# Applications of Different Design Methodologies in Navigation Systems and Development at JPL

S. W. Thurman
Navigation Systems Section

*The NASA/JPL deep space navigation system consists of a complex array of measurement systems, data processing systems, and support facilities, with components located both on the ground and on board interplanetary spacecraft. From its beginnings nearly 30 years ago, this system has steadily evolved and grown to meet the demands for ever-increasing navigation accuracy placed on it by a succession of unmanned planetary missions. Principal characteristics of this system are its unique capabilities and great complexity. In this article, three examples in the design and development of interplanetary space navigation systems are examined in order to make a brief assessment of the usefulness of three basic design theories, known as normative, rational, and heuristic. Evaluation of the examples indicates that a heuristic approach, coupled with rational-based mathematical and computational analysis methods, is used most often in problems such as orbit determination strategy development and mission navigation system design, while normative methods have seen only limited use in such applications as the development of large software systems and in the design of certain operational navigation subsystems.*

## I. Introduction

The ability to accurately navigate a spacecraft often plays a significant role in the success or failure of unmanned interplanetary space missions. In the context of the NASA planetary exploration program, navigation is defined as the process of determining the current and predicted flight path of a space probe, and controlling that flight path to meet stated mission objectives [1,2]. The navigation system to be used for each mission is developed and configured during pre-flight mission design and planning, based on the trajectory design developed as part of the same process. The result is a mission-specific system which is composed of several basic "building blocks" common to all missions and elements unique to the mission at hand.

Figure 1 shows a simplified diagram of the "generic" deep space navigation system.[1] As is evident from Fig. 1,

[1] C. E. Kohlhase, "Navigation Systems Overview," Presentation Viewgraphs, Navigation Systems Section, Jet Propulsion Laboratory, Pasadena, California, March 16, 1973.

the principal traits of the system are its great complexity and unique capabilities (it is a special purpose system and essentially the only one of its kind). Each planetary mission has its own special capabilities, flight path, and science objectives, which make it necessary to develop a custom navigation system for every mission flown, even though these systems share many elements in common such as the tracking facilities of the Deep Space Network (DSN). Some examples of the building blocks used in deep space navigation are orbit determination and maneuver analysis software, spacecraft propulsion and imaging systems, and radio tracking networks (primarily the DSN).

What follows is a brief evaluation of three examples that show how different system design methodologies—normative, rational, and heuristic—are used in the design and development of navigation system software, orbit determination strategy and methods, and the design of navigation systems for specific missions. Before beginning, a brief description of each of the three design theories is in order. The descriptions given here are those put forth by Rechtin [3]. A normative theory is one in which system design is accomplished by following a set of rules and principles that are rooted in the values of the creator(s) of the theory. The definitions of what constitutes "good" and "bad" designs are largely judgmental pronouncements. A rational design theory is based on the idea that some generalized set of procedures for design and problem solving can be used to develop any system design, regardless of the system's purpose or functions. Rational theories typically make extensive use of mathematical analysis-based tools, such as the calculus of variations and probability theory. The heuristic approach to design shares some commonality with both the normative and rational approaches, but it is based more on insights and guidelines derived from experience rather than on rules and pronouncements or mathematical methods. Studying past and present applications of these design methodologies may provide some insight into their use for architecting the navigation systems of future planetary missions.

## II. Example 1: Development of the JPL Orbit Determination Program

The Orbit Determination Program (ODP) is actually a large set of programs used to process radio tracking data and spacecraft onboard optical data, then construct a flight path which fits all of the observational data included in the solution. The design and evolution of this system provide an opportunity to evaluate the usefulness of some of the software engineering disciplines in developing very large, computationally intensive software.

Although it is difficult to rigidly classify many software engineering methods as being solely normative, rational, or heuristic, certain ones can be identified fairly closely with a single design methodology. One of the better known techniques in software engineering is "top-down" development, which consists of a set of guidelines, based on experience, for recursively partitioning a large problem into smaller ones in a hierarchical manner. Top-down development is primarily heuristic in nature [4]. Structured coding, on the other hand, is a programming technique that is somewhat heuristic, but has many traits that match the description of a normative theory given above; that is, it consists of a set of rules and principles that are heavily experience-based and largely judgmental.

The basic idea behind structured programming is that code written with only a specific, well-defined set of control constructs and principles will result in the best possible program—best being defined primarily in terms of readability and ease of maintenance, which were considered by the creators of the discipline to be of overriding importance [5]. If several different programmers were given a program design and using structured coding asked to write code to implement it, the end product arrived at by each programmer should be nearly the same. Other important criteria deemed good in structured programming are small, single-purpose program modules, each having only one entry point and one exit point.

The single most outstanding trait of the ODP throughout its 27-year history is its tremendous complexity. Even in its original form in 1962, the ODP contained programs capable of calculating spacecraft trajectories throughout a sizeable portion of the solar system and solving for up to 63 parameters using 13 different tracking data types [6,7]. While the description which follows is of the original first-generation ODP, the basic structure and organization of the system is preserved in the current ODP, even though the system is run differently on today's computers.

The basic design of the system can be seen in Fig. 2, which shows a high-level view of the organization of ten programs, known as links, comprising the first-generation ODP [7]. Figure 2 shows that the ODP was organized in a top-down, hierarchical fashion even in the early 1960's when software engineering was still in its infancy. The system was broken down in such a way that each major function in the orbit determination process was performed by a stand-alone program. Each program used output files generated by the previous programs as inputs. A single set of user-supplied instructions was used to execute all of the programs, which were run sequentially. This design espouses such heuristics as the matching of form to function and the use of system elements with high internal complexity and low external complexity [3], resulting in a system that performed many complicated functions but was relatively simple to use.

Of primary importance for programs of such size and complexity is the efficient use of computer memory (this was especially true in the first ODP, which was run on an IBM 7090 computer with limited memory) and the minimization of communications between subprograms. Minimizing communications also minimizes the amount of time required to run the various links, as the individual programs which make up the ODP are known. Even when using modern computers with virtual memory capabilities (the ability to run a program of unlimited size), in order to minimize run time, it is important to keep to a minimum the memory required to run a given program so that the computer spends as little time as possible swapping inactive program segments and active segments in and out of its memory.

The modern ODP system, now in its third generation at JPL, was designed and written using top-down design and structured coding principles. It has been used successfully in all planetary missions dating back to the Mariner 6 and 7 missions to Mars in 1969: Mariner–Mars '69, Mariner–Mars '71, Mariner–Venus–Mercury '73, Pioneers 10 and 11, Viking, Voyager, Pioneer–Venus Orbiter, and currently the Magellan and Galileo missions. However, there are two goals of the ODP that have continually been in conflict with the rules of structured coding—minimizing run time and storage use. The ODP source code was written in FORTRAN, and extensive experience in the use of FORTRAN for computationally intensive applications has led to the development of heuristic guidelines for the optimization of FORTRAN programs in terms of run time and storage use [8]. Examples of guidelines for storage and run time optimization are the use of COMMON storage blocks for sharing data among multiple subprograms, minimizing memory usage and subprogram communication, and the use of a minimum number of subprograms to accomplish necessary tasks, since communication among subprograms is very slow relative to most operations performed by FORTRAN.

Some ODP links make extensive use of COMMON, have many subprograms that are quite large (thousands of lines), and have many subprograms with multiple entry points. While the actual code sometimes differs from what is considered to be good code according to structured programming rules, it was implemented in this way so that the links could be run quickly (run times in minutes), and so the software could be accommodated on computers with limited memory capacity. In the early history of the ODP, the concern with regards to memory usage was caused by the limited capability of the mainframe computers in use at the time (the IBM 7090). More recently, the motivation for minimizing storage usage was the desire to use the ODP on smaller minicomputers for dedicated use by designated groups of users. The rigorous use of structured code, with its small, self-contained single-purpose modules, may have yielded an ODP system that would be more readable and maintainable than it is now; but this probably would be of little comfort to the users of the software when faced with run times measured in hours and the need for large, expensive mainframe computers to run the ODP system.

The implementation of the ODP is a balance between the good qualities (readability, maintainability) of structured coding and the special considerations of memory and processing time requirements. It can be seen that while structured programming did play a role in the development of the ODP, the architects of this system tempered the rules of structured coding with previous experience, a more heuristic approach to programming.[2] In summary, the ODP is a result of compromises made by its architects, who had to reconcile the requirements of their system with the rules and guidelines of their software engineering tools.

The modern ODP has evolved into a multi-mission orbit determination tool. It has also been adapted for use on a variety of computers, from mainframes to desk-top workstations. For example, the DSN Multi-Mission Navigation Team runs the ODP system on a VAX 8530 minicomputer, supporting missions for NASA and several international space agencies. The Galileo Navigation Team is running the ODP in its traditional "home" environment, a UNIVAC 1108 mainframe. The Magellan Navigation Team, on the other hand, is presently running the ODP on its own dedicated computer system, centered around a Sun 3/150 workstation. Although each of these user groups is running the same set of programs, each group uses the ODP in a different manner, that is, with different input and output data to meet its own special needs.

## III. Example 2: The Orbit Determination Strategy Design Process

This example provides a brief illustration of the process through which orbit determination strategy is conceived and developed. The development of an orbit determination strategy basically consists of choosing such things as

[2] Interview with John E. Ekelund, ODP cognizant engineer, Navigation Systems Section, Jet Propulsion Laboratory, Pasadena, California, November 2, 1989.

the quantity and types of measurements to be used (Earth-based ranging, Doppler, spacecraft onboard optical, etc.), the type of estimation algorithm employed (batch, sequential, reduced-order, etc.), and the mathematical model to be used by the estimation algorithm. This activity usually takes place as part of a feasibility study, when a new mission is being considered. At this early stage, there is usually little consideration given to any constraints that may be imposed by a specific spacecraft, since a firm spacecraft design is usually not in existence yet. The result of this process is usually a set of preliminary navigation accuracy requirements and an orbit determination strategy (or set of strategies) capable of meeting them. As might be expected, these requirements are often a compromise between what is desired and what is really possible.

Because of its complicated nature, the design and development of orbit determination strategy is primarily a heuristic process, but one that also makes extensive use of rational-based analysis tools to serve as a guide. The description that follows delineates why this is so and gives a brief history of this field as practiced at JPL, using a few specific examples for illustration. Orbit determination is a classic example of the limitations of mathematical tools: There are none that can tell the analyst what the best navigation system will be for a given spacecraft flying a given trajectory. The navigation analyst can choose from a number of different kinds of measurements, each with its own strengths and weaknesses, which can be combined into an orbit determination strategy in a great variety of ways. The number of possibilities quickly destroys any hopes of constructing a mathematical search procedure to seek out and find the "best" system.

The mathematical tools used to exercise and evaluate candidate strategies normally come from optimal estimation theory, a body of knowledge describing how to obtain the best possible estimate of a system given a mathematical description of that system [9,10]. Estimation theory is rational in nature; subject to certain assumptions, it theoretically specifies the estimation algorithm that will yield the best estimate, in a statistical sense, of any system which can be described using a basic mathematical framework. The user only needs a mathematical model for the system in question and some knowledge of pertinent mathematical methods (matrix algebra, linear systems of differential equations) to define the optimal estimator. In orbit determination, the system consists of a set of parameters describing the spacecraft trajectory, tracking station locations, and numerous error sources.

Although it is very powerful, optimal estimation theory has two fundamental limitations in orbit determination applications (and many other fields for that matter). The first is that it gives its user only one accuracy estimate at a time for a single measurement strategy; for example, it does not indicate how much performance improvement would be obtained by the addition of more data and/or different types of measurements in the solution. The second limitation is that the optimal estimates computed are only correct if the mathematical model of the system is correct, which in practice it never really is.

The sensitivity of the results obtained from the estimation techniques used by JPL to unmodeled or poorly modeled parameters were discovered early on in the planetary exploration program. Spacecraft such as Mariner-Mars '65 (Mariner 4) experienced unexpected deviations from the estimated flight path, which were later determined to be caused by small gas leaks in the valves of the attitude control thrusters [11]. Gas leaks and small thruster misalignments—as well as many other effects which are known to exist but are extremely difficult to model—are now known to be present on all interplanetary spacecraft. Another example is the effect of small variations in the Earth's rotation rate on the timing of tracking measurements. A mathematical method, known as consider state analysis, was developed to estimate the effects on orbit determination accuracy of parameters that were known or suspected to influence the problem but were too poorly known to be estimated themselves [7].[3]

While consider state analysis is capable of estimating the effects of parameters not present in the system model, it provides no guide to the navigation analyst about how to change the orbit determination strategy to minimize the influence of these parameters. The general rules given by optimal estimation theory which apply to the behavior of changes in solution accuracy no longer apply when consider states are taken into account; for example, according to optimal estimation theory, if more data are added to a trajectory solution, then the accuracy of that solution must increase (it does not say how much, just that it must increase). Also, if the data used in determining a trajectory are made to be more accurate, then the accuracy of the resulting estimate must improve. The author has personally experienced the contradictions to these well-established rules which commonly arise in consider state analysis.

As discussed above, there are no well-defined rules that tell the analyst which direction to follow when searching

[3] N. D. Panagiotacopulos, *An Introduction to JPL's Orbit Determination Program*, JPL Document 1846-37 (internal document), Jet Propulsion Laboratory, Pasadena, California, pp. 28-32, May 21, 1974.

for a good orbit determination strategy. Over the years, a small body of knowledge has been collected on the effects of certain poorly modeled parameters, such as spacecraft gas leaks, station location errors and clock errors on deep space navigation measurements. This knowledge, in the form of heuristics developed from flight experience and analysis of simple analytic approximations of tracking measurements, is used by navigation analysts when designing and developing orbit determination strategies. These heuristics are occasionally written down, usually as guidelines gleaned from the analysis of simple models ([12,13] are two examples), but they exist mostly in the minds of the individuals who learned through a great deal of experience how to use them.

## IV. Example 3: Navigation System Design for Mission Operations

This final example describes an activity that begins roughly at the point where preliminary orbit determination strategy design, described in Example 2, leaves off. At this point in mission planning, a set of navigation requirements and a tentative orbit determination strategy, or set of strategies, has been developed as a part of preliminary mission design activities. Both the requirements and the orbit determination strategy chosen to meet them may be modified somewhat during the process of designing the operational navigation system. The end product of this design phase is a detailed navigation plan, specifying the number and type of measurements needed throughout the mission, the model to be used operationally in the orbit determination software (the ODP), and an exhaustive set of computer simulations demonstrating compliance with the requirements over the entire mission (see Mohan and Kirhofer for examples).[4,5]

As in so many other aspects of space navigation, the essence of operational navigation system design is to bring order to a very complex situation. Although there may be similarities among different planetary missions, no two are ever the same; consequently the navigation system designed for each spacecraft must be a special purpose system that is tailored to meet the specific requirements of a particular mission. This implies that mission navigation system design is heuristic in nature, requiring knowledgeable system architects who use their previous experience to provide guidelines which make sense in the context of the problem at hand [3], but who do much more than just follow a set of rules based on previous experience (a normative approach) or use a formula which yields a good system (a rational approach).

Figure 3 shows the functional organization of the navigation system used in the Viking mission to Mars [14], known as the Flight Path Analysis Group (FPAG). This system consisted of people, hardware, and software. The navigation team, which was responsible for carrying out all of these functions, was organized in the same manner as seen in Fig. 3. Each functional block was implemented as a group of people and equipment whose job it was to perform all of the functions in that block. Just as in the architecture of the ODP, form matched function. Notice that the architecture of the navigation system in Fig. 3 has some elements that are common to all deep space missions, such as the tracking data conditioning team, while other elements present are designed to fulfill the specific needs of the Viking mission, such as the lander flight path analysis team. Obviously, one would not expect to find such a group in the Voyager navigation team, whose mission consists solely of planetary flybys. This brief example suggests that operational navigation systems have some components which are unique to a specific mission and others which are very nearly the same across different missions.

While the navigation system for a given mission will not be exactly the same as that of another, there are certain subsystems within the overall navigation system which change very little from mission to mission, as pointed out above. Subsystems that have become formalized structures can be used in virtually all missions with very few changes required. Continuing with the previous example, a subset of the navigation system used for the Viking mission is shown in Fig. 4 (also see [14]). This diagram shows the "flow" of the orbit determination process as envisioned and implemented by the Viking navigation team. This design is the result of many years of flight experience. The majority of Fig. 4 depicts functions performed with the orbit determination software. Since the orbit determination process requires infrequent changes (which may be brought about with the introduction of a new tracking data type, for example), the design of the orbit determination subsystem for a new mission is more of a normative process; it consists of merely arranging the required functions, shown in Fig. 4, in the proper sequence and station in the operations flow.

A good example of the applicability of the orbit determination process (Fig. 4) to a variety of different missions can be found in the DSN Multi-Mission Naviga-

[4] S. N. Mohan, *Magellan Navigation Plan*, Magellan Project Document 630-51 (internal document), Revision B, Jet Propulsion Laboratory, Pasadena, California, March 23, 1988.

[5] *Galileo Navigation Plan*, Galileo Project Document 625-566 (internal document), Revision A, Jet Propulsion Laboratory, Pasadena, California, October 1989.

tion Team's activities. As mentioned earlier, this group supports a large number of both domestic and international missions. The orbit determination operations architecture implemented by the DSN Multi-Mission Navigation Team consists of a basic set of tasks or functions comprising a "generic" mission operations scenario, which looks almost exactly like the diagram in Fig. 4. Although this "generic" architecture is modified slightly to meet the needs of each mission, it is the foundation upon which all mission-specific orbit determination systems used operationally are built.

## V. Summary and Conclusions

Three examples of system design and development used in the field of deep space navigation have been briefly examined to make some assessment of the role of the normative, rational, and heuristic design theories in this area. The examples studied were the design of the JPL orbit determination software system, the design and evaluation of orbit determination strategies, and the design of the operational navigation system and orbit determination subsystem for the Viking mission to Mars. The examples show that all three of the design theories studied are used in some capacity in this field.

One of the primary characteristics of heuristic design theory is that the architect using it must possess a body of expertise that is relevant to the specific context of the system to be designed. By this standard, the examples considered indicate that while the design of certain subsystems can be done using a normative approach to design, not requiring a great deal of expertise from the architect, the design of deep space navigation systems at JPL has been accomplished using a primarily heuristic approach due to the complexity of the problems and the specialized nature of the functions to be performed. There are some rational theories, such as optimal estimation theory, which play an important but supporting role in the design process. The ultimate test of a design theory's effectiveness is the success or failure of the resulting systems; the success of a great number of unmanned planetary missions, from Mariner to Voyager, appears to indicate that the heuristic approach to design has produced deep space navigation systems which have worked very well.

## References

[1] J. F. Jordan, "Deep Space Navigation Systems and Operations," European Space Agency International Symposium on Spacecraft Flight Dynamics, May 1981.

[2] L. J. Wood and J. F. Jordan, "Interplanetary Navigation in the 1980's and 1990's," Paper AAS 81-113, AAS/AIAA Astrodynamics Conference, Lake Tahoe, Nevada, August 1981.

[3] E. Rechtin, *Systems Architecting*, University of Southern California, Los Angeles, California, 1989 (in press).

[4] P. T. Ward and S. J. Mellor, *Structured Development for Real-Time Systems, Vol. 1: Introduction and Tools*, New York: Yourdon Press, 1985.

[5] T. de Marco, *Concise Notes on Software Engineering*, New York: Yourdon Press, 1979.

[6] D. B. Holdridge, *Space Trajectories Program for the IBM 7090 Computer*, Technical Report No. 32–223, Jet Propulsion Laboratory, Pasadena, California, March 2, 1962.

[7] M. R. Warner, M. W. Nead, and R. H. Hudson, *The Orbit Determination Program of the Jet Propulsion Laboratory*, Technical Memorandum No. 33–168, Jet Propulsion Laboratory, Pasadena, California, March 18, 1964.

[8] M. Metcalf, *FORTRAN Optimization*, New York: Academic Press, 1982.

[9] J. D. Anderson, "Trajectory Determination from Observation Data," from *Recent Developments in Space Flight Mechanics*, vol. 9, *American Astronautical Society Science and Technology Series*, pp. 133–158, 1966.

[10] A. Gelb, ed., *Applied Optimal Estimation*, Cambridge, Massachusetts: M.I.T. Press, 1974.

[11] G. W. Null and H. J. Gordon, *The Mariner IV Flight Path and Its Determination From Tracking Data*, Technical Report 32–1108, Jet Propulsion Laboratory, Pasadena, California, pp. 8-17, August 1, 1967.

[12] T. W. Hamilton and W. G. Melbourne, "Information Content of a Single Pass of Doppler Data From a Distant Spacecraft," *JPL Space Programs Summary No. 37–39*, vol. III, Jet Propulsion Laboratory, Pasadena, California, pp. 18–23, May 31, 1966.

[13] J. F. Jordan, G. A. Madrid, and G. E. Pease, "The Effects of Major Error Sources on Planetary Spacecraft Navigation Accuracies," AIAA Paper 70–1077, AAS/AIAA Astrodynamics Conference, Santa Barbara, California, August 19–21, 1970.

[14] W. J. O'Neil, *Viking Navigation*, JPL Publication 78–38, Jet Propulsion Laboratory, Pasadena, California, November 15, 1979.

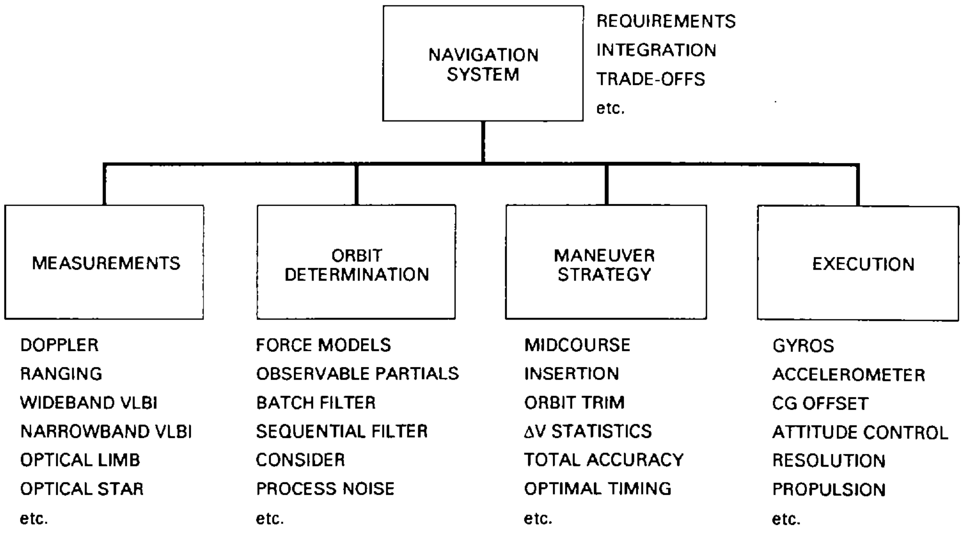


**Fig. 1. Composition of generic navigation system.**

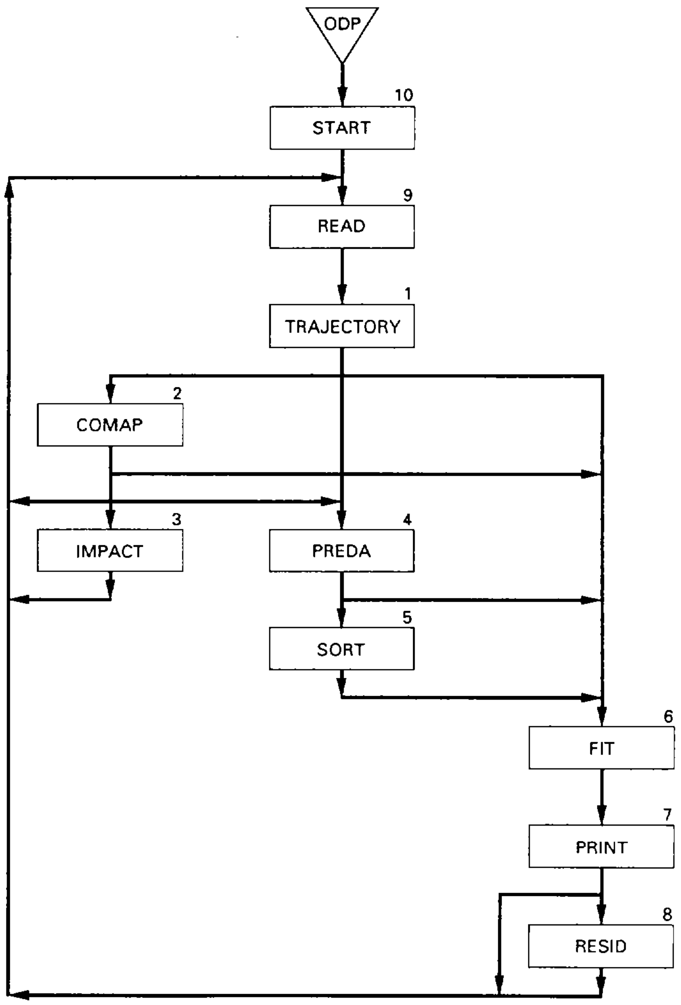
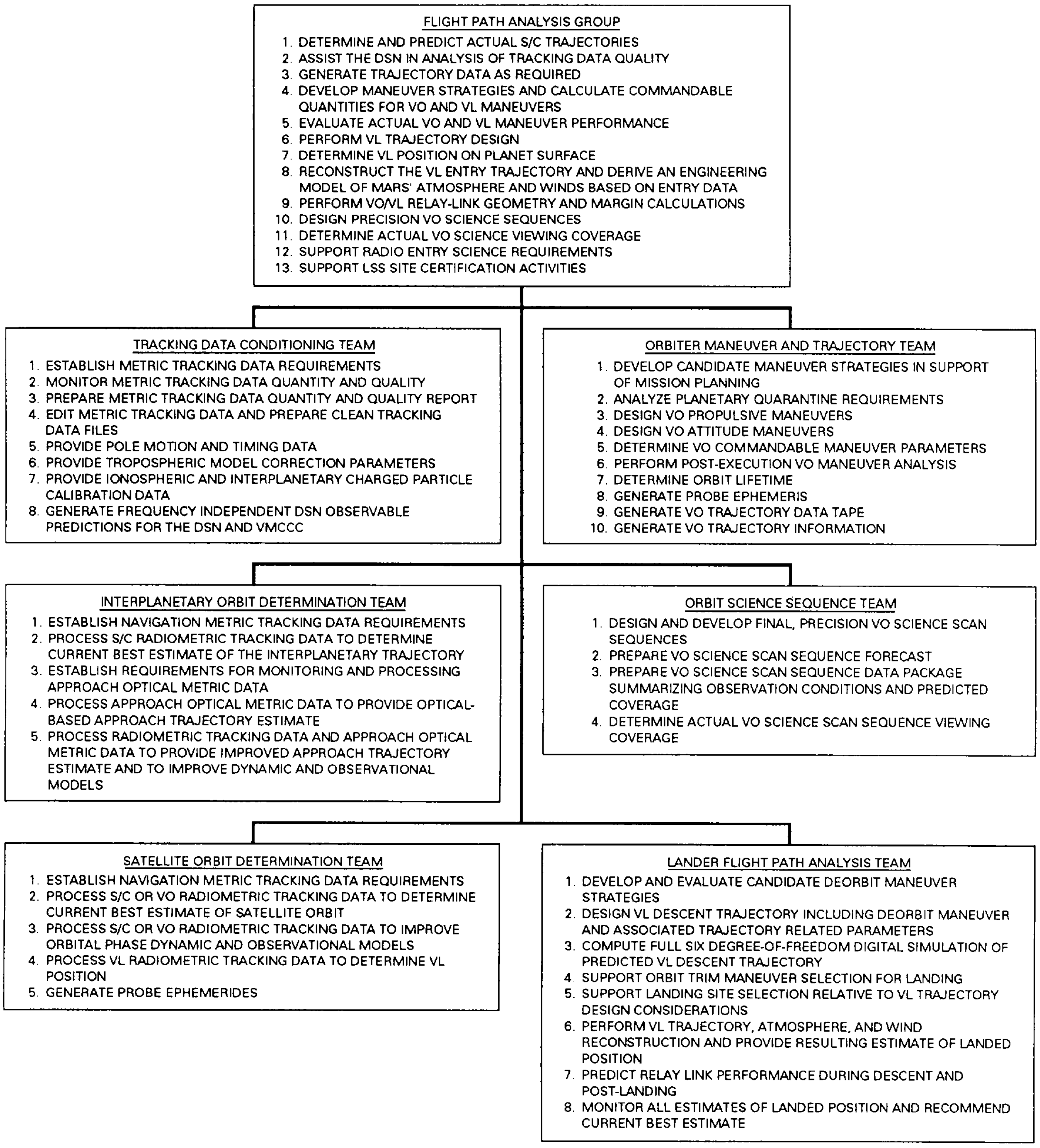


**Fig. 2. High-level view of first-generation ODP structure.**



FLIGHT PATH ANALYSIS GROUP

1. DETERMINE AND PREDICT ACTUAL S/C TRAJECTORIES
2. ASSIST THE DSN IN ANALYSIS OF TRACKING DATA QUALITY
3. GENERATE TRAJECTORY DATA AS REQUIRED
4. DEVELOP MANEUVER STRATEGIES AND CALCULATE COMMANDABLE QUANTITIES FOR VO AND VL MANEUVERS
5. EVALUATE ACTUAL VO AND VL MANEUVER PERFORMANCE
6. PERFORM VL TRAJECTORY DESIGN
7. DETERMINE VL POSITION ON PLANET SURFACE
8. RECONSTRUCT THE VL ENTRY TRAJECTORY AND DERIVE AN ENGINEERING MODEL OF MARS' ATMOSPHERE AND WINDS BASED ON ENTRY DATA
9. PERFORM VO/VL RELAY-LINK GEOMETRY AND MARGIN CALCULATIONS
10. DESIGN PRECISION VO SCIENCE SEQUENCES
11. DETERMINE ACTUAL VO SCIENCE VIEWING COVERAGE
12. SUPPORT RADIO ENTRY SCIENCE REQUIREMENTS
13. SUPPORT LSS SITE CERTIFICATION ACTIVITIES

TRACKING DATA CONDITIONING TEAM

1. ESTABLISH METRIC TRACKING DATA REQUIREMENTS
2. MONITOR METRIC TRACKING DATA QUANTITY AND QUALITY
3. PREPARE METRIC TRACKING DATA QUANTITY AND QUALITY REPORT
4. EDIT METRIC TRACKING DATA AND PREPARE CLEAN TRACKING DATA FILES
5. PROVIDE POLE MOTION AND TIMING DATA
6. PROVIDE TROPOSPHERIC MODEL CORRECTION PARAMETERS
7. PROVIDE IONOSPHERIC AND INTERPLANETARY CHARGED PARTICLE CALIBRATION DATA
8. GENERATE FREQUENCY INDEPENDENT DSN OBSERVABLE PREDICTIONS FOR THE DSN AND VMCCC

ORBITER MANEUVER AND TRAJECTORY TEAM

1. DEVELOP CANDIDATE MANEUVER STRATEGIES IN SUPPORT OF MISSION PLANNING
2. ANALYZE PLANETARY QUARANTINE REQUIREMENTS
3. DESIGN VO PROPULSIVE MANEUVERS
4. DESIGN VO ATTITUDE MANEUVERS
5. DETERMINE VO COMMANDABLE MANEUVER PARAMETERS
6. PERFORM POST-EXECUTION VO MANEUVER ANALYSIS
7. DETERMINE ORBIT LIFETIME
8. GENERATE PROBE EPHEMERIS
9. GENERATE VO TRAJECTORY DATA TAPE
10. GENERATE VO TRAJECTORY INFORMATION

INTERPLANETARY ORBIT DETERMINATION TEAM

1. ESTABLISH NAVIGATION METRIC TRACKING DATA REQUIREMENTS
2. PROCESS S/C RADIOMETRIC TRACKING DATA TO DETERMINE CURRENT BEST ESTIMATE OF THE INTERPLANETARY TRAJECTORY
3. ESTABLISH REQUIREMENTS FOR MONITORING AND PROCESSING APPROACH OPTICAL METRIC DATA
4. PROCESS APPROACH OPTICAL METRIC DATA TO PROVIDE OPTICAL-BASED APPROACH TRAJECTORY ESTIMATE
5. PROCESS RADIOMETRIC TRACKING DATA AND APPROACH OPTICAL METRIC DATA TO PROVIDE IMPROVED APPROACH TRAJECTORY ESTIMATE AND TO IMPROVE DYNAMIC AND OBSERVATIONAL MODELS

ORBIT SCIENCE SEQUENCE TEAM

1. DESIGN AND DEVELOP FINAL, PRECISION VO SCIENCE SCAN SEQUENCES
2. PREPARE VO SCIENCE SCAN SEQUENCE FORECAST
3. PREPARE VO SCIENCE SCAN SEQUENCE DATA PACKAGE SUMMARIZING OBSERVATION CONDITIONS AND PREDICTED COVERAGE
4. DETERMINE ACTUAL VO SCIENCE SCAN SEQUENCE VIEWING COVERAGE

SATELLITE ORBIT DETERMINATION TEAM

1. ESTABLISH NAVIGATION METRIC TRACKING DATA REQUIREMENTS
2. PROCESS S/C OR VO RADIOMETRIC TRACKING DATA TO DETERMINE CURRENT BEST ESTIMATE OF SATELLITE ORBIT
3. PROCESS S/C OR VO RADIOMETRIC TRACKING DATA TO IMPROVE ORBITAL PHASE DYNAMIC AND OBSERVATIONAL MODELS
4. PROCESS VL RADIOMETRIC TRACKING DATA TO DETERMINE VL POSITION
5. GENERATE PROBE EPHEMERIDES

LANDER FLIGHT PATH ANALYSIS TEAM

1. DEVELOP AND EVALUATE CANDIDATE DEORBIT MANEUVER STRATEGIES
2. DESIGN VL DESCENT TRAJECTORY INCLUDING DEORBIT MANEUVER AND ASSOCIATED TRAJECTORY RELATED PARAMETERS
3. COMPUTE FULL SIX DEGREE-OF-FREEDOM DIGITAL SIMULATION OF PREDICTED VL DESCENT TRAJECTORY
4. SUPPORT ORBIT TRIM MANEUVER SELECTION FOR LANDING
5. SUPPORT LANDING SITE SELECTION RELATIVE TO VL TRAJECTORY DESIGN CONSIDERATIONS
6. PERFORM VL TRAJECTORY, ATMOSPHERE, AND WIND RECONSTRUCTION AND PROVIDE RESULTING ESTIMATE OF LANDED POSITION
7. PREDICT RELAY LINK PERFORMANCE DURING DESCENT AND POST-LANDING
8. MONITOR ALL ESTIMATES OF LANDED POSITION AND RECOMMEND CURRENT BEST ESTIMATE

**Fig. 3. Navigation system design for the Viking Mission.**

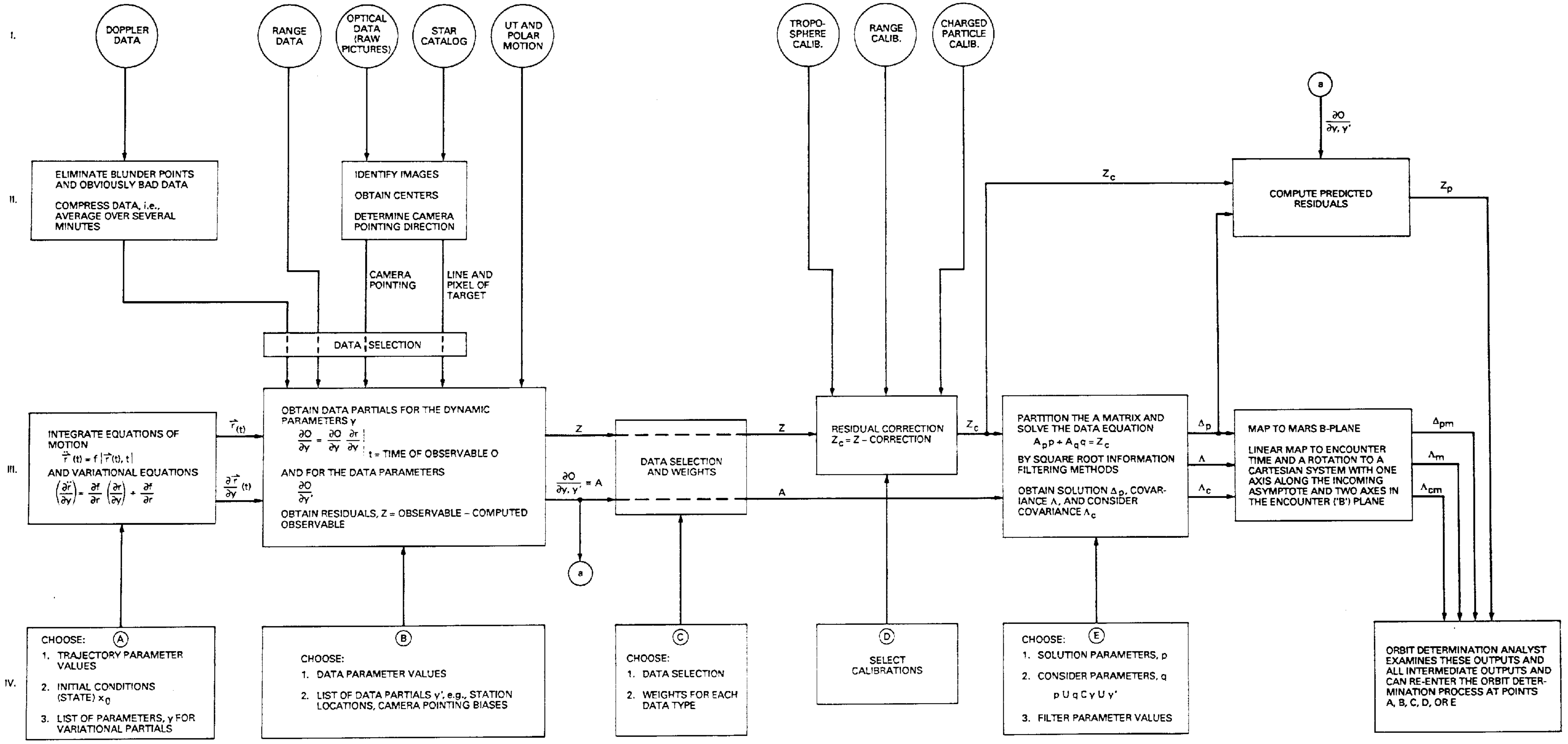

I.
II.
III.
IV.
DOPPLER DATA
RANGE DATA
OPTICAL DATA (RAW PICTURES)
STAR CATALOG
UT AND POLAR MOTION
TROPO-SPHERE CALIB.
RANGE CALIB.
CHARGED PARTICLE CALIB.
ELIMINATE BLUNDER POINTS AND OBVIOUSLY BAD DATA
COMPRESS DATA, i.e., AVERAGE OVER SEVERAL MINUTES
IDENTIFY IMAGES
OBTAIN CENTERS
DETERMINE CAMERA POINTING DIRECTION
CAMERA POINTING
LINE AND PIXEL OF TARGET
DATA SELECTION
INTEGRATE EQUATIONS OF MOTION
$\ddot{\vec{r}}(t) = f\,|\vec{r}(t), t|$
AND VARIATIONAL EQUATIONS
$\left(\frac{\partial \ddot{r}}{\partial y}\right) = \frac{\partial f}{\partial r}\left(\frac{\partial r}{\partial y}\right) + \frac{\partial f}{\partial r}$
$\vec{r}_{(t)}$
$\frac{\partial \vec{r}}{\partial y}(t)$
OBTAIN DATA PARTIALS FOR THE DYNAMIC PARAMETERS y
$\frac{\partial O}{\partial y} = \frac{\partial O}{\partial y}\frac{\partial r}{\partial y}\Big|_{t}$ t = TIME OF OBSERVABLE O
AND FOR THE DATA PARAMETERS
$\frac{\partial O}{\partial y'}$
OBTAIN RESIDUALS, Z = OBSERVABLE – COMPUTED OBSERVABLE
Z
$\frac{\partial O}{\partial y, y'} \equiv A$
DATA SELECTION AND WEIGHTS
Z
A
RESIDUAL CORRECTION
$Z_c = Z - \text{CORRECTION}$
$Z_c$
PARTITION THE A MATRIX AND SOLVE THE DATA EQUATION
$A_p p + A_q q = Z_c$
BY SQUARE ROOT INFORMATION FILTERING METHODS
OBTAIN SOLUTION $\Delta_p$, COVARIANCE $\Lambda$, AND CONSIDER COVARIANCE $\Lambda_c$
$\Delta_p$
$\Lambda$
$\Lambda_c$
MAP TO MARS B-PLANE
LINEAR MAP TO ENCOUNTER TIME AND A ROTATION TO A CARTESIAN SYSTEM WITH ONE AXIS ALONG THE INCOMING ASYMPTOTE AND TWO AXES IN THE ENCOUNTER ('B') PLANE
$\Delta_{pm}$
$\Lambda_m$
$\Lambda_{cm}$
a
$\frac{\partial O}{\partial y, y'}$
$Z_c$
COMPUTE PREDICTED RESIDUALS
$Z_p$
a
CHOOSE: A
1. TRAJECTORY PARAMETER VALUES
2. INITIAL CONDITIONS (STATE) $x_0$
3. LIST OF PARAMETERS, y FOR VARIATIONAL PARTIALS
B
CHOOSE:
1. DATA PARAMETER VALUES
2. LIST OF DATA PARTIALS y', e.g., STATION LOCATIONS, CAMERA POINTING BIASES
C
CHOOSE:
1. DATA SELECTION
2. WEIGHTS FOR EACH DATA TYPE
D
SELECT CALIBRATIONS
CHOOSE: E
1. SOLUTION PARAMETERS, p
2. CONSIDER PARAMETERS, q
$p \cup q \subset y \cup y'$
3. FILTER PARAMETER VALUES
ORBIT DETERMINATION ANALYST EXAMINES THESE OUTPUTS AND ALL INTERMEDIATE OUTPUTS AND CAN RE-ENTER THE ORBIT DETERMINATION PROCESS AT POINTS A, B, C, D, OR E

**Fig. 4. Orbit determination process.**