must initiate the navigation program, calibrate the optics, select the desired star and planet horizon, make the sightings, and finally accept or reject the resulting state-vector update computed by the AGC. In essence, the navigator is system manager, mission-sequence controller, subsystem-interface coordinator, and performer of specialized tasks too difficult or costly to automate. The AGC performs the basic navigation computations using the manually-controlled optical sighting data. Manual control was deemed desirable for the Apollo cislunar-navigation sighting operation to minimize the number of active GN&C units, thereby conserving power (since only the computer and sextant are needed). Recall that the GN&C System acts in a backup navigation capacity for safe earth return during those cislunar phases when the navigator has ample time to conduct navigation sightings. Operational experience during the lunar missions indicates that attitude maneuvering of the vehicle to a landmark or horizon while maintaining star acquisition manually for accurate sightings is a difficult task. For this and other reasons, such as the desire for passive thermal control of the vehicle in an automatic mode, it was decided to keep the GN&C System IMU powered during cislunar flight. The IMU availability affords additional assistance to the navigator by providing automatic control of the optics to the selected reference star and automatic control of the attitude maneuver to the computed substellar point. Furthermore, automatic vehicle-attitude hold during star acquisition and star-acquisition maintenance during the maneuver to the horizon also are particularly helpful under light-vehicle conditions during the transearth phase. With these aids, the astronaut's navigation-sighting task is effectively eased, and his major task becomes one of fine correction of the vehicle attitude—performing the delicate task of superimposing the star/horizon images and marking when superposition is achieved.

In the analysis of the cislunar-navigation data, it is felt that, even though the computer-aided manual-sighting performance is adequate for the Apollo missions, further accuracy can be achieved by making the sighting operation more automatic with the implementation of a horizon photometer designed to utilize two spectral regions of the sunlit horizon. The role of the navigator would still be important in handling other unforeseen problems that might arise during the missions, such as scattered-light conditions in the optics requiring alternate stars to be chosen for navigation, and reflections from debris and particles making star recognition difficult, if not impossible, under some conditions. The human navigator is well suited to handle problems of this type, while the GN&C System can be designed to relieve the navigator from the more routine but overburdening detail of the navigation operation.

A.2 Rendezvous Navigation

The nominal lunar-orbit rendezvous-trajectory profile for Apollo missions is illustrated in Fig. A.2-1. This profile is referred to as the concentric flight

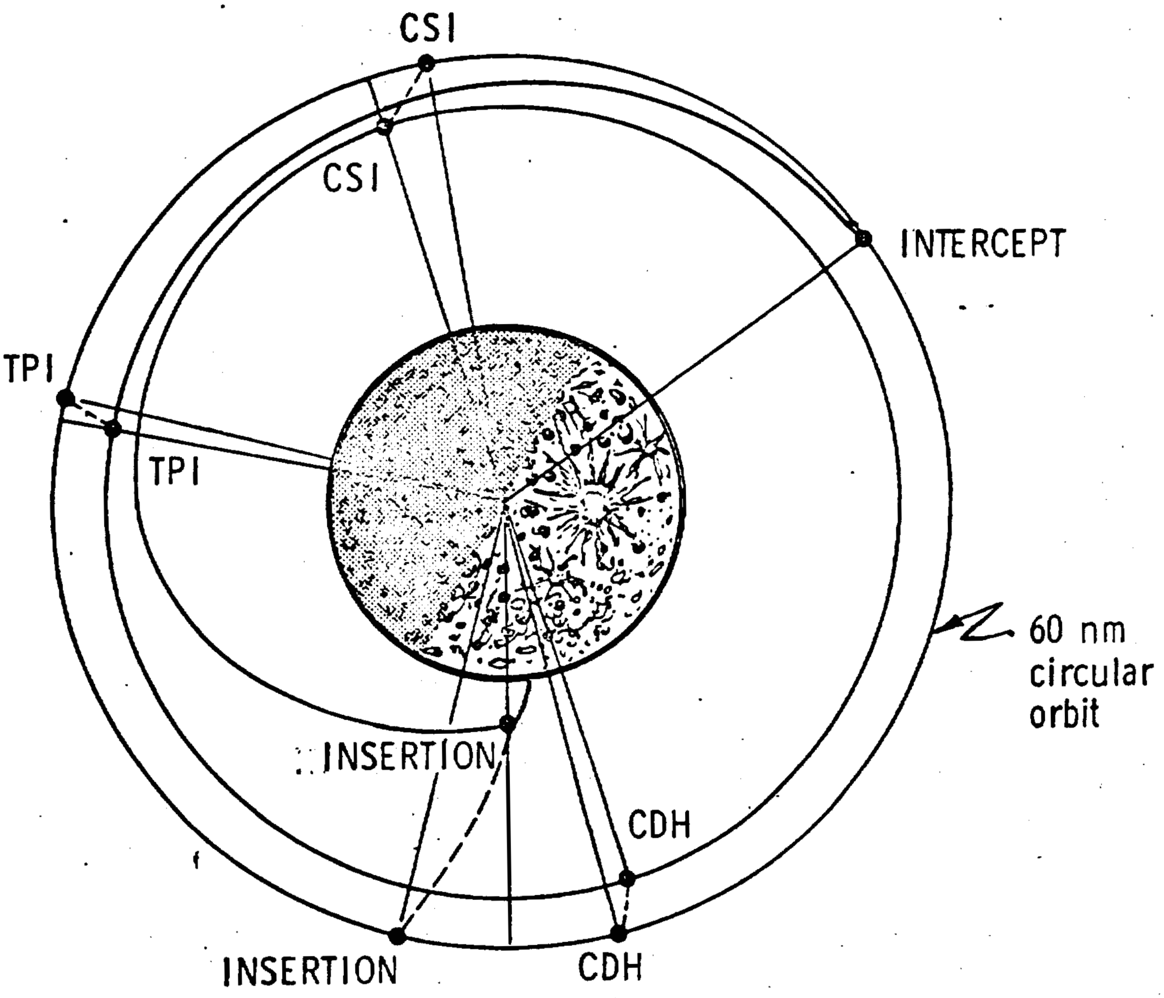


Figure A. 2-1   Nominal Apollo Rendezvous Profile plan and consists of two phasing-type maneuvers after lunar-ascent insertion (CSI-coelliptic sequence initiation; CDH-constant differential height maneuver) to place the active vehicle (LM) in a coelliptic orbit at an essentially constant altitude difference below the passive vehicle (CSM). After these conditions are established, the transfer-phase-initiation maneuver (TPI) places the LM on an intercept trajectory with the CSM. A series of midcourse correction maneuvers (MCCs) are normally made to improve or maintain this intercept trajectory so that the astronaut can manually perform the terminal-braking maneuvers before the intercept point. The objectives of the spacecraft rendezvous-navigation system are to maintain and update the estimated vehicle position and velocity vectors with relative tracking data so that the three major maneuvers and the midcourse corrections of the rendezvous profile can be correctly computed and executed, thereby minimizing propellant usage and achieving an accurate intercept trajectory with the CSM such that the manual terminal-rendezvous maneuvers can be efficiently performed.

In the rendezvous profile of Fig. A.2-1, both the active and passive vehicles conduct simultaneous rendezvous navigation; thus the CSM can provide maneuver information to the LM for backup purposes or execute retrieval maneuvers, if required. The LM rendezvous-tracking sensor is an amplitude-comparison, monopulse tracking radar which tracks a transponder on the Command Module. This radar provides range, range rate, and the two antenna-tracking angles (specified shaft and trunnion) as measurement data to the onboard rendezvous-navigation program. This operation is automatic and requires only general monitoring by the astronauts in the LM. The Command Module astronaut uses the optical sextant to manually track a flashing beacon, located below the LM rendezvous-radar antenna, or reflected sunlight from the LM to provide tracking data to the CM rendezvous program. This operation is similar to that used in cislunar navigation except that only the sextant-articulated star line-of-sight is used for rendezvous tracking, and the sextant tracking angles are referenced to a stable coordinate frame to which the inertial measurement unit is aligned. On the Apollo 7 and 9 missions, CM rendezvous navigation employed only optical-tracking data. On following missions a modification to the vehicle very-high-frequency (VHF) communication system provided relative-range information to the CM rendezvous-navigation program. After initially starting the VHF range system, these range data are processed automatically by the onboard Apollo CM guidance computer, while sextant optical tracking is still a manual task. Figure A.2-2 summarizes the tracking measurement

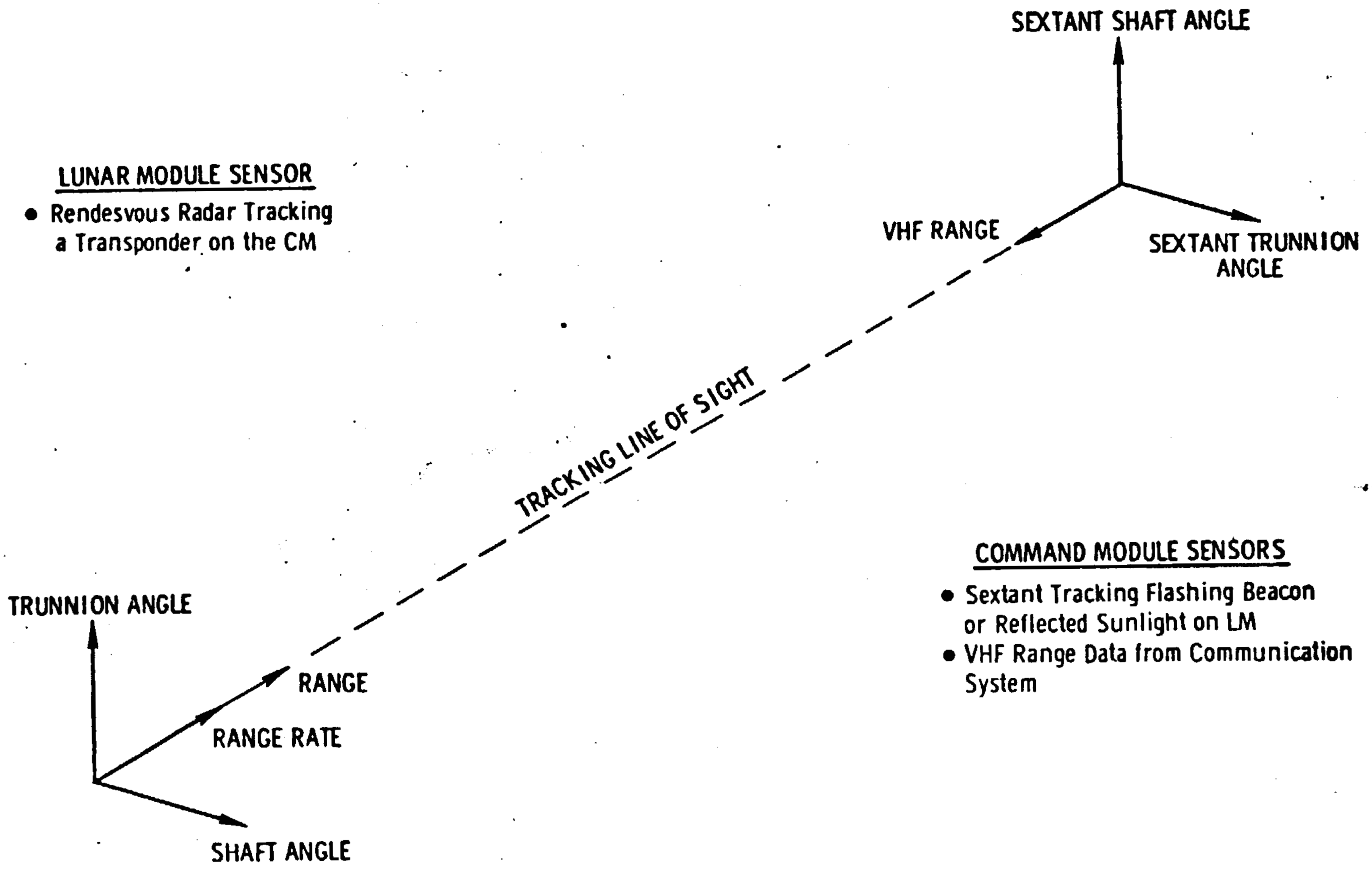

SEXTANT SHAFT ANGLE
LUNAR MODULE SENSOR
• Rendesvous Radar Tracking a Transponder on the CM
VHF RANGE
SEXTANT TRUNNION ANGLE
TRACKING LINE OF SIGHT
COMMAND MODULE SENSORS
• Sextant Tracking Flashing Beacon or Reflected Sunlight on LM
• VHF Range Data from Communication System
TRUNNION ANGLE
RANGE
RANGE RATE
SHAFT ANGLE

Figure A. 2-2 Command and Lunar Module Rendezvous Measurement data used in both the LM and CM for rendezvous navigation. Even though the tracking sensors on the two vehicles are quite different, the identical navigation concept is used in each vehicle for rendezvous navigation.

As mentioned before, a single navigation concept is used in the Apollo spacecraft GN&C Systems for all coasting phases of the mission. A simplified rendezvous navigation functional diagram is shown in Fig. A.2-3 and is similar to that for the cislunar navigation of Fig. A.1-4, except for differences required by the tracking sensor and target vehicle. It might be noted that the cislunar navigation and trajectory control is essentially a rendezvous problem between the spacecraft and the moon, so it is not surprising that the same navigation concept can be applied for the cislunar and rendezvous phases. With reference to Fig. A.2-3, the active and passive vehicle state vectors are extrapolated to the time a navigation measurement is to be taken by the coasting-integration program. An estimate of the rendezvous measurement, A_{EST}, is computed from the two extrapolated state vectors and subsequently compared with the measured tracking data, A_M. The difference, δQ, is then combined with the appropriate weighting vector to compute an update, $(\delta \underline{r}, \delta \underline{v})$, to the spacecraft estimated state vector. Several important points should be noted in this operation. First, the operation just described is done sequentially for each of the four tracking data (range, range rate, antenna shaft angle, antenna trunnion angle) that constitute a navigation measurement in the Lunar Module GN&C System (Fig. A.2-2) and, likewise, sequentially for the two sextant angles and VHF range data in the Command Module. Second, the computed state-vector update, $(\delta \underline{r}, \delta \underline{v})$, for each tracking measurement is automatically checked in the computer against a preselected threshold. If the magnitudes of the computed $\delta \underline{r}$ and $\delta \underline{v}$ are both less than their respective threshold levels in the state-vector alarm test, the update is automatically incorporated in the state-vector estimate. If they exceed the threshold levels, the astronaut is informed and must decide whether to incorporate or reject the update. Typically, the navigator would reject the update until he were sure that the proper target was being correctly tracked and then accept later updates. The state-vector update monitoring in the rendezvous-navigation program is, therefore, a semiautomatic operation, whereas it is a completely manual operation in the cislunar navigation program. As shown in Fig. A.2-3, either the active or passive vehicle state vector can be updated by the rendezvous navigation program. This decision is made early in the mission and is not normally changed thereafter. The velocity changes resulting from rendezvous maneuvers are automatically incorporated into the active vehicle

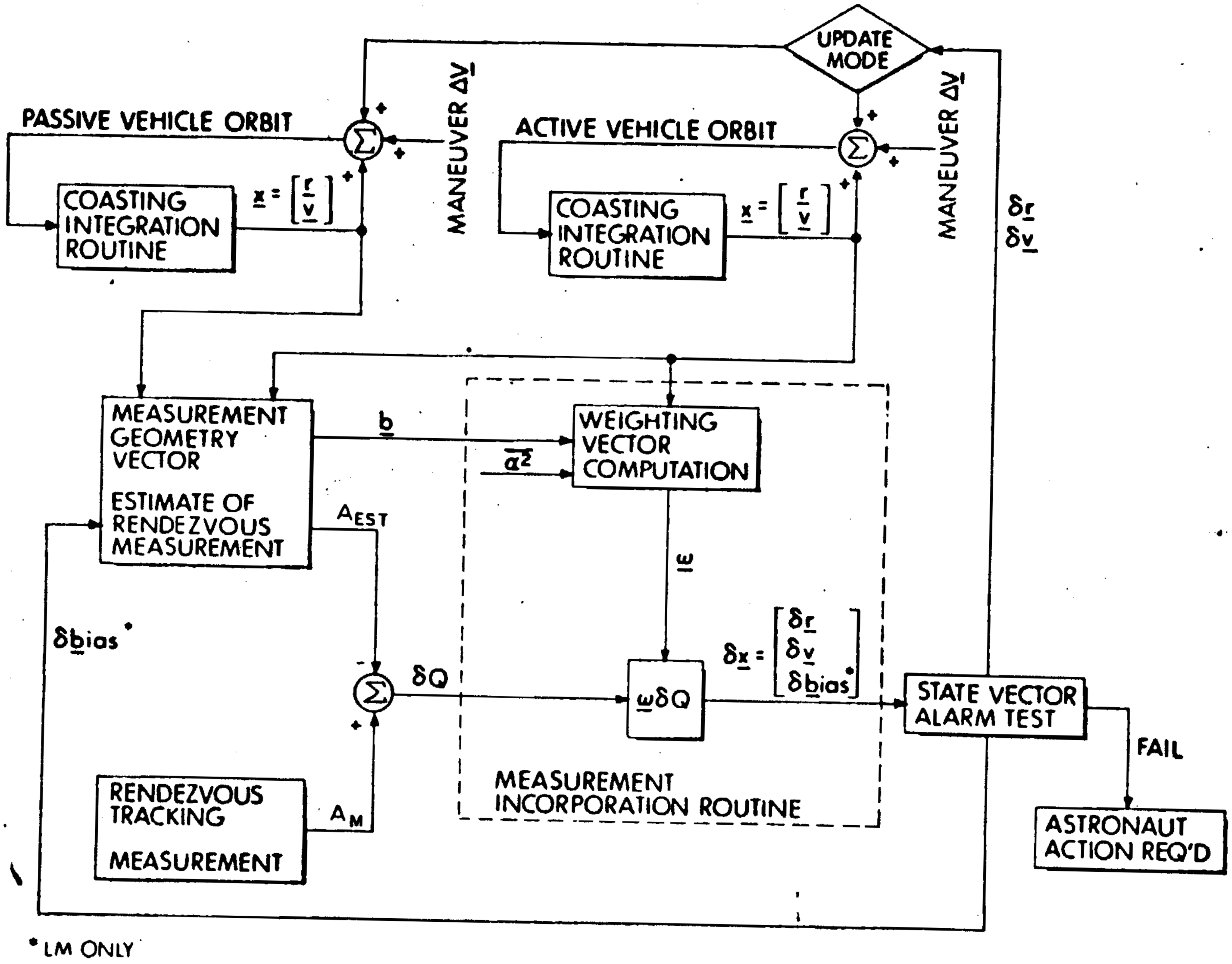

UPDATE MODE
PASSIVE VEHICLE ORBIT
ACTIVE VEHICLE ORBIT
MANEUVER $\Delta \underline{V}$
MANEUVER $\Delta \underline{V}$
COASTING INTEGRATION ROUTINE
COASTING INTEGRATION ROUTINE
$\underline{x} = \begin{bmatrix} \underline{r} \\ \underline{v} \end{bmatrix}$
$\underline{x} = \begin{bmatrix} \underline{r} \\ \underline{v} \end{bmatrix}$
Σ
Σ
Σ
+
+
+
+
+
+
+
-
$\delta \underline{r}$
$\delta \underline{v}$
MEASUREMENT GEOMETRY VECTOR
ESTIMATE OF RENDEZVOUS MEASUREMENT
$\underline{b}$
$\overline{\alpha^2}$
WEIGHTING VECTOR COMPUTATION
$\underline{\omega}$
A_{EST}
$\delta \underline{b}_{ias}$*
δQ
$\underline{\omega} \delta Q$
$\delta \underline{x} = \begin{bmatrix} \delta \underline{r} \\ \delta \underline{v} \\ \delta \underline{b}_{ias}^* \end{bmatrix}$
STATE VECTOR ALARM TEST
FAIL
ASTRONAUT ACTION REQ'D
RENDEZVOUS TRACKING MEASUREMENT
A_M
MEASUREMENT INCORPORATION ROUTINE
* LM ONLY

Figure A.2-3 Simplified Functional Diagram of Rendezvous Navigation

system through the IMU and are then communicated to the other vehicle for incorporation by the astronaut in the passive GN&CS. Finally, in the case of LM rendezvous navigation, the state vector and error-transition matrix used in the navigation-measurement incorporation routine of Fig. A.2-2 are increased from six to nine dimensions to estimate the angle biases in the rendezvous-radar tracking data. The rendezvous radar is not rigidly mounted to the inertial-measurement and navigation base (as are the CM optics), and the structural bias between the radar antenna and inertial unit is, therefore, estimated along with the six dimensions of the position and velocity of the state vector. In practice, only two of the additional dimensions are used for antenna-bias estimation, with the ninth element set to zero. As shown in Fig. A.2-3, the estimated antenna-bias angles are automatically incorporated in the estimated rendezvous measurement calculation.

Figure A.2-4 illustrates a simplified rendezvous-navigation angle-measurement incorporation similar to that shown in Fig. A.1-5 for the cislunar-navigation case. In the example of Fig. A.2-4, the position correction $\delta \underline{x}$ does not lie completely along the measurement geometry vector, $\underline{b}$, because of the correlation represented in the weighting vector between the error in the measured direction (represented by $\underline{b}$ in this case) and the errors in the unmeasured position and velocity directions. As mentioned in the discussion of cislunar navigation, this correlation is zero for the first angle measurement, but then builds up over subsequent measurements taken along the trajectory. In the rendezvous-navigation case, direct measurements of range and range rate are made in the LM GN&CS (Fig. A.2-2), so the velocity components normal to the line-of-sight are the only dimensions of the state vector dependent upon correlation for updating. In the CM GN&CS case, there are no direct navigation measurements of velocity in any direction, and this update information is completely dependent upon correlation. The navigation concept was most dramatically demonstrated in the first manned Apollo mission (Apollo 7), in which optical-sextant tracking was used to control a successful rendezvous intercept (TPI) and the following midcourse-correction maneuvers. During Apollo 9, the CM acted as a backup and monitor to the active LM during rendezvous, again using only sextant tracking data for the onboard navigation measurement.

The rendezvous recursive-navigation program employs various approximations and linearizations to make the implementation of the navigation computation,

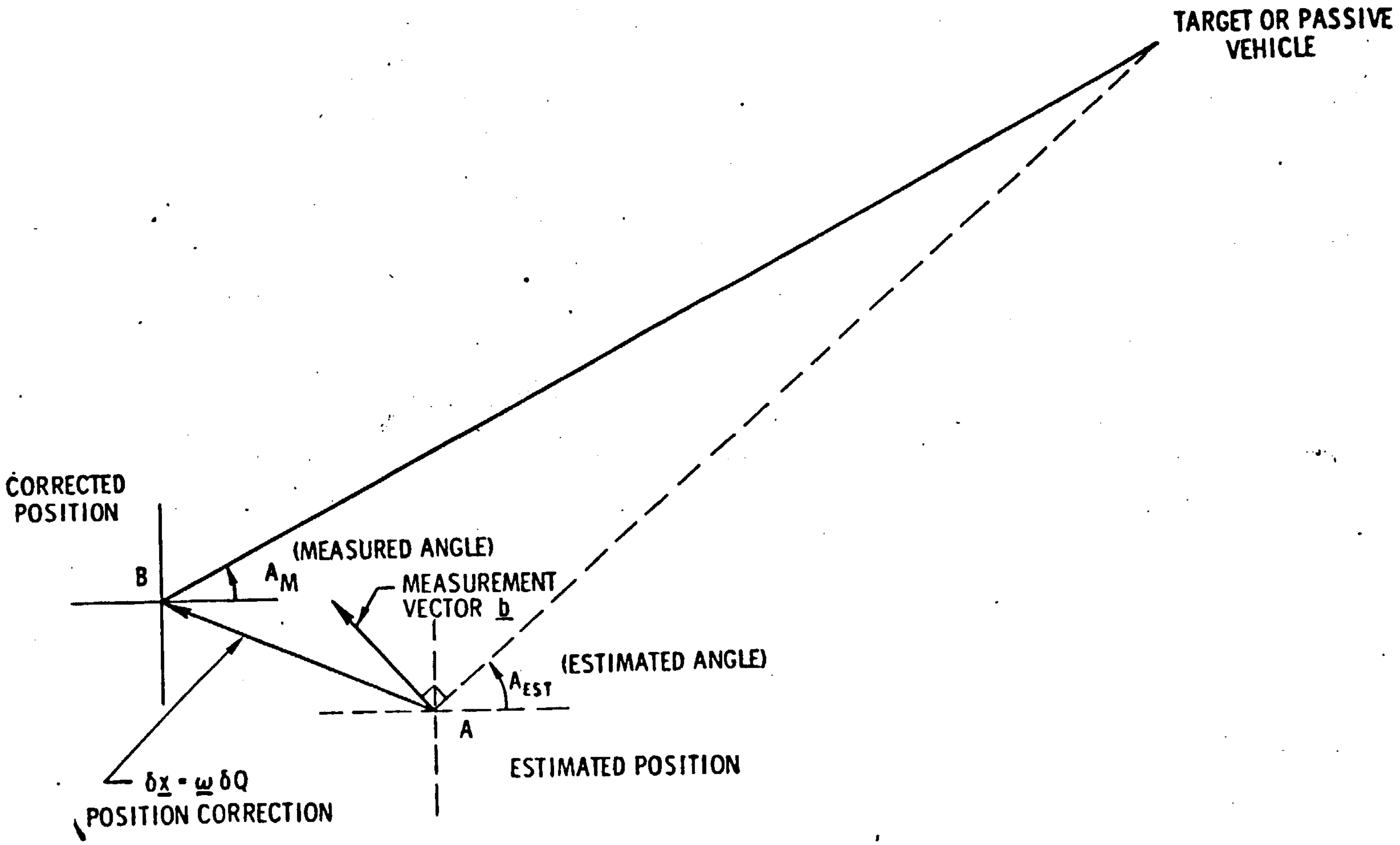
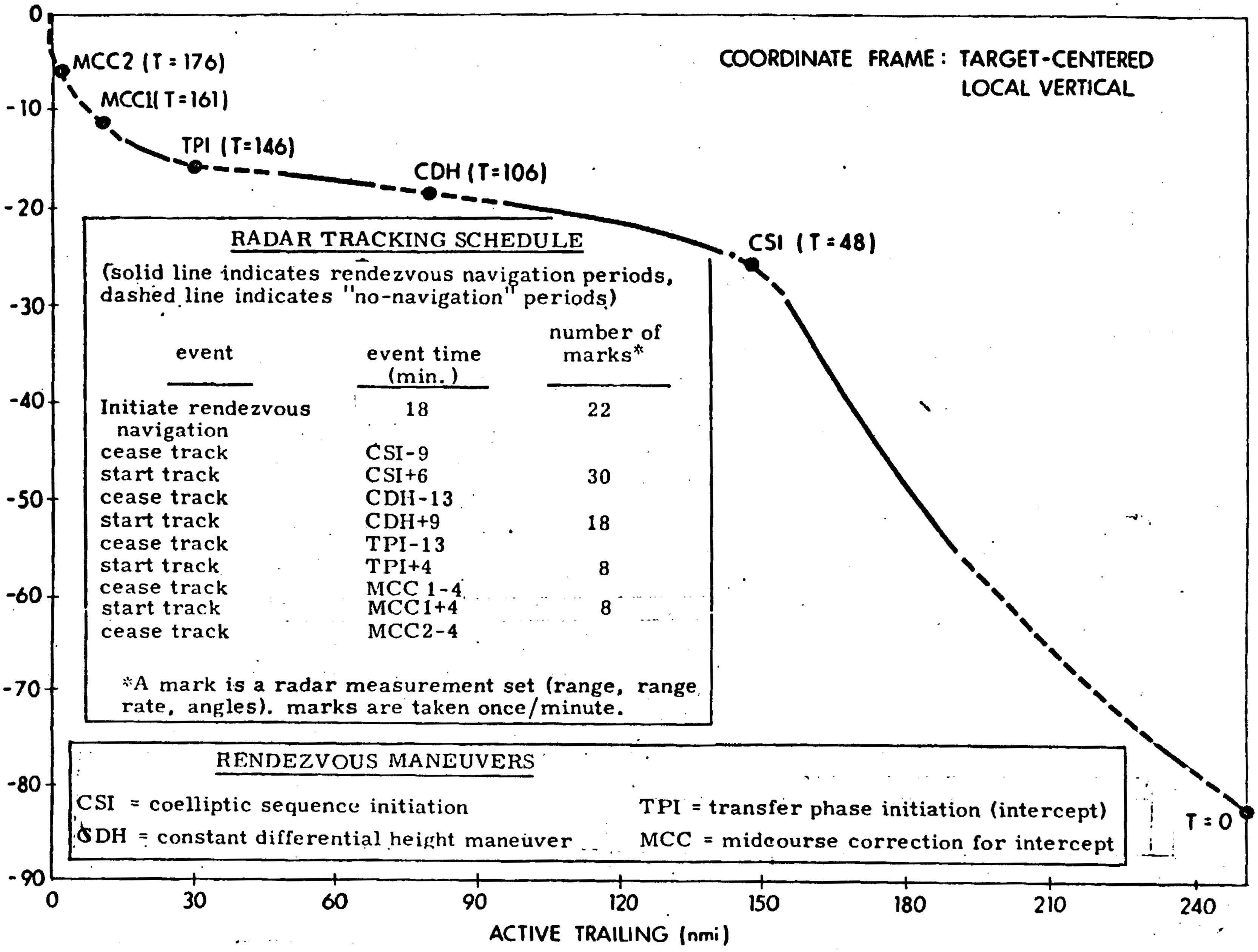

TARGET OR PASSIVE VEHICLE
CORRECTED POSITION
B
(MEASURED ANGLE)
A_M
MEASUREMENT VECTOR $\underline{b}$
(ESTIMATED ANGLE)
A_{EST}
A
ESTIMATED POSITION
$\delta\underline{x} = \underline{\omega}\,\delta Q$
POSITION CORRECTION

Figure A.2-4 Simplified Rendezvous-Navigation Angle Measurement Incorporation

practical. As a result, the accuracy of the error-transition matrix used in the weighting vector computation of Fig. A.2-3 degrades, resulting in erroneous correlation information after extended tracking periods. The filter matrix is, therefore, periodically reinitialized during the rendezvous-navigation tracking phases. It has been determined in cislunar-navigation simulations, however, that a single W-matrix initialization at the start of the sighting schedule provides sufficient accuracy.

Figure A.2-5 represents a relative-trajectory profile for the nominal lunar-landing-mission rendezvous phase. This trajectory is the same as that shown in Fig. A.2-1, except that the coordinates (0,0) are centered on the passive CM vehicle. The solid-line portions of the trajectory in Fig. A.2-5 represent the rendezvous-navigation phases where tracking data are taken at one-minute intervals in the LM GN&CS. The CM takes similar, if not slightly extended, tracking intervals, but both vehicles suspend navigation prior to major trajectory-correction maneuvers in order to prepare for targeting and execution of these maneuvers. The CM GN&CS normally computes a mirror-image maneuver of that computed by the LM so that it can execute a retrieval, should the LM fail to complete the maneuver.

During the rendezvous phases of an Apollo mission, three active navigation systems normally operate during the entire rendezvous profile. These are the LM, CM, and earth-tracking navigation systems. During the later phases of the rendezvous profile (TPI maneuver preparation to intercept), two additional navigation monitors are active: the LM Abort Guidance System, and crew observations checked against precomputed "chart" solutions for the major rendezvous maneuvers. A measure of the consistency of the three major navigation systems during rendezvous can be gauged by comparing the rendezvous maneuvers computed by each of these systems, since their computations are based upon the navigated state vectors established independently by each system using different tracking sensors. Post-flight analysis of the lunar-rendezvous phases of the Apollo 10 and 11 missions show that there was very close agreement in all cases where comparisons can be made, indicating a high degree of accuracy for all three rendezvous-navigation systems and concepts. The flight experience provided by the five Apollo rendezvous missions to date indicates that the rendezvous navigation concept used in the spacecraft GN&C Systems is highly accurate and versatile in its capability to use a variety of types of navigation measurements.



RADAR TRACKING SCHEDULE

(solid line indicates rendezvous navigation periods, dashed line indicates "no-navigation" periods)

event	event time (min.)	number of marks*
Initiate rendezvous navigation	18	22
cease track	CSI-9	
start track	CSI+6	30
cease track	CDH-13	
start track	CDH+9	18
cease track	TPI-13	
start track	TPI+4	8
cease track	MCC 1-4	
start track	MCC1+4	8
cease track	MCC2-4	

*A mark is a radar measurement set (range, range rate, angles). marks are taken once/minute.

RENDEZVOUS MANEUVERS

CSI = coelliptic sequence initiation

CDH = constant differential height maneuver

TPI = transfer phase initiation (intercept)

MCC = midcourse correction for intercept

Figure A. 2-5 Typical Relative Motion Plot

A.3 Orbital Navigation

As mentioned earlier, the basic objective of all the coasting-flight navigation routines is to maintain estimates of the position and velocity vectors of both the CSM and LM. Coasting-flight navigation achieves this goal by extrapolating the six-dimensional state vector, using a Coasting Integration Routine; and by updating or modifying this estimate with tracking data gathered by the recursive method of navigation (see Section A.1).

As with cislunar and rendezvous navigation, the basic input to the orbital-navigation routine is scanning telescope or sextant tracking-angle data indicated to the computer when the astronaut depresses the MARK button signifying that he has centered the optical reticle on the tracking target—which in orbital navigation is a landmark. The primary output of the orbital navigation routine is the estimated CSM state vector and estimated landmark coordinates. A simplified orbital-navigation functional diagram is shown in Fig. A.3-1 as similar to Figs. A.1-4 and A.2-3 for cislunar and rendezvous navigation. The navigation procedure involves computing an estimated tracking measurement, A_{EST} based on the current state-vector estimates. This estimated measurement is then compared with the actual tracking measurement, A_M, to form a measured deviation δQ. A statistical weighting vector, $\underline{\omega}$, is computed from statistical knowledge of state-vector uncertainties and tracking performance, $\overline{\alpha^2}$, plus a geometry vector, $\underline{b}$, determined by the type of measurement being made. The weighting vector, $\underline{\omega}$, is defined such that a statistically-optimum linear estimate of the deviation, $\delta\underline{x}$, from the estimated state-vector is obtained when the weighting vector is multiplied by the measured deviation δQ. The vectors $\underline{\omega}$, $\underline{b}$ and $\delta\underline{x}$ are of nine dimensions for orbital navigation.

To prevent unacceptably large incorrect state-vector changes, certain validity tests are included in the navigation procedure; the astronaut tracks a landmark and acquires a number of sets of optical angle data before the state-vector updating process begins. During the data-processing procedure the landmark is out of sight, and it is not possible to repeat the tracking. Before the first set of data is used to update the estimated state vector, the magnitudes of the proposed changes in the estimated CSM position and velocity vectors, δr and δv, respectively, are displayed for astronaut approval. In general, successive accepted values of δr and δv decrease during the processing of the tracking data associated with one landmark. Thus, if

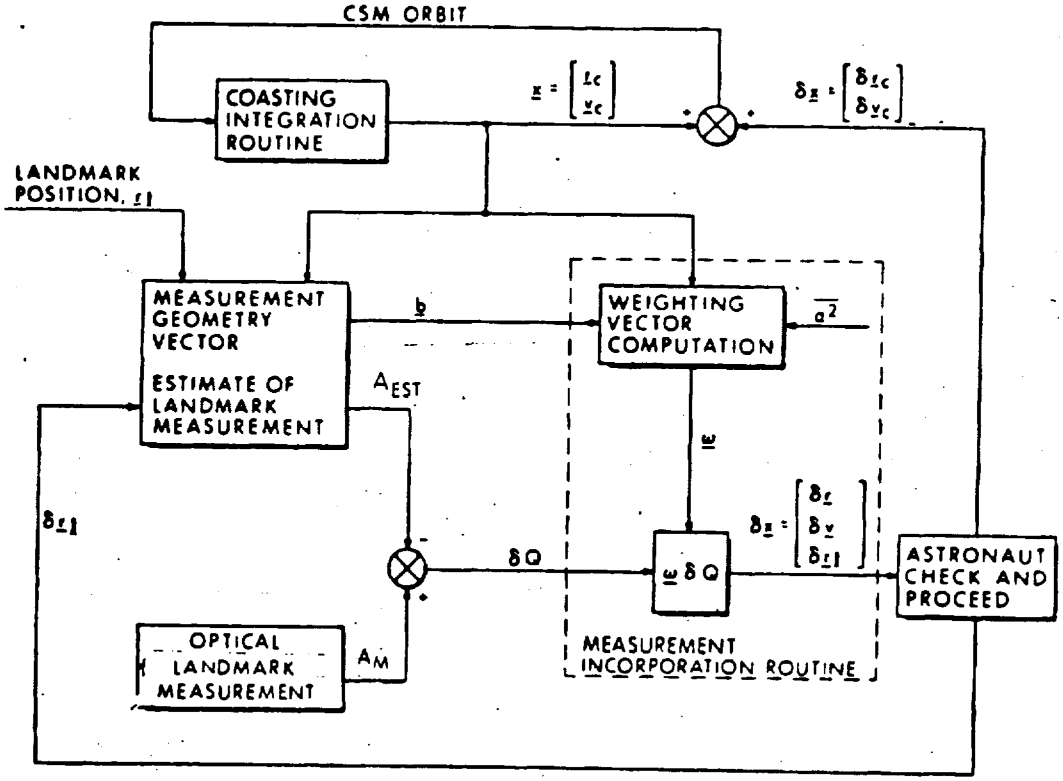

CSM ORBIT
COASTING INTEGRATION ROUTINE
$\underline{x} = \begin{bmatrix} \underline{r}_c \\ \underline{v}_c \end{bmatrix}$
+
+
$\delta\underline{x} = \begin{bmatrix} \delta\underline{r}_c \\ \delta\underline{v}_c \end{bmatrix}$
LANDMARK POSITION, $\underline{r}_l$
MEASUREMENT GEOMETRY VECTOR
ESTIMATE OF LANDMARK MEASUREMENT
$\underline{b}$
A_{EST}
WEIGHTING VECTOR COMPUTATION
$\overline{a^2}$
$\underline{\omega}$
$\delta\underline{x} = \begin{bmatrix} \delta\underline{r} \\ \delta\underline{v} \\ \delta\underline{r}_l \end{bmatrix}$
$\delta\underline{r}_l$
-
+
δQ
$\underline{\omega}\,\delta Q$
ASTRONAUT CHECK AND PROCEED
OPTICAL LANDMARK MEASUREMENT
A_M
MEASUREMENT INCORPORATION ROUTINE

Figure A. 3-1 Simplified Functional Diagram of Orbital Navigation the MARK REJECT button has been used to erase all inaccurate marks, all state-vector updates should be either accepted or rejected. If the first displayed values of δr and δv are judged to be valid, all data associated with that landmark are accepted.

The orbit navigation routine can be used in lunar orbit in lunar-landing missions and in earth orbit during abort situations or alternate missions. This routine has further extensive onboard-navigation and landmark-mapping capabilities, but these are not yet being used in the fashion originally intended, since optical marks are not processed onboard. Consequently, as in cislunar navigation, the primary mode of navigation for the Apollo orbital phases is the Manned Space Flight Network and its Real Time Computation Center in Houston.

Procedures that ensure proper landmark acquisition and marktaking are a precondition to successful landmark navigation. To initially acquire and maintain optical tracking, the CSM must be oriented such that the CSM-to-landmark line-of-sight falls within the scanning telescope's field of view. In the CSM GN&CS there is no automatic vehicle-attitude control during the landmark-tracking procedure. Consequently, any desired attitude control must be accomplished manually by the astronaut using the Rotational Hand Controller or Minimum Impulse Controller or by use of the Barbecue Mode Routine.

Should the astronaut wish, he may use the Automatic Optics Positioning Routine to aid in the acquisition of the landmark. This routine has two modes which are relevant to orbit navigation. In the landmark mode (which is useful for acquisition of a specified landmark), the routine drives the optics to the estimated direction of the specified landmark. The computations and positioning commands in this routine are repeated periodically provided the optics mode switch is properly set. In the advanced ground-track mode (which is useful in lunar orbit for surveillance, selection, and tracking of possible landing sites), the routine drives the CSM optics to the direction of the point on the ground track of the spacecraft at a time slightly more than a specified number of orbital revolutions ahead of current time. Thus, in the advanced ground-track mode, the astronaut is shown continuously the ground track of the CSM for a future revolution. The basis for this mode is that it is desirable to select a landing site near the CSM orbital plane at the LM lunar-landing time.

After the astronaut has acquired the desired landmark (not necessarily the one specified to the Automatic Optics Positioning Routine), he switches the optics mode to MANUAL and centers the scanning telescope or sextant reticle on the landmark. When accurate tracking is achieved, he presses the optics MARK button, causing the time of the measurement and all optics and IMU gimbal angles to be stored in the AGC. Up to five unrejected navigation sightings of the same landmark may be made during the tracking interval, and all sets of navigation data must be acquired before processing of the data begins.

After the astronaut has completed the tracking of a landmark, he is asked by the computer whether he wishes to identify the tracked landmark. If he does, he then enters into the AGC through the keyboard, the coordinates of the landmark.

Should the astronaut not identify the landmark, the Landing Site Designation procedure is then used for the navigation-data processing. In this process the landmark is considered to be unknown, and the first set of navigation data is used to compute an initial estimate of the landmark location. The remaining sets of data are then processed to update the estimated nine-dimensional CSM-landmark state vector.

Whether the landmark is identified or not, one further option is available to the astronaut. He may specify that one of the navigation sightings is to be considered the designator for an offset landing site near the tracked landmark. In this case, the designated navigation data set is saved, the remaining sets of data are processed as described above, and then the estimated offset landing-site location is determined from the saved data. This procedure offers the possibility of designating a landing site in a flat area of the moon near a landmark suitable for optical navigation tracking, but not for landing.

Each set of navigation data used for state-vector updating and not for landing-site designation or offset produces two updates. For the first navigation-data set, the magnitudes of the first proposed changes in the estimated CSM position and velocity vectors, δr and δv, respectively, are displayed for astronaut approval. If the astronaut accepts these proposed changes, then all state-vector updates will be performed, and all the information obtained during the tracking of this landmark will be incorporated into the state-vector estimates.

After all of the sets of navigation data have been processed, the final estimated landmark-position vector is converted to latitude, longitude, altitude coordinates displayed on the DSKY. Should the astronaut designate the tracked landmark to be the landing site, then the landing-site coordinates and landing-site vector are saved in erasable memory. In this manner, the original coordinates of the landing site can be revised or a new landing site selected.

Figure A.3-2 shows the geometry for tracking a landmark in a 60-nmi circular lunar orbit. Recommended marktaking technique requires that five marks be taken equally spaced over the plus-55-to-minus-55-deg marktaking window. The advantage of oblique lines-of-sight on the first and last marks diminishes rapidly beyond ± 45 deg. Consequently, marks taken symmetrically and at equally spaced intervals are preferred to marks taken asymmetrically at the extremes of the marktaking window. The interval between marks for the 76-deg (100-sec) minimum marktaking window is 19 deg (25 sec); for the 110-deg (180-sec) maximum window, the interval between marks is 27.5 deg (45 sec).

The final operation is to convert the nine-dimensional error-transition matrix to a six-dimensional matrix with the same CSM position and velocity estimation error variances and covariances. The reason for this procedure is that the nine-dimensional matrix, when it is initialized for processing the data associated with the next landmark, must reflect the fact that the initial landmark-location errors are not correlated with the errors in the estimated CSM position and velocity vectors. Of course, after processing measurement data, these cross correlations become non-zero, and it is for this reason that the nine-dimensional procedure works, and that it is necessary to convert it finally to a six-dimensional form.

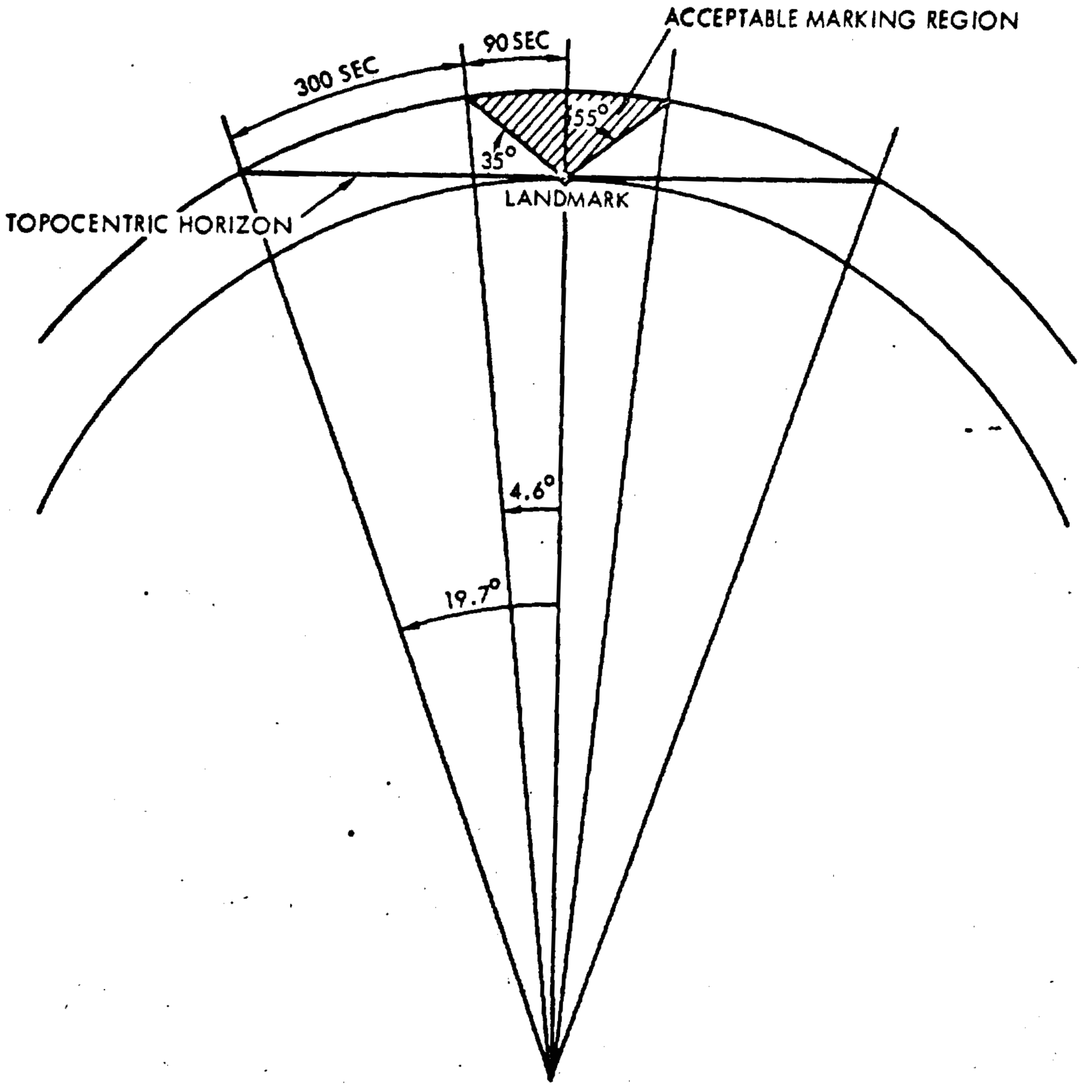


Figure A. 3-2 Landmark-Tracking Geometry for a 60-Nautical-Mile Circular Lunar Orbit

APPENDIX B

MAJOR PROGRAM CAPABILITIES— Targeting

As stated in Section 2.2.1, targeting is the computation of the maneuver required to continue on to the next step in the mission. Specifically, targeting computes the velocity change required of the spacecraft to obtain a certain objective, such as a change of orbit or a certain reentry corridor or an aimpoint. An aimpoint can itself be quite variable; it can be a point on the surface of the moon or a point in space where another vehicle will be at a specified time in the future or a location below and behind that vehicle at that same projected time.

The AGC does not possess a targeting capability for every phase of the lunar landing mission; consequently, the Real Time Computation Center (RTCC) in Houston provides the targeting for many of the nominal and abort phases of the lunar mission. There are, in fact, five classes of maneuvers, four of which involve targeting:

1. Pretargeted maneuvers comprise earth-orbit insertion, translunar injection, lunar landing, lunar ascent, and reentry.
2. Ground-targeted maneuvers comprise lunar-orbit insertion, transearth injection, descent-orbit insertion, various orbital changes around the moon, translunar and transearth midcourse corrections and return-to-earth aborts.
3. Rendezvous maneuvers comprise coelliptic-sequence initiation (CSI), constant differential height (CDH), transfer-phase initiation (TPI), transfer-phase midcourse (TPM), and out-of-plane maneuvers.
4. Return-to-earth (RTE) maneuvers comprise cislunar aborts which might occur after loss of communication with the ground.
5. Untargeted maneuvers comprise docking, passive thermal control, crew-originated attitude maneuvers, etc.

Clearly, untargeted maneuvers need not be discussed here. Pretargeted maneuvers have unchanging objectives which are included within the actual program computations. Ground-targeted, rendezvous and return-to-earth maneuvers may have varying objectives which may not be anticipated beforehand; consequently, onboard targeting programs permit considerable flexibility in the prevailing conditions and objectives when they are implemented. It is the latter targeting programs which will be discussed in this appendix, as well as the targeting computations upon which these programs are based.

B.1 Targeting Computations

Targeting programs are classified by the type of maneuver targeted (either External ΔV or Lambert) or by the method of computation (iterative or noniterative). External ΔV is an open-loop, constant-attitude maneuver which permits easy out-of-the-window monitoring. To date, all ground-targeted maneuvers have used External ΔV. The principal disadvantage of External ΔV is that it is open loop with respect to the targeted conditions (required velocity). Any variations from the RTCC-assumed models for thrust and mass flow during the burn can result in a trajectory which could require a further trimming maneuver—and hence cause a propellant penalty. Computation of required velocity for a generalized External-ΔV maneuver (External-ΔV targeting, as opposed to External-ΔV guidance) is extremely complicated, effectively precluding an onboard AGC External-ΔV targeting capability.

Lambert maneuvers, however, are closed-loop with respect to the targeted conditions, in that they periodically update required velocity, a function of present and targeted state. Thus the effects of non-nominal thrust and flow rate are minimized. A disadvantage of Lambert targeting is that it lends itself to intercept problems, as opposed to trajectory-shaping problems. As a result, Lambert targeting accommodates only a small proportion of the maneuvers required in an Apollo mission.

Several targeting techniques are used in Apollo—some of which are External-ΔV or Lambert and all of which employ External-ΔV or Lambert guidance. The onboard return-to-earth targeting program (P37) produces a conic solution which is utilized by Lambert guidance. CSI and CDH targeting prepare inputs for External-ΔV guidance. TPI and TPM are Lambert problems and utilize Lambert targeting to generate Lambert-guidance inputs directly. Ground-targeted maneuvers use whichever targeting techniques will accomplish the current goal and generate inputs to External-ΔV guidance.

The choice between an iterative or noniterative method of computation depends, generally, on the extent to which perturbations affect the solution. Since no analytic expression completely describes the forces acting upon a vehicle traveling between the earth and the moon, targeting of such a trajectory involves first an analytic approximation, then orbital integration to determine the error, a second approximation to compensate for the error, and so on, bracketing the solution until either an imposed iteration limit is reached or the approximation converges on the desired solution.

The accuracy of any rendezvous computation depends upon a good knowledge of the state vectors of the two vehicles with respect to each other. Since coelliptic-sequence initiation is performed after injection or abort, the initial estimate of the LM state vector could be quite poor. Normally, ample time is available for repeated rendezvous navigation to improve the probability of good state-vector estimates before the CSI maneuver. Even in the off-nominal case, there would be sufficient time to take a certain minimum number of marks to ensure a good rendezvous.

Average G (see Section C.1.1.1), which improves knowledge of the state vector during powered flight, tends also to slightly degrade the estimate of that vector due to accelerometer uncertainties; thus rendezvous navigation is needed repeatedly to ensure the high quality of the state vector.

B.2 Ground-Targeted Maneuvers

All ground-targeted maneuvers are transmitted to the AGC via voice or telemetry uplink. Sufficient data could be transmitted to permit immediate execution of a powered-flight program but, instead, an onboard pseudo-targeting buffer program (P30) is executed prior to the maneuver. This pseudo-targeting approach has several advantages over direct maneuver execution: it provides meaningful (perhaps critical) displays to the astronaut; it can itself generate many of the inputs required by the guidance program, permitting a significant reduction in the required number of uplink variables (especially important for voice uplinks which must be entered via the DSKY); and it is designed to accept conceptually simple inputs for a crew-originated maneuver in an emergency situation when ground communication is unavailable. Furthermore, this approach serves as a backup for the onboard rendezvous-targeting programs in the highly unlikely event that the onboard primary systems in both the CM and LM fail.