for surface, aerial, space, and water robotic vehicle visualization during field testing. It has been used to enable operations of wheeled and legged surface vehicles, and can be readily adapted to facilitate other mechanical mobility solutions.

The 3D visualization can render an articulated 3D model of a robotic platform for any environment. Given the model, the software receives real-time telemetry from the avionics system onboard the vehicle and animates the robot visualization to reflect the telemetered physical state. This is used to track the position and attitude in real time to monitor the progress of the vehicle as it traverses its environment. It is also used to monitor the state of any or all articulated elements of the vehicle, such as arms, legs, or control surfaces. The visualization can also render other sorts of telemetered states visually, such as stress or strains that are measured by the avionics. Such data can be used to color or annotate the virtual vehicle to indicate nominal or off-nominal states during operation.

The visualization is also able to render the simulated environment where the vehicle is operating. For surface and aerial vehicles, it can render the terrain under the vehicle as the avionics sends it location information (GPS, odometry, or star tracking), and locate the vehicle over or on the terrain correctly. For long traverses over terrain, the visualization can stream in terrain piecewise in order to maintain the current area of interest for the operator without incurring unreasonable resource constraints on the computing platform. The visualization software is designed to run on laptops that can operate in field-testing environments without Internet access, which is a frequently encountered situation when testing in remote locations that simulate planetary environments such as Mars and other planetary bodies.

This work was done by Mark W. Powell, Recaredo J. Torres, David S. Mittman, James A. Kurien, and Lucy Abramyan 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 Daniel Broderick of the California Institute of Technology at danielb@caltech.edu. Refer to NPO-47945.

Athena

NASA's Jet Propulsion Laboratory, Pasadena, California

The Athena simulation software supports an analyst from DoD or other federal agency in making stability and reconstruction projections for operational analyses in areas like Iraq or Afghanistan. It encompasses the use of all elements of national power: diplomatic, information, military, and economic (DIME), and anticipates their effects on political, military, economic, social, information, and infrastructure (PMESII) variables in real-world battle space environments. Athena is a stand-alone model that provides analysts with insights into the effectiveness of complex operations by anticipating second-, third-, and higher-order effects. For example, the first-order effect of executing a curfew may be to reduce insurgent activity, but it may also reduce consumer spending and keep workers home as second-order effects. Reduced spending and reduced labor may reduce the gross domestic product (GDP) as a third-order effect. Damage to the economy will have further consequences.

The Athena approach has also been considered for application in studies related to climate change and the smart grid. It can be applied to any project where the impacts on the population and their perceptions are important, and where population perception is important to the success of the project.

This work was done by Robert G. Chamberlain, William H. Duquette, Joseph P. Provenzano, and Theodore J. Brunzie of Caltech, and Benjamin Jordan of the U.S. Army for NASA's Jet Propulsion Laboratory. For more information, contact iaoffice@jpl.nasa.gov.

This software is available for commercial licensing. Please contact Daniel Broderick of the California Institute of Technology at danielb@caltech.edu. Refer to NPO-47857.

In Situ Surface Characterization

NASA's Jet Propulsion Laboratory, Pasadena, California

Operation of *in situ* space assets, such as rovers and landers, requires operators to acquire a thorough understanding of the environment surrounding the spacecraft. The following programs help with that understanding by providing higher-level information characterizing the surface, which is not immediately obvious by just looking at the XYZ terrain data.

This software suite covers three primary programs: marsuvw, marsrough, and marsslope, and two secondary programs, which together use XYZ data derived from *in situ* stereo imagery to characterize the surface by determining surface normal, surface roughness, and various aspects of local slope, respectively.

These programs all use the Planetary Image Geometry (PIG) library to read mission-specific data files. The programs themselves are completely multimission; all mission dependencies are handled by PIG. The input data consists of images containing XYZ locations as derived by, e.g., marsxyz.

The marsuvw program determines surface normals from XYZ data by gathering XYZ points from an area around each pixel and fitting a plane to those points. Outliers are rejected, and various consistency checks are applied. The result shows the orientation of the local surface at each point as a unit vector. The program can be run in two modes: standard, which is typically used for *in situ* arm work, and slope, which is typically used for rover mobility. The difference is primarily due to optimizations necessary for the larger patch sizes in the slope case.

The marsrough program determines surface roughness in a small area

around each pixel, which is defined as the maximum peak-to-peak deviation from the plane perpendicular to the surface normal at that pixel.

The marsslope program takes a surface normal file as input and derives one of several slope-like outputs from it. The outputs include slope, slope rover direction (a measure of slope radially away from the rover), slope heading, slope magnitude, northerly tilt, and solar energy (compares the slope with the Sun's location at local noon).

The marsuvwproj program projects a surface normal onto an arbitrary plane in space, resulting in a normalized 3D vector, which is constrained to lie in the plane. The marsuvwrot program rotates the vectors in a surface normal file, generating a new surface normal file. It also can change coordinate systems for an existing surface normal file.

While the algorithms behind this suite are not particularly unique, what makes the programs useful is their integration into the larger *in situ* image processing system via the PIG library. They work directly with space *in situ* data, understanding the appropriate image metadata fields and updating them properly.

The secondary programs (marsuvwproj, marsuvwrot) were originally developed to deal with anomalous situations on Opportunity and Spirit, respectively, but may have more general applicability.

This work was done by Robert G. Deen, Patrick C. Leger, and Igor Yanovsky 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 Daniel Broderick of the California Institute of Technology at danielb@caltech.edu. Refer to NPO-47724.

Ndarts

NASA's Jet Propulsion Laboratory, Pasadena, California

Ndarts software provides algorithms for computing quantities associated with the dynamics of articulated, rigid-link, multibody systems. It is designed as a general-purpose dynamics library that can be used for the modeling of robotic platforms, space vehicles, molecular dynamics, and other such applications. The architecture and algorithms in Ndarts are based on the Spatial Operator Algebra (SOA) theory for computational multibody and robot dynamics developed at JPL. It uses minimal, internal coordinate models. The algorithms are low-order, recursive scatter/gather algorithms.

In comparison with the earlier Darts++ software, this version has a more general and cleaner design needed to support a larger class of computational dynamics needs. It includes a frames infrastructure, allows algorithms to operate on subgraphs of the system, and implements lazy and deferred computation for better efficiency.

Dynamics modeling modules such as Ndarts are core building blocks of control and simulation software for space, robotic, mechanism, bio-molecular, and material systems modeling.

This work was done by Abhinandan Jain of Caltech for NASA's Jet Propulsion Laboratory. For more information, contact iaoffice@jpl.nasa.gov.

This software is available for commercial licensing. Please contact Daniel Broderick of the California Institute of Technology at danielb@caltech.edu. Refer to NPO-47703.