DEVELOPMENT AND USAGE OF THE ROCKET TRAJECTORY PROGRAM

Barry W. Boehm

August 1965

P-3187

DEVELOPMENT AND USAGE OF THE ROCKET TRAJECTORY PROGRAM

Barry W. Boehm*

The RAND Corporation, Santa Monica, California

PURPOSE OF THE PROGRAM

ROCKET is a digital computer program, written in FORTRAN, which simulates trajectories of aerospace vehicles. It has the following characteristics.

1) Ease of use and of learning to use: Inputs are accepted and outputs produced by ROCKET in the language of the engineer. Thorough documentation of the program and its capabilities has been provided, with numerous examples.†

2) Versatility: The program can simulate the flight of large or small multistage boosters or missiles, satellites, re-entry vehicles, airplanes, or any hybrids thereof, with approximately equal facility. Numerous options are available for specifying the earth model, atmosphere model, aerodynamic effects, propulsion capability, and guidance programs.

3) Flexibility: A clean, well-defined interface is provided between the basic program and the options available to the user, making the addition of new options a smooth and straightforward process.

---

*Any views expressed in this paper are those of the author. They should not be interpreted as reflecting the views of The RAND Corporation or the official opinion or policy of any of its governmental or private research sponsors. Papers are reproduced by The RAND Corporation as a courtesy to members of its staff.

This paper was presented to the Design Automation Working Group, San Francisco, California, June 8, 1965.

†Barry W. Boehm, ROCKET: RAND's Omnibus Calculator of the Kinematics of Earth Trajectories, Prentice-Hall, Inc. Englewood Cliffs, New Jersey, 1964.

PROGRAM DEVELOPMENT AND USAGE

The development of the program in 1961 required six months of full-time effort by one man. After an extensive field-test period at RAND, ROCKET was submitted to SHARE, the IBM users' group, in 1963.* Since then, ROCKET has been used on the Univac 1107 and the CDC 3600 as well as the IBM 704, 7090, 7044, and 7094. Users of the program, presently numbering at least 40, have utilized it mainly for preliminary design studies of aerospace vehicles, tracking systems, and guidance concepts. Other applications have included verification of aerodynamic theories of orbit decay, re-entry and meteor entry, aerodynamic heating analyses, and studies of satellite operational capabilities.

ROCKET PROGRAM CAPABILITIES

Basically, ROCKET numerically integrates the set of differential equations of motion of the specified vehicle. The program automatically calculates such basic aspects as earth gravitational and rotational effects. The user specifies boundary conditions--initial position, velocity,

---

*It may be obtained upon request (enclosing a blank magnetic tape) from

SHARE Distribution Agency
IBM Corporation
112 East Post Road
White Plains, New York

The program's SHARE distribution number is 3001. One does not need to be a user of IBM equipment to request the program.

attitude and weight of the vehicle, and termination conditions for the various sections of the trajectory--on the Flight Control portion of the ROCKET input form. He specifies the propulsive and aerodynamic forces acting along the axes of his vehicle, and the orientation of the vehicle axes, by a sequence of CALLs to standard FORTRAN subroutines which compute the desired propulsion, aerodynamic, and guidance quantities as a function of the vehicle's current state. These CALLs are entered on the Flight Programming portion of the ROCKET input form; they may vary from section to section. Both portions of the ROCKET input form are presented as figures here. The program resolves the forces, divides by the current mass of the vehicle to obtain accelerations, and integrates the accelerations to obtain the vehicle's state at the next point in time. A fourth-order Adams-Moulton method is used for integrations, with a fourth-order Runge-Kutta starting procedure.

Besides tracing the trajectory of the vehicle as a function of time, ROCKET also provides detailed information on its performance during flight. The program can print, at the user's option, statements of the current aerodynamic forces and heating rates, propulsion components, guidance quantities, tracking coordinates and rates for as many as three tracking stations, and osculating orbital elements; it can be modified easily to print any other special quantities desired.

The program can investigate a number of variations on a given trajectory in a single run, greatly simplifying the process of trajectory-matching or of making parameter

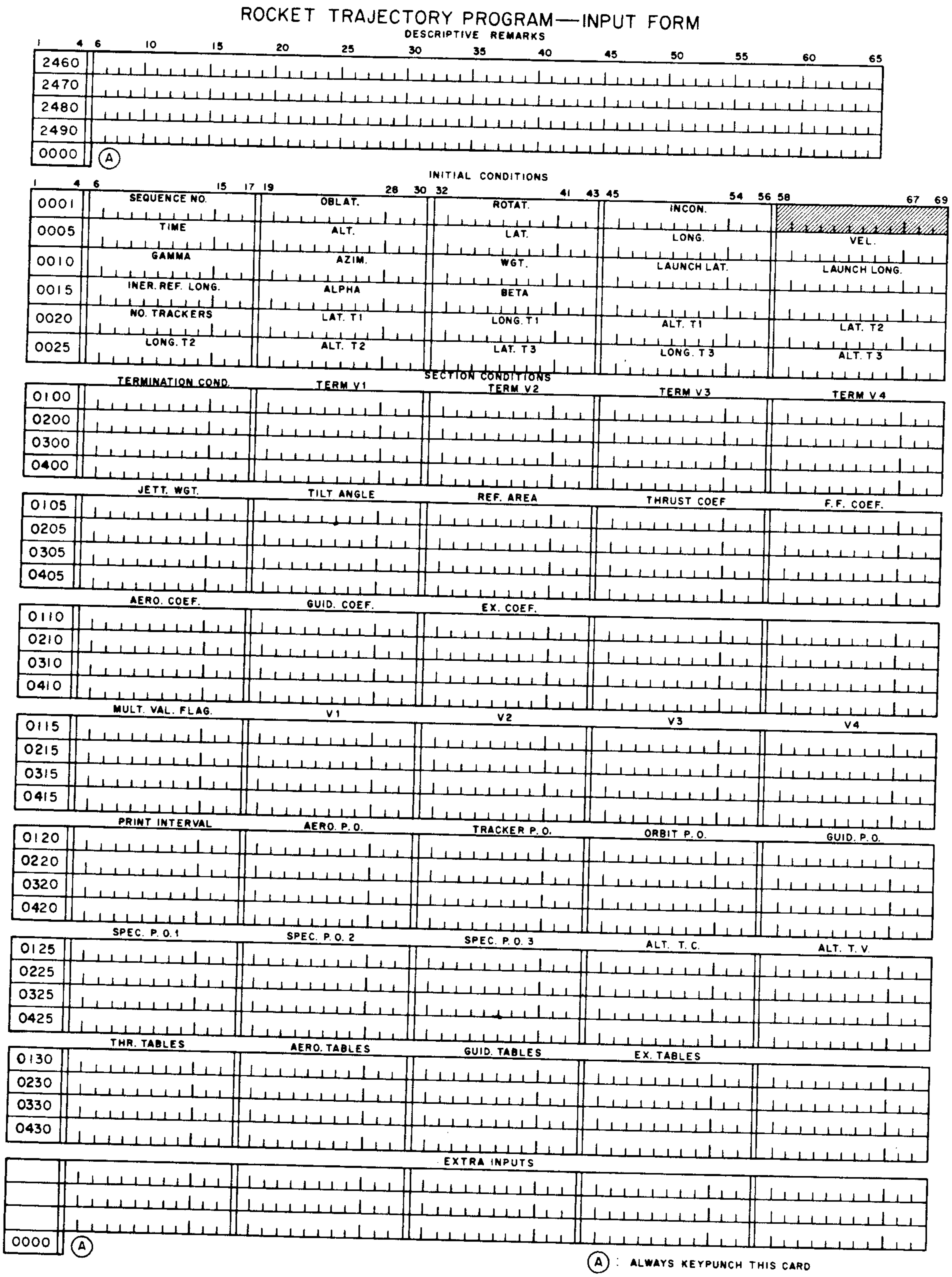

Fig. 1--ROCKET Flight Control Form

C	STATE Nº	C T	STATEMENT

```
      SUBROUTINE SECT1

      CALL
      CALL
      CALL

      CALL
      CALL
      CALL

      RETURN
      END

      SUBROUTINE SECT2

      CALL
      CALL
      CALL

      CALL
      CALL
      CALL

      RETURN
      END

      SUBROUTINE SECT3

      CALL
      CALL
      CALL

      CALL
      CALL
      CALL

      RETURN
      END
```

C	STATE Nº	C T	STATEMENT

```
      SUBROUTINE SECT4

      CALL
      CALL
      CALL

      CALL
      CALL
      CALL

      RETURN
      END
```

Fig. 2--ROCKET Flight Programming Form

studies. Another tool is a recursive scheme by which one may match or maximize an end condition by iterating a section on a control parameter.

REMARKS ON DEVELOPMENT OF GENERAL COMPUTER PROGRAMS

Experience with the ROCKET program suggests the following conclusions for developers of similar large general-purpose computer programs:

1) Use a general programming language which is not tied to a particular machine;
2) Slight gains in efficiency, purchased at the cost of logical simplicity of the program, are a poor bet in the long run;
3) Develop the sections of the program in modular form;
4) Thorough documentation with numerous examples saves everybody's time in the long run;
5) An extensive field-test period for both program and documentation eliminates a lot of embarrasing situations;
6) Anticipate the direction of extensions to the program and provide a clean, well-defined interface for tying them into the program.