mode also is used to assess each space shuttle mission to ISS, in support of the CoFR process.

This work was done by Jeffrey Hojnicki, David McKissock, James Fincannon, Robert Green, Thomas Kerslake, Ann Delleur, Jeffrey Follo, Jeffrey Trudell, David J. Hoffman, Anthony Jannette, and Carlos Rodriguez for **Glenn Research Center.** *Further information is contained in a TSP (see page 1).*

Inquiries concerning rights for the commercial use of this invention should be addressed to NASA Glenn Research Center, Commercial Technology Office, Attn: Steve Fedor, Mail Stop 4–8, 21000 Brookpark Road, Cleveland, Ohio 44135. Refer to LEW-17067.

Software for Automation of Real-Time Agents, Version 2

Version 2 of Closed Loop Execution and Recovery (CLEaR) has been developed. The previous version was reported in "Software for Automation of Real-Time Agents" (NPO-21040), *NASA Tech Briefs*, Vol. 26, No. 7 (July 2002), page 34. To recapitulate: CLEaR is an artificial intelligence computer program for use in planning and execution of actions of autonomous agents, including, for example, Deep Space Network (DSN) antenna ground stations, robotic exploratory ground vehicles (rovers), robotic aircraft (UAVs), and robotic spacecraft. CLEaR automates the generation and execution of command sequences, monitoring the sequence execution, and modifying the command sequence in response to execution deviations and failures as well as new goals for the agent to achieve. The development of CLEaR has focused on the unification of planning and execution to increase the ability of the autonomous agent to perform under tight resource and time constraints coupled with uncertainty in how much of resources and time will be required to perform a task. This unification is realized by extending the traditional three-tier robotic control architecture by increasing the interaction between the software components that perform deliberation and reactive functions. The increase in interaction reduces the need to replan, enables earlier detection of the need to replan, and enables replanning to occur before an agent enters a state of failure.

This program was written by Forest Fisher, Tara Estlin, Daniel Gaines, Steve Schaffer, Caroline Chouinard, Barbara Engelhardt, Colette Wilklow, Darren Mutz, Russell Knight, Gregg Rabideau, and Steve Chien of Caltech, and Reid Simmons and David Apfelbaum of CMU 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 Karina Edmonds of the California Institute of Technology at (818) 393-2827. Refer to NPO-30745.

Software for Optimizing Plans Involving Interdependent Goals

A computer program enables construction and optimization of plans for activities that are directed toward achievement of goals that are interdependent. Goal interdependence is defined as the achievement of one or more goals affecting the desirability or priority of achieving one or more other goals. This program is overlaid on the Automated Scheduling and Planning Environment (ASPEN) software system, aspects of which have been described in a number of prior *NASA Tech Briefs* articles. Unlike other known or related planning programs, this program considers interdependences among goals that can change between problems and provides a language for easily specifying such dependences. Specifications of the interdependences can be formulated dynamically and provided to the associated planning software as part of the goal input. Then an optimization algorithm provided by this program enables the planning software to reason about the interdependences and incorporate them into an overall objective function that it uses to rate the quality of a plan under construction and to direct its optimization search. In tests on a series of problems of planning geological experiments by a team of instrumented robotic vehicles (rovers) on new terrain, this program was found to enhance plan quality.

This program was written by Tara Estlin, Daniel Gaines, and Gregg Rabideau 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 Karina Edmonds of the California Institute of Technology at (818) 393-2827. Refer to NPO-30735.

Computing Gravitational Fields of Finite-Sized Bodies

A computer program utilizes the classical theory of gravitation, implemented by means of the finite-element method, to calculate the near gravitational fields of bodies of arbitrary size, shape, and mass distribution. The program was developed for application to a spacecraft and to floating proof masses and associated equipment carried by the spacecraft for detecting gravitational waves. The program can calculate steady or time-dependent gravitational forces, moments, and gradients thereof. Bodies external to a proof mass can be moving around the proof mass and/or deformed under thermoelastic loads. An arbitrarily shaped proof mass is represented by a collection of parallelepiped elements. The gravitational force and moment acting on each parallelepiped element of a proof mass, including those attributable to the self-gravitational field of the proof mass, are computed exactly from the closed-form equation for the gravitational potential of a parallelepiped. The gravitational field of an arbitrary distribution of mass external to a proof mass can be calculated either by summing the fields of suitably many point masses or by higher-order Gauss-Legendre integration over all elements surrounding the proof mass that are part of a finite-element mesh. This computer program is compatible with more general finite-element codes, such as NASTRAN, because it is configured to read a generic input data file, containing the detailed description of the finite-element mesh.

This program was written by Marco Quadrelli 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 Karina Edmonds of the California Institute of Technology at (818) 393-2827.Refer to NPO-40651.

Custom Sky-Image Mosaics From NASA's Information Power Grid

yourSkyG is the second generation of the software described in "yourSky: Custom Sky-Image Mosaics via the Internet" (NPO-30556), *NASA Tech Briefs*, Vol. 27, No. 6 (June 2003), page 45. Like its predecessor, yourSkyG supplies custom astronomical image mosaics of sky regions specified by requesters using client computers connected to the Internet. Whereas yourSky constructs mosaics on a local multiprocessor system, yourSkyG performs the computations on NASA's Information Power Grid (IPG), which is capable of performing much larger mosaicking tasks. (The IPG is high-performance computation and data grid that integrates geographically distributed

computers, databases, and instruments.) A user of yourSkyG can specify parameters describing a mosaic to be constructed. yourSkyG then constructs the mosaic on the IPG and makes it available for downloading by the user. The complexities of determining which input images are required to construct a mosaic, retrieving the required input images from remote sky-survey archives, uploading the images to the computers on the IPG, performing the computations remotely on the Grid, and downloading the resulting mosaic from the Grid are all transparent to the user.

This program was written by Joseph Jacob, James Collier, Loring Craymer, and David Curkendall 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 Karina Edmonds of the California Institute of Technology at (818) 393-2827.Refer to NPO-40761.

ANTLR Tree Grammar Generator and Extensions

A computer program implements two extensions of ANTLR (Another Tool for Language Recognition), which is a set of software tools for translating source codes between different computing languages. ANTLR supports predicated-LL(k) lexer and parser grammars, a notation for annotating parser grammars to direct tree construction, and predicated tree grammars. ["LL(k)" signifies "left-right, leftmost derivation with k tokens of look-ahead," referring to certain characteristics of a grammar.] One of the extensions is a syntax for tree transformations. The other extension is the generation of tree grammars from annotated parser or input tree grammars. These extensions can simplify the process of generating source-to-source language translators and they make possible an approach, called "polyphase parsing," to translation between computing languages. The typical approach to translator development is to identify high-level semantic constructs such as "expressions," "declarations," and "definitions" as fundamental building blocks in the grammar specification used for language recognition. The polyphase approach is to lump ambiguous syntactic constructs during parsing and then disambiguate the alternatives in subsequent tree transformation passes. Polyphase parsing is believed to be useful for generating efficient recognizers for C++ and other languages that, like C++, have significant ambiguities.

This program was written by Loring Craymer 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 Karina Edmonds of the California Institute of Technology at (818) 393-2827. Refer to NPO-30565.

Generic Kalman Filter Software

The Generic Kalman Filter (GKF) software provides a standard basis for the development of application-specific Kalman-filter programs. Historically, Kalman filters have been implemented by customized programs that must be written, coded, and debugged anew for each unique application, then tested and tuned with simulated or actual measurement data. Total development times for typical Kalman-filter application programs have ranged from months to weeks. The GKF software can simplify the development process and reduce the development time by eliminating the need to re-create the fundamental implementation of the Kalman filter for each new application. The GKF software is written in the ANSI C programming language. It contains a generic Kalman-filter-development directory that, in turn, contains a code for a generic Kalman filter function; more specifically, it contains a generically designed and generically coded implementation of linear, linearized, and extended Kalman filtering algorithms, including algorithms for state- and covariance-update and -propagation functions. The mathematical theory that underlies the algorithms is well known and has been reported extensively in the open technical literature. Also contained in the directory are a header file that defines generic Kalman-filter data structures and prototype functions and template versions of application-specific subfunction and calling "navigation/estimation" routine code and headers. Once the user has provided a calling routine and the required application-specific subfunctions, the application-specific Kalman-filter software can be compiled and executed immediately. During execution, the generic Kalman-filter function is called from a higher-level "navigation" or "estimation" routine that preprocesses measurement data and postprocesses output data. The generic Kalman-filter function uses the aforementioned data structures and five implementation-specific subfunctions, which have been developed by the user on the basis of the aforementioned templates. The GKF software can be used to develop many different types of unfactorized Kalman filters. A developer can choose to implement either a linearized or an extended Kalman filter algorithm, without having to modify the GKF software. Control dynamics can be taken into account or neglected in the filter-dynamics model. Filter programs developed by use of the GKF software can be made to propagate equations of motion for linear or nonlinear dynamical systems that are deterministic or stochastic. In addition, filter programs can be made to operate in user-selectable "covariance analysis" and "propagation-only" modes that are useful in design and development stages.

This program was written by Michael E. Lisano II and Edwin Z. Crues of LinCom for **Johnson Space Center**. *For further information, contact the Johnson Technology Transfer Office at (281) 483-3809.*
MSC-23006