ing occurs during scheduled periods every day of the week, the ACR software continuously monitors the antenna equipment.

This work was done by Roger Y. Chao, Scott C. Morgan, Martha M. Strain, Stephen T. Rockwell, Kenneth J. Shimizu, Barzia J. Tehrani, Jaclyn H. Kwok, Michelle Tuazon-Wong, and Henry Valtier of Caltech; Reza Nalbandi of MTC; Michael Wert of ITT; and Patrick Leung of ISDS/Averstar 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-47002.

Efficient Parallel Engineering Computing on Linux Workstations

NASA's Jet Propulsion Laboratory, Pasadena, California

A C software module has been developed that creates lightweight processes (LWPs) dynamically to achieve parallel computing performance in a variety of engineering simulation and analysis applications to support NASA and DoD project tasks. The required interface between the module and the application it supports is simple, minimal and almost completely transparent to the user applications, and it can achieve nearly ideal computing speed-up on multi-CPU engineering workstations of all operating system platforms. The module can be integrated into an existing application (C, C++, Fortran and others) either as part of a compiled module or as a dynamically linked library (DLL).

This software has the following major advantages over existing commercial and public domain software of similar functionality.

1. It is especially applicable to and powerful on commercially, widely available, multi-CPU engineering workstations;
2. It has a very simple software architecture and user interface and can be quickly integrated into an existing application; and
3. Its code size is very small, and its performance overhead is minimal, resulting in nearly ideal parallel-computing performance for many computing-intensive scientific and engineering applications.

The approach adopted in this technology development does not require any additional hardware and software beyond what's typically available on any commercial engineering workstations, that is a native operating system and C, C++ or FORTRAN compilers that an application needs.

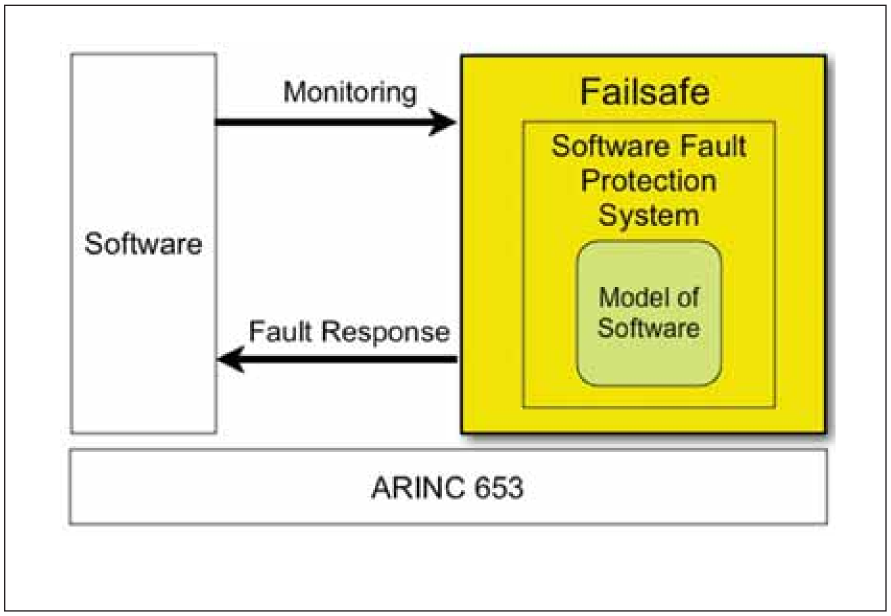
This work was done by John Z. Lou of Caltech for NASA's Jet Propulsion Laboratory. For more information, contact iaoffice@jpl.nasa.gov.

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-46892.

FAILSAFE Health Management for Embedded Systems

NASA's Jet Propulsion Laboratory, Pasadena, California

The FAILSAFE project is developing concepts and prototype implementations for software health management in mission-critical, real-time embedded systems. The project unites features of the industry-standard ARINC 653 Avionics Application Software Standard Interface and JPL's Mission Data System (MDS) technology (see figure). The ARINC 653 standard establishes requirements for the services provided by partitioned, real-time operating systems. The MDS technology provides a state analysis method, canonical architecture, and software framework that facilitates the design and implementation of software-intensive complex systems. The MDS technology has been used to provide the health management function for an ARINC 653 application implementation. In particular, the focus is on showing how this combination enables reasoning about, and recovering from, application software problems.



The **FAILSAFE** model-based health management concept is depicted in the block diagram.

The application itself consists of two unique applications running in the ARINC 653 system: a target application and the FAILSAFE model-based health monitoring application. The target application is a high-level simulation of the Shuttle Abort Control System (ACS), developed specifically for this task. The target application is a two-partition application with one partition allocated to the sequencing behavior, and one partition allocated to the application I/O. The health monitor application executes in its own partition. The three application partitions communicate via ARINC 653 ports and message queues, which are specified in the system module.xml configuration file. Real-time system data is provided to the health monitor via the use of ARINC 653 sampling ports that allows the health monitor application to intercept any traffic coming across the ports of interest.

This task was turned into a goal-based function that, when working in concert with the software health manager, aims to work around software and hardware problems in order to maximize abort performance results. In order to make it a compelling demonstration for current aerospace initiatives, the prototype has been additionally imposed on a number of requirements derived from NASA's Constellation Program.

Lastly, the ARINC 653 standard imposes a number of requirements on the system integrator for developing the requisite error handler process. Under ARINC 653, the health monitoring (HM) service is invoked by an application calling the application error service, or by the operating system or hardware detecting a fault. It is these HM and error process details that are implemented with the MDS technology, showing how a static-analytic approach is appropriate for identifying fault determination details, and showing how the framework supports acting upon state estimation and control features in order to achieve safety-related goals.

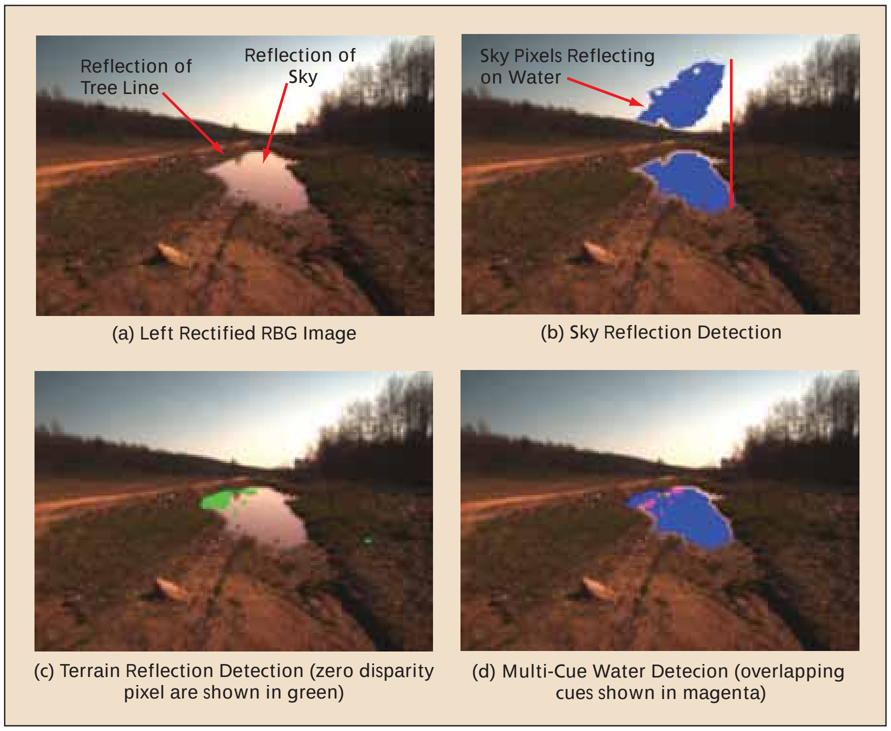
This work was done by Gregory A. Horvath, David A. Wagner, and Hui Ying Wen of Caltech and Matthew Barry of Kestrel Technology for NASA's Jet Propulsion Laboratory. For more information, contact iaoffice@jpl.nasa.gov.

This software is available for commercial licensing. Please contact Daniel Broderick of the California Institute of Technology at danielb@caltech.edu. Refer to NPO-46981.

Water Detection Based on Sky Reflections

NASA's Jet Propulsion Laboratory, Pasadena, California

This software has been designed to detect water bodies that are out in the open on cross-country terrain at mid- to far-range (approximately 20–100 meters), using imagery acquired from a stereo pair of color cameras mounted on a terrestrial, unmanned ground vehicle (UGV). Non-traversable water bodies, such as large puddles, ponds, and lakes, are indirectly detected by detecting reflections of the sky below the horizon in color imagery. The appearance of water bodies in color imagery largely depends on the ratio of light reflected off the water surface to the light coming out of the water body. When a water body is far away, the angle of incidence is large, and the light reflected off the water surface dominates. We have exploited this behavior to detect water bodies out in the open at mid- to far-range. When a water body is detected at far range, a UGV's path planner can begin to look for alternate routes to the goal position sooner, rather than later. As a result, detecting water hazards at far range generally reduces the time required to reach a goal position during autonomous navigation. This software implements a new water detector based on sky reflections that geometrically locates the exact pixel in the sky that is reflecting on a candidate water pixel on the ground, and predicts if the ground pixel is water based on color similarity and local terrain features (see figure).


Reflection of Tree Line
Reflection of Sky
(a) Left Rectified RBG Image
Sky Pixels Reflecting on Water
(b) Sky Reflection Detection
(c) Terrain Reflection Detection (zero disparity pixel are shown in green)
(d) Multi-Cue Water Detecion (overlapping cues shown in magenta)

Water is Detected indirectly by detecting sky and terrain reflections.

Assuming a water body can be modeled as a horizontal mirror, a ray of incident light reflected off the surface of a water body enters a pixel of a camera's focal plane array (FPA). Since the angle of incidence is equal to the angle of reflection (according to the law of reflection), a direct ray from the tail of the incident ray (and within the same vertical plane as the incident ray) will enter the camera's FPA at a pixel whose color will indicate the color of the sky being reflected along the reflected ray. Because the distance between the camera and the sky is much larger than the distance between the camera and candidate water points at normal detection ranges, the direct ray and the incident