Mission Simulation Toolkit
Ames Research Center, Moffett Field, California

The Mission Simulation Toolkit (MST) is a flexible software system for autonomy research. It was developed as part of the Mission Simulation Facility (MSF) project that was started in 2001 to facilitate the development of autonomous planetary robotic missions. Autonomy is a key enabling factor for robotic exploration. There has been a large gap between autonomy software (at the research level), and software that is ready for insertion into near-term space missions. The MST bridges this gap by providing a simulation framework and a suite of tools for supporting research and maturation of autonomy.

Solving Equations of Multibody Dynamics
NASA’s Jet Propulsion Laboratory, Pasadena, California

Darts++ is a computer program for solving the equations of motion of a multibody system or of a multibody model of a dynamic system. It is intended especially for use in dynamical simulations performed in designing and analyzing, and developing software for the control of, complex mechanical systems. Darts++ is based on the Spatial-Operator-Algebra formulation for multibody dynamics. This software reads a description of a multibody system from a model data file, then constructs and implements an efficient algorithm that solves the dynamical equations of the system. The efficiency and, hence, the computational speed is sufficient to make Darts++ suitable for use in real-time closed-loop simulations. Darts++ features an object-oriented software architecture that enables reconfiguration of system topology at run time; in contrast, in related prior software, system topology is fixed during initialization. Darts++ provides an interface to scripting languages, including Tcl and Python, that enable the user to configure and interact with simulation objects at run time.

Mapped Landmark Algorithm for Precision Landing
NASA’s Jet Propulsion Laboratory, Pasadena, California

A report discusses a computer vision algorithm for position estimation to enable precision landing during planetary descent. The Descent Image Motion Estimation System for the Mars Exploration Rovers has been used as a starting point for creating code for precision, terrain-relative navigation during planetary landing. The algorithm is designed to be general because it handles images taken at different scales and resolutions relative to the map, and can produce mapped landmark matches for any planetary terrain of sufficient texture. These matches provide a measurement of horizontal position relative to a known landing site specified on the surface map. Multiple products including dynamic models and terrain databases. Although the communication objects and some of the simulation components that are provided with this toolkit are specifically designed for terrestrial surface rovers, the MST can be applied to any other domain, such as aerial, aquatic, or space.

This project was developed by Gregory Pisanich, Lorenzo Fluenciger, Christian Neukom, Mike Wagner, Eric Buchanan, and Laura Plice of QSS Group, Inc. for Ames Research Center. For further information, access http://opensource.arc.nasa.gov/ or contact the Ames Technology Partnerships Division at (650) 604-2954. Refer to NPO-42890.
mapped landmarks generated per image allow for automatic detection and elimination of bad matches. Attitude and position can be generated from each image; this image-based attitude measurement can be used by the onboard navigation filter to improve the attitude estimate, which will improve the position estimates. The algorithm uses normalized correlation of grayscale images, producing precise, sub-pixel images. The algorithm has been broken into two sub-algorithms: (1) FFT Map Matching (see figure), which matches a single large template by correlation in the frequency domain, and (2) Mapped Landmark Refinement, which matches many small templates by correlation in the spatial domain. Each relies on feature selection, the homography transform, and 3D image correlation. The algorithm is implemented in C++ and is rated at Technology Readiness Level (TRL) 4.

This work was done by Andrew Johnson, Adnan Ansar, and Larry Matthies of Caltech for NASA’s Jet Propulsion Laboratory. Further information is contained in a TSP (see page 1).

The software used in this innovation is available for commercial licensing. Please contact Karina Edmonds of the California Institute of Technology at (626) 395-2322. Refer to NPO-44463.

Web-Based Environment for Maintaining Legacy Software
Lyndon B. Johnson Space Center, Houston, Texas

“Advanced Tool Integration Environment” ("ATIE") is the name of both a software system and a Web-based environment created by the system for maintaining an archive of legacy software and expertise involved in developing the legacy software. ATIE can also be used in modifying legacy software and developing new software. The information that can be encapsulated in ATIE includes experts’ documentation, input and output data of tests cases, source code, and compilation scripts. All of this information is available within a common environment and retained in a database for case of access and recovery by use of powerful search engines. ATIE also accommodates the embedment of supporting software that users require for their work, and even enables access to supporting commercial-off-the-shelf (COTS) software within the flow of the experts’ work. The flow of work can be captured by saving the sequence of computer programs that the expert uses. A user gains access to ATIE via a Web browser. A modern Web-based graphical user interface promotes efficiency in the retrieval, execution, and modification of legacy code. Thus, ATIE saves time and money in the support of new and pre-existing programs.

This program was written by Michael Tiggies of Johnson Space Center; Nelson Thompson, Mark Orr, and Richard Fox of Dynacs, Inc.; and Rich Rohan of Lockheed Martin Corp. Further information is contained in a TSP (see page 1). MSC-23810-1

Information Metacatalog for a Grid
Ames Research Center, Moffett Field, California

SWIM is a Software Information Metacatalog that gathers detailed information about the software components and packages installed on a grid resource. Information is currently gathered for Executable and Linking Format (ELF) executables and shared libraries, Java classes, shell scripts, and Perl and Python modules. SWIM is built on top of the POUR framework, which is described in the preceding article. SWIM consists of a set of Perl modules for extracting software information from a system, an XML schema defining the format of data that can be added by users, and a POUR XML configuration file that describes how these elements are used to generate pe-