Analyzing Dynamics of Cooperating Spacecraft

A software library has been developed to enable high-fidelity computational simulation of the dynamics of multiple spacecraft distributed over a region of outer space and acting with a common purpose. All of the modeling capabilities afforded by this software are available independently in other, separate software systems, but have not previously been brought together in a single system. A user can choose among several dynamical models, many high-fidelity environment models, and several numerical-integration schemes. The user can select whether to use models that assume weak coupling between spacecraft, or strong coupling in the case of feedback control or tethering of spacecraft to each other. For weak coupling, spacecraft orbits are propagated independently, and are synchronized in time by controlling the step size of the integration. For strong coupling, the orbits are integrated simultaneously. Among the integration schemes that the user can choose are Runge-Kutta Verner, Prince-Dormand, Adams-Bashforth-Moulton, and Bulirsh-Stoer. Comparisons of performance are included for both the weak- and strong-coupling dynamical models for all of the numerical integrators. The library was designed for ease of integration with high-fidelity environment models already in use in the Flight Dynamics Analysis Branch, which is one of seven institutional support branches within the Mission Engineering and Systems Analysis Division at Goddard Space Flight Center.

This program was written by Stephen P. Hughes and David C. Folta of **Goddard Space Flight Center** *and Darrel J. Conway of Thinking Systems, Inc. Further information is contained in a TSP (see page 1).*
GSC-14735-1

Spacecraft Attitude Maneuver Planning Using Genetic Algorithms

A key enabling technology that leads to greater spacecraft autonomy is the capability to autonomously and optimally slew the spacecraft from and to different attitudes while operating under a number of celestial and dynamic constraints. The task of finding an attitude trajectory that meets all the constraints is a formidable one, in particular for orbiting or fly-by spacecraft where the constraints and initial and final conditions are of time-varying nature. This approach for attitude path planning makes full use of *a priori* constraint knowledge and is computationally tractable enough to be executed onboard a spacecraft. The approach is based on incorporating the constraints into a cost function and using a Genetic Algorithm to iteratively search for and optimize the solution. This results in a directed random search that explores a large part of the solution space while maintaining the knowledge of good solutions from iteration to iteration. A solution obtained this way may be used 'as is' or as an initial solution to initialize additional deterministic optimization algorithms. A number of representative case examples for time-fixed and time-varying conditions yielded search times that are typically on the order of minutes, thus demonstrating the viability of this method. This approach is applicable to all deep space and planet Earth missions requiring greater spacecraft autonomy, and greatly facilitates navigation and science observation planning.

This work was done by Richard P. Kornfeld of Caltech for **NASA's Jet Propulsion Laboratory**. *Further information is contained in a TSP (see page 1).*

This software is available for commercial licensing. Please contact Don Hart of the California Institute of Technology at (818) 393-3425. Refer to NPO-40107.

Forensic Analysis of Compromised Computers

Directory Tree Analysis File Generator is a Practical Extraction and Reporting Language (PERL) script that simplifies and automates the collection of information for forensic analysis of compromised computer systems. During such an analysis, it is sometimes necessary to collect and analyze information about files on a specific directory tree. Directory Tree Analysis File Generator collects information of this type (except information about directories) and writes it to a text file. In particular, the script asks the user for the root of the directory tree to be processed, the name of the output file, and the number of subtree levels to process. The script then processes the directory tree and puts out the aforementioned text file. The format of the text file is designed to enable the submission of the file as input to a spreadsheet program, wherein the forensic analysis is performed. The analysis usually consists of sorting files and examination of such characteristics of files as ownership, time of creation, and time of most recent access, all of which characteristics are among the data included in the text file.

This program was written by Thomas Wolfe of Caltech for **NASA's Jet Propulsion Laboratory**. *Further information is contained in a TSP (see page 1).*

This software is available for commercial licensing. Please contact Don Hart of the California Institute of Technology at (818) 393-3425. Refer to NPO-40165.

Document Concurrence System

The Document Concurrence System is a combination of software modules for routing users expressions of concurrence with documents. This system enables determination of the current status of concurrences and eliminates the need for the prior practice of manually delivering paper documents to all persons whose approvals were required. This system runs on a server, and participants gain access via personal computers equipped with Web-browser and electronic-mail software. A user can begin a concurrence routing process by logging onto an administration module, naming the approvers and stating the sequence for routing among them, and attaching documents. The server then sends a message to the first person on the list. Upon concurrence by the first person, the system sends a message to the second person, and so forth. A person on the list indicates approval, places the documents on hold, or indicates disapproval, via a Web-based module. When the last person on the list has concurred, a message is sent to the initiator, who can then finalize the process through the administration module. A background process running on the server identifies concurrence processes that are overdue and sends reminders to the appropriate persons.