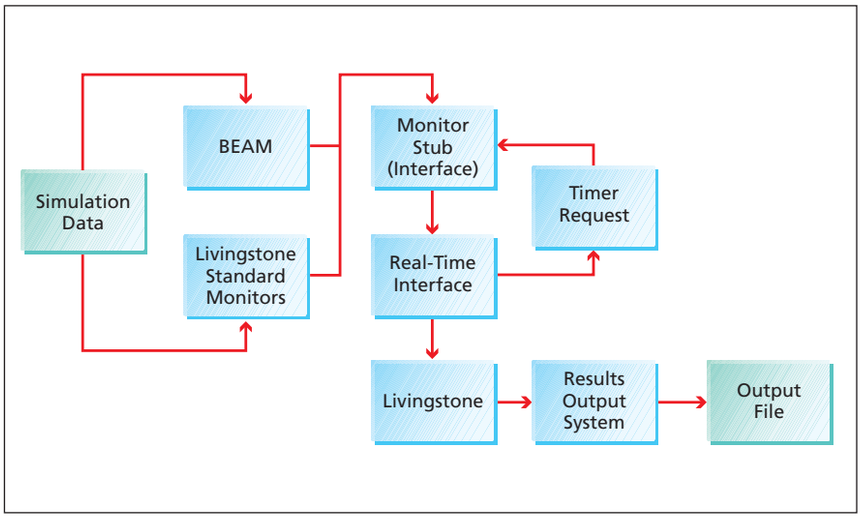


This **Data-Flow Diagram** illustrates the role of BEAM as a "smart" monitor for Livingstone.

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.

- The second challenge is associated with the lack of range information when camera measurements are the only measurements available. Camera's measurements consist only of bearings to specific feature points in images. The PFT data type is especially challenging inasmuch as recognized features do not necessarily represent known objects and do not contain location information.
- The third challenge is posed by the fact that computer vision information often relates to images taken in the past. For example, the PFT data type reports features that were recognized as being common to two images taken at earlier times. The need to update the current state estimate by use of information from the past presents a challenge because prior recursive state-estimating algorithms typically only propagate the current state.

The present algorithm addresses these challenges by incorporating the following innovations:

The first innovation is a preprocessing step, based on QR factorization (a particular matrix factorization, a description of which would exceed the scope of this article), that provides for optimal compression of LMT, PFT, and RPT updates that involve large numbers of recognized features. This compression eliminates the need for a considerable amount of real-time computation.

The second innovation is a mathematical annihilation method for forming a linear measurement equation from the PFT data. The annihilation method is equivalent to a mathematical projection that eliminates the dependence on the unknown scale factor.

The third innovation is a state-augmentation method for handling PFT and other data types that relate states from two or more past instants of time. The state-augmentation method stands in contrast to a prior stochastic cloning method. State augmentation provides an optimal solution to the state-estimation problem, while stochastic cloning can be shown to be suboptimal.

This work was done by David Bayard and Paul Brugarolas 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-41321

Representing Functions in *n* Dimensions to Arbitrary Accuracy

Computation can be simplified in cases in which data are noiseless.

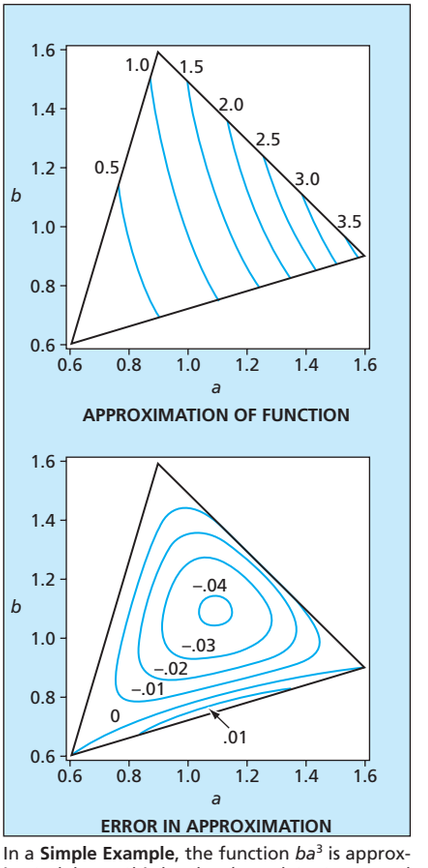
Langley Research Center, Hampton, Virginia

A method of approximating a scalar function of n independent variables (where n is a positive integer) to arbitrary accuracy has been developed. This method is expected to be attractive for use in engineering computations in which it is necessary to link global models with local ones or in which it is necessary to interpolate noiseless tabular data that have been computed from analytic functions or numerical models in n-dimensional spaces of design parameters.

This method is related to prior statistically based methods of fitting low-order approximate functional representations (response surfaces) to noisy experimental data. The prior methods are advantageous in situations in which large amounts of noisy data are available, but in situations in which the data and the functions that they represent are noiseless, it is computationally inefficient to generate the large quantities of data needed for fitting. Moreover, in the prior statistically based methods, the low-order functional representation cannot be defined to a specified degree of accuracy. The latter shortcoming limits the usefulness of response surfaces in design-optimization computations because (1) optimization calculations involve gradients of functions, which are approximated to orders lower than those of the functions themselves and, hence, can be so inaccurate as to yield poor results; and (2) the accuracy of a response surface can vary widely over its domain. The present method overcomes these shortcomings of the prior methods.

Increasingly, modern computational-simulation programs generate values of gradients of functions in addition to values of the functions themselves, in order to satisfy the need for accurate gradient as well as function values for optimizations. Taking advantage of this trend, the present method relies on the availability of both gradient and function data. In this method, the space of n independent variables is subdivided into an n-dimensional mesh of simplex elements (simplices) that amount to n-dimensional generalizations of modeling techniques used in the finite-element method. The exact values of the scalar function and its gradient, as generated by the applicable computational model, are specified at the simplex nodes, which are intersections of coordinate axes of the n-dimensional mesh. Within each simplex, the function and its gradient are interpolated approximately by a set of basis functions of the n coordinates.

In order to minimize the computational burden, one tries to use basis functions of order no higher than that needed to limit the error in the approximation to an acceptably low value. It would be preferable if, in a given case,



In a **Simple Example**, the function ba^3 is approximated by a third-order (complete to second order) polynomial on a triangular simplex. The error is zero at the nodes of the simplex and greatest near the middle.