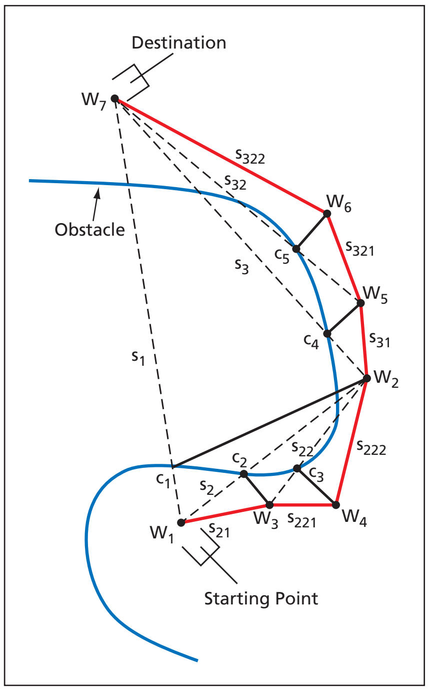
ented bounding boxes (OBBs), in which an object is represented approximately, for computational purposes, by a box that encloses its outer boundary. Because many parts of a robotic manipulator are cylindrical, the OBB method has been extended in this method to enable the approximate representation of cylindrical parts by use of octagonal or other multiple-OBB assemblies denoted oriented bounding prisms (OBPs).

A multiresolution OBB/OBP representation of the robot and its manipulator arm and a multiresolution OBB representation of external objects (including terrain) are constructed and used in a process in which collisions at successively finer resolutions are detected through computational detection of overlaps between the corresponding OBB and OBP models. For computational efficiency, the process is started at the coarsest resolution and stopped as soon as possible, preferably before reaching the finest resolution. At the coarsest resolution, there is a single OBB enclosing all relevant external objects and a single OBB enclosing the entire robot. At the next finer level of resolution, the coarsest-resolution OBB is divided into two OBBs, and so forth. If no collision is detected at the coarsest resolution, then there is no need for further computation to detect collisions. If a collision is detected at the coarsest resolution, then tests for collisions are performed at the next finer level of resolution. This process is continued to successively finer resolutions until either no more collisions are detected or the finest resolution is reached.

The path-planning algorithm operates on a representation of the robot arm and obstacles in a Cartesian coordinate system. The figure schematically depicts a simplified example of the geometric effects of the algorithm. In this example, the robot arm has been commanded to move from a starting point to a destination. The problem to be solved by the algorithm is to choose waypoints (W_1, W_2, ...) and straight-line path segments connecting the waypoints (including the starting point and destination as waypoints) so that there is no collision along any segment. The algorithm can be summarized as follows:

1. Generate a straight-line path (s_1) from the starting point (W_1) to the destination.
2. Using the collision-detection method described above, test for collisions along this path.
3. If there is a collision (denoted by collision point c_1), then by use of a geometry-based subalgorithm too complex to be described within the space available for this article, generate two new sub-paths (s_2 and s_3) that connect a new waypoint (W_2) with the ends of s_1.
4. Test each new sub-path for collisions.
5. If a collision is detected on either sub-path (e.g., at collision point c_2 on s_2), then in the manner of step 3, generate new sub-sub paths (s_{21} and s_{22}) that connect new way point W_3 with W_1 and W_2.
6. Test for collisions and generate new path segments in the manner described above until the starting and destination points are connected by collision-free path segments. In this example, the result is a total of seven waypoints connected by six path segments.



A **Multi-Segment Path** is generated in an iterative process of generating candidate segments and testing them for collisions.

This work was done by Paul Backes and Antonio Diaz-Calderon of Caltech for NASA's Jet Propulsion Laboratory.

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-41697

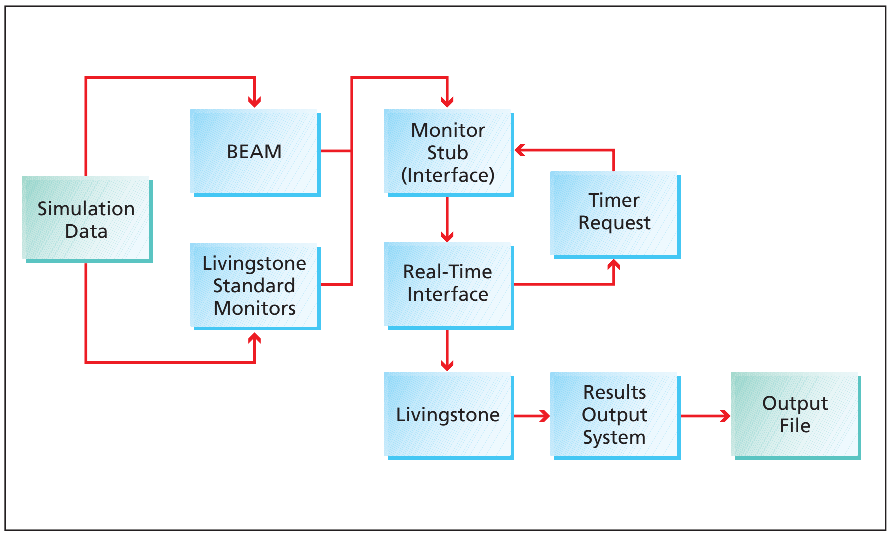
Hybrid Automated Diagnosis of Discrete/Continuous Systems

Integration of complementary tools offers new approach to hybrid diagnosis.

NASA's Jet Propulsion Laboratory, Pasadena, California

A recently conceived method of automated diagnosis of a complex electromechanical system affords a complete set of capabilities for hybrid diagnosis in the case in which the state of the electromechanical system is characterized by both continuous and discrete values (as represented by analog and digital signals, respectively). The method is an integration of two complementary diagnostic systems: (1) beacon-based exception analysis for multimissions (BEAM), which is primarily useful in the continuous domain and easily performs diagnoses in the presence of transients; and (2) Livingstone, which is primarily useful in the discrete domain and is typically restricted to quasi-steady conditions. BEAM has been described in several prior *NASA Tech Briefs* articles: "Software for Autonomous Diagnosis of Complex Systems" (NPO-20803), Vol. 26, No. 3 (March 2002), page 33; "Beacon-Based Exception Analysis for Multimissions" (NPO-20827), Vol. 26, No. 9 (September 2002), page 32; "Wavelet-Based Real-Time Diagnosis of Complex Systems" (NPO-20830), Vol. 27, No. 1 (January 2003), page 67; and "Integrated Formulation of Beacon-Based Exception Analysis for Multimissions" (NPO-21126), Vol. 27, No. 3 (March 2003), page 74.

Briefly, BEAM is a complete data-analysis method, implemented in software, for real-time or off-line detection and characterization of faults. The basic premise of BEAM is to characterize a system from all available observations and train the characterization with respect to normal phases of operation. The observations are primarily continuous in nature. BEAM isolates anomalies by analyzing the deviations from nominal for each phase of operation. Livingstone is a



This **Data-Flow Diagram** illustrates the role of BEAM as a "smart" monitor for Livingstone.

model-based reasoner that uses a model of a system, controller commands, and sensor observations to track the system's state, and detect and diagnose faults. Livingstone models a system within the discrete domain. Therefore, continuous sensor readings, as well as time, must be discretized. To reason about continuous systems, Livingstone uses "monitors" that discretize the sensor readings using trending and thresholding techniques.

In development of the a hybrid method, BEAM results were sent to Livingstone to serve as an independent source of evidence that is in addition to the evidence gathered by Livingstone standard monitors. The figure depicts the flow of data in an early version of a hybrid system dedicated to diagnosing a simulated electromechanical system. In effect, BEAM served as a "smart" monitor for Livingstone. BEAM read the simulation data, processed the data to form observations, and stored the observations in a file. A monitor stub synchronized the events recorded by BEAM with the output of the Livingstone standard monitors according to time tags. This information was fed to a real-time interface, which buffered and fed the information to Livingstone, and requested diagnoses at the appropriate times. In a test, the hybrid system was found to correctly identify a failed component in an electromechanical system for which neither BEAM nor Livingstone alone yielded the correct diagnosis.

This work was done by Han Park, Mark James, Ryan Mackey of Caltech; Howard Cannon and Anapa Bajwa of NASA's Ames Research Center; and William Maul of NASA's Glenn Research Center for NASA's Jet Propulsion Laboratory.

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-40910.

State-Estimation Algorithm Based on Computer Vision

Available data are utilized optimally without incurring an excessive computational burden.

NASA's Jet Propulsion Laboratory, Pasadena, California

An algorithm and software to implement the algorithm are being developed as means to estimate the state (that is, the position and velocity) of an autonomous vehicle, relative to a visible nearby target object, to provide guidance for maneuvering the vehicle. In the original intended application, the autonomous vehicle would be a spacecraft and the nearby object would be a small astronomical body (typically, a comet or asteroid) to be explored by the spacecraft. The algorithm could also be used on Earth in analogous applications — for example, for guiding underwater robots near such objects of interest as sunken ships, mineral deposits, or submerged mines.

For the purpose of the algorithm, it is assumed that the robot would be equipped with a vision system that would include one or more electronic cameras, image-digitizing circuitry, and an image-data-processing computer that would generate feature-recognition data products. Such products customarily include bearing angles of lines of sight from the camera(s) [and, hence, from the vehicle] to recognized features. The data products that are processed by the present algorithm are of the following types:

- The Cartesian vector from the camera to a reference point on or in the target body;
- Bearing angles from the camera to the reference point;
- A landmark table (LMT);
- A paired-feature table (PFT); and
- A range point table (RPT).

The incorporation of the LMT and PFT is particularly important. LMT and PFT data are generated by typical computer-vision systems that could be used in the contemplated applications. In an LMT, a vision system recognizes landmarks from an onboard catalog and reports their bearing angles and associated known locations on the target body. In a PFT, a vision system reports bearing angles to features recognized as being common to two images taken at different times. Relative to the LMT, the PFT can be generated with less computation because it is necessary only to track features frame-to-frame; it is not necessary to associate the features with landmarks. However, it is more challenging to incorporate the PFT in a state-estimation algorithm for reasons discussed below. The LMT and PFT are complementary in the sense that the LMT provides position-type information while the PFT provides velocity-type information. However, the velocity-type information from the PFT is incomplete because it includes an unknown scale factor. A state-estimation algorithm must fuse the aforementioned data types to make an optimal estimate.

The following three main challenges arise as parts of this data-fusion problem:

- The first challenge is posed by the large number of features (typically ≥50) that a typical computerized vision system can recognize during any given frame period. The large number of features imposes a heavy burden for real-time computation.