

BEAM: Technology for Autonomous Vehicle Health Monitoring

Han Park, Ryan Mackey, Mark James, Michail Zak, Edmund Baroth
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 91109

ABSTRACT

BEAM (Beacon-based Exception Analysis for Multimissions) is an end-to-end method of data analysis intended for real-time fault detection and characterization. It provides a generic signal-level system analysis capability for potential application to deep space probes and other highly automated systems.

This paper describes in brief the architecture, applications, and operating theory of BEAM. BEAM provides a generalized formalism for diagnostics and prognostics in virtually any instrumented systems. Consideration is given to all standard forms of data, both time-varying (sensor or extracted feature) quantities and discrete measurements, embedded physical and symbolic models, and communication with other autonomy-enabling components such as planners and schedulers. This approach can be adapted to on-board or ground-based implementations with no change to the basic operating theory. The approach will be illustrated with an overview of application types, past validations, and ongoing efforts, including propulsion systems such as the Space Shuttle Main Engine.

INTRODUCTION

BEAM stands for Beacon-based Exception Analysis for Multimissions, which is a complete method of signal-level data analysis for real-time fault detection and characterization. The original intended application of this project was to provide a generic system analysis capability for deep space probes and other highly automated systems. Such systems are typified by complex and unique architectures, high volumes of data collection, limited bandwidth, and a critical need for flexible and timely decision abilities.

Beacon monitoring was the original impetus for this design. Beacon monitoring is a telemetry method wherein a subset of available engineering data is broadcast by the monitored system as follows: Rather than downlink the entire engineering dataset at all times, the approximate condition of the spacecraft or remote system is categorized into one of several "beacon tones." These tones correspond to (a) nominal system operation, (b) anomalous or interesting operation, (c) degradation or significant faults, or (d) total failure. The beacon tone is sent at all times using a much simpler telemetry system. Additional data is sent if and only if specific attention is requested or required of system operators. In other words, beacon monitoring represents a step towards machine self-awareness.

Beacon monitoring is only practicable if a number of hurdles can be overcome. Central to the method is the ability of a spacecraft to perform an accurate, timely, and verifiable self-diagnosis. Such a self-diagnosis must not only provide a safe operating envelope, but must perform, at worst, comparably to spacecraft experts in a control room. A system that is insensitive, generates false alarms, or requires oversight will not be effective, because such inefficiencies will be amplified in a "streamlined" process.

The basic premise of BEAM is the following: Construct a strategy to characterize a system from all available observations, and then train this characterization with respect to normal phases of operation. In this regard the BEAM engine functions much as a human operator does – through experience and other available resources (known architecture, models, simulation, etc.) an allowed set of behavior is "learned" and deviations from this are noted and examined. The approach naturally lends itself to integration with higher-level reasoners such as Model-based Reasoners (MBRs), an example of which is Livingstone. BEAM converts quantitative measurements into qualitative assessments that high-level reasoners can

then reason on. In Livingstone's terminology, BEAM can be thought of as a general purpose Monitor. Furthermore, BEAM can provide health-state information to decision makers, such as human operators or automated planning software. The flexibility of BEAM makes it not just solely suited to beacon monitoring, but more broadly applicable to monitored or wholly autonomous systems.

Two important features make BEAM a standout among the various fault-detection technologies that have been advanced. The first is its broad range of applicability. This approach has been used with sensor and computed data of radically different types, on numerous systems, without detailed system knowledge or a priori training. Separate components are included to treat time-varying signals and discrete data, and to interpret the combination of results.

The second is its ability to detect faults for which the detector has not been trained. This flexibility is of prime importance in systems with low temporal margins and those with complex environmental interaction. This ability also comes with few requirements in terms of detector training.

BEAM ARCHITECTURE

BEAM is a complete data analysis system for real-time or off-line fault detection and characterization at the signal-level. While the originally intended for generic system analysis on-board deep space probes and other highly automated systems, the compact and modular nature of its subroutines naturally lends itself to ground-based deployment.

At the simplest level of abstraction, BEAM is software, which takes data as input and reports fault status as output. Implementation of this software depends on the application, but a typical application would have a system with a number of individual components, each of which reports health or performance data to a local computer. To accommodate such a wide range of possibilities, the computational engine of BEAM itself is highly adaptable with respect to subsystem size and complexity.

For each single compartment or subsystem, we can expect to receive four types of data:

1. Discrete status variables changing in time – modes, switch positions, health bits, etc. – from sensors or software
2. Real-valued sensor data varying at fixed rates – performance sensors or dedicated diagnostic sensors
3. Command information – typically discrete as in 1.
4. Fixed parameters – varying only when commanded to change but containing important state information

These types of data are all valuable but used in different ways. Status variables and commands are useful to a symbolic model. Commands and fixed parameters are used in a physical system model while the time-varying sensor data are used in signal processing components. An optimal strategy must take each of these into account and produce a single, unified decision. In order to study each and combine results, we propose the following BEAM architecture. (Figure 1)

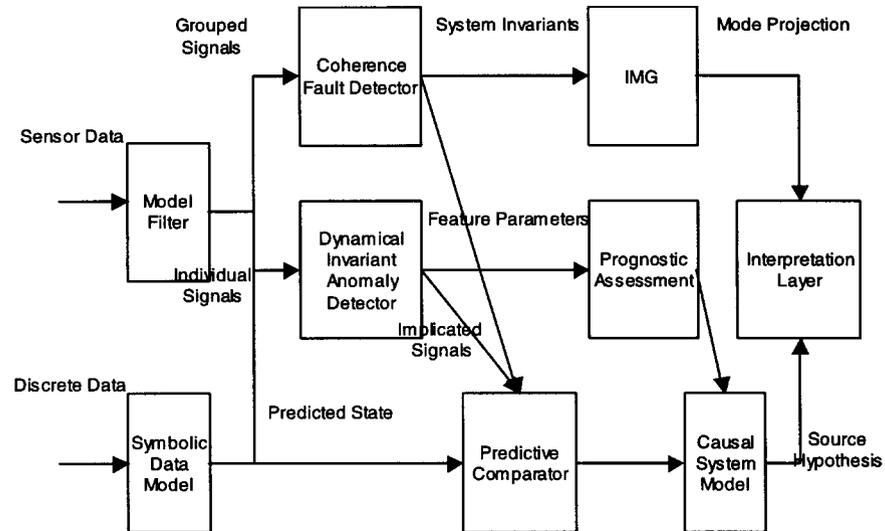


Figure 1. Top-level BEAM architecture.

A simple description of each module is given as follows. A more complete description can be found in [1].

1. **Model Filter (MF):** Receives sensor or other quantitative data, conditions it in terms of synchronicity or drop-outs, and combines results with physics model (simulation) predictions.
2. **Coherence Detector (CD):** Receives multiple conditioned quantitative signals and performs anomaly detection using cross-signal statistical features.
3. **Dynamical Invariant Anomaly Detector (DIAD):** Receives a single quantitative signal one at a time and performs anomaly detection using a parametric estimate of the residuals.
4. **Informed Maintenance Grid (IMG):** Combines outputs from the Coherence Detector over long operating periods to detect subthreshold degradation and predict functional failures.
5. **Prognostic Assessment (PA):** Uses the parametric signal estimates from DIAD to forecast future signal values and identify potential signal faults.
6. **Symbolic Data Model (SDM):** Receives discrete data and constructs an internal state estimate of the system. It detects discrete signal mismatches (explicit faults) and identifies system mode for use by other components.
7. **Predictive Comparator (PC):** Combines signal implications from the CD and DIAD as well as discrete reports from SDM in an attempt to unify results from the signal-based and symbolic reasoning components.
8. **Causal System Model (CSM):** Backtracks implications along the system structure to reduce the complexity of fault reports. This is a simplistic form of diagnosis.
9. **Interpretation Layer (IL):** Fuses results from different components and constructs a final report. It serves as an interface between BEAM and other components.

Specific instantiations of BEAM may omit some of these components depending on the particular characteristics of each system, such as sensor types, cost to maintain, characteristic time scales, and availability of model or training information. For example, only the Dynamical Invariant Anomaly Detector

was implemented for Space Shuttle Main Engine (SSME) analysis due to the transient nature of SSME faults.

DATA PROCESSING AND DATA FLOW

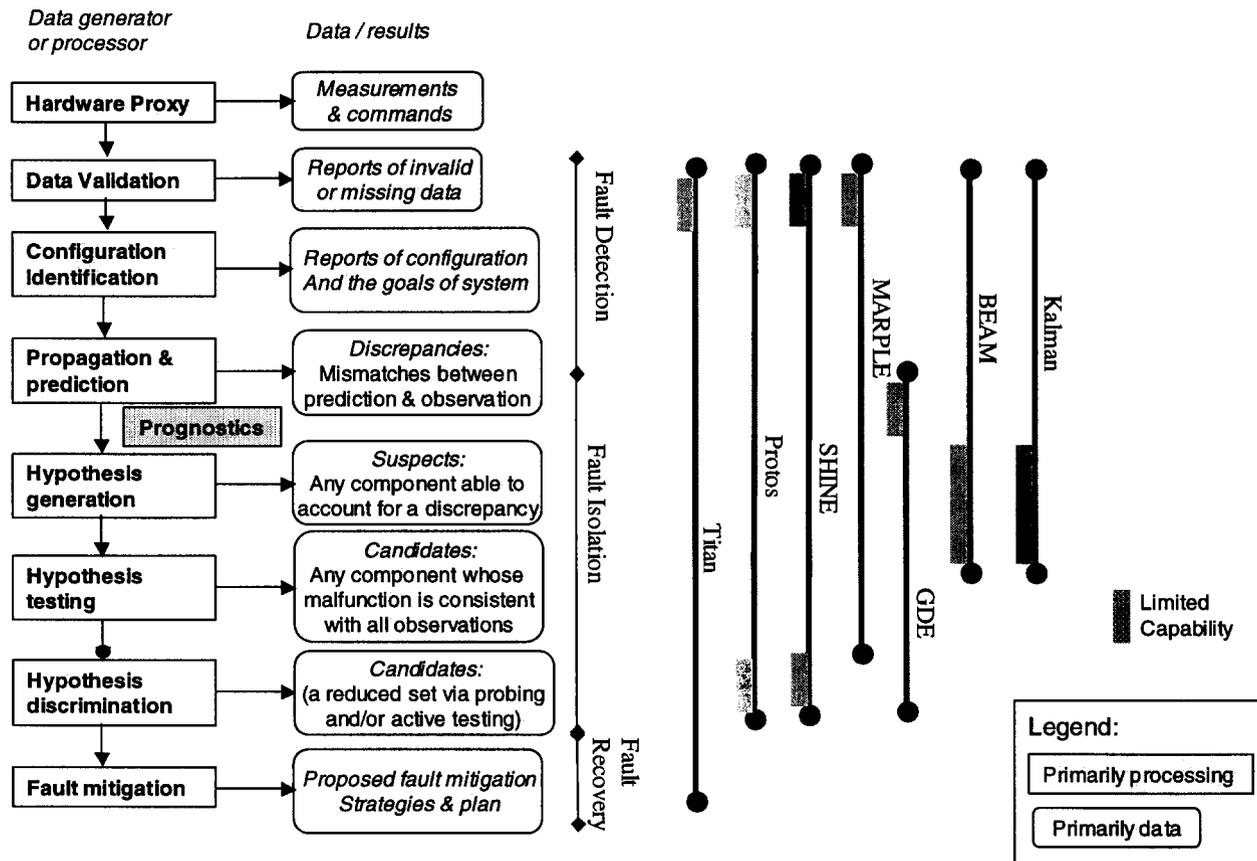


Figure 2. Processing Steps and Data Flow.

APPLICATIONS

Since its original inception, BEAM has been matured and proven on many separate applications, both on-board and off-board. BEAM has been studied on numerous applications and is currently being formulated for inclusion into a number of systems, some of which are described in related papers. In this section we will consider the basic classes of application and the value of such a sensing strategy. The shape and scope of BEAM is highly dependent on the application, according not only on the specific system but on its mission plan as a whole. Refer to the related references for more in-depth analysis of particular cases, which highlight certain qualities of BEAM.

Telemetry Limited Remote Systems

The first and most obvious customer for BEAM is typical of systems developed at JPL. The core mission of JPL is robotic space exploration. Such systems are usually managed by a team of experts who review the engineering data on a regular basis. These missions are also very expensive, and represent a considerable investment in manpower and time.

Cases like these, where we can count on human experts to oversee the analysis process, are well suited for BEAM. In these cases BEAM may serve to enhance operator effectiveness by providing a first look at the data, effectively compressing it to contain only the relevant, "most interesting" parts. Benefits as a result can be expressed in terms of workload, safety, turnaround time, reliability, or cost in work-hours.

Complex spacecraft are almost always accompanied by accurate models and simulation left over from the design process. They undergo a rigorous qualification and are tested through a variety of fault cases. However, they are impossible to repair or upgrade (except through software), they are typically unique, and because of this and the unique environments they encounter, they face a large number of "novel" situations.

There are two basic philosophies to BEAM for spacecraft applications: on-board and off-board. Off-board implementation provides operators with a simple tool to take full advantage of design models and to study systemwide interactions. It isolates all incidences of "failed" and "interesting" data, and for these cases produces a best-guess of the source. These results are logged for future comparison and long-term degradation analysis.

On-board application is more difficult and less flexible, owing to more rigorous software qualification and more stringent processing requirements. However, there are specific benefits that can only be realized in an on-board implementation. Most important is the time-criticality of detection. Faults can interrupt critical phases of deep space missions, such as spacecraft maneuvering or encounter. Because of the large delays in communication, it is highly desirable to provide detection and recovery capabilities to handle a wide variety of unforeseen occurrences.

In both cases, BEAM functions to reduce operator workload and mission cost while providing quicker spacecraft understanding and broader safety margins. On-board implementation offers a further increase in system safety, but is much less tolerant of false alarms. When operated strictly off-board, much higher sensitivity may be requested by the operators to aid in their analysis of special cases.

Experiments using BEAM in both configurations were conducted using the Cassini spacecraft Attitude and Articulation Control Subsystem (AACS) simulation (on-board analysis) and actual flight data (off-board analysis). The AACS is an extremely complex subsystem, incorporating approximately 1,600 possible telemetry signals at rates of up to 8 Hz.

Results of these analyses were presented at the AIAA '98 conference in Huntsville, Alabama [5]. For the on-board simulation case, BEAM was able to match the performance of existing fault detection software -- and in some cases exceed it, particularly with respect to time of detection -- over a portion of the fault protection test grid. The off-board analysis experiment was a blind study applied to the first six months of flight data, including pre-flight, launch, and systems checkout.

Despite lacking the commands sent to the spacecraft and knowledge of the system design, BEAM was able to automatically detect and isolate all unusual events. This was compared against the Incident / Surprise / Anomaly (ISA) reports, produced by the operations team, at the end of the experiment. Such a result means "no false negatives," i.e., no missed detections, were produced by the detector.

With regard to false positives, the question of performance is more complicated. In addition to replicating the ISA reports both in terms of (a) detection, (b) timing, and (c) affected signals, BEAM also detected three events not reported in the ISA logs. These three events were later revealed to be (1) launch of the spacecraft, (2) first trajectory correction maneuver (TCM), and (3) second TCM. However, given that the BEAM system was unaware of the commands sent to the spacecraft, this result is neither surprising nor troublesome. Cassini is an example of a unique system with a large amount of ground support, and is therefore relatively tolerant of false positives. Such is not the case for many of the examples described below.

Deep Space Station Controller

A very different class of problem is faced by many earth-bound systems that nonetheless can benefit from this approach. There are numerous examples of complex machinery that rely upon internal sensing and expert operators to provide operating margins for reasons of safety or efficiency. In such an application, like the spacecraft example above, pure autonomy is not necessary. Instead it is desirable to empower the system operators by making system health management an easily accessible and controllable function.

This problem is typified by the Deep Space Network (DSN), which is managed by JPL. Downlink is a crucial element in space exploration, as spacecraft observations are useless if their results cannot be retrieved. The communications antennae are highly subscribed, making reliable operation of great importance. DSN antennae are complex electromechanical structures, in many ways no different from spacecraft or aircraft systems. They contain power supplies, hydraulics, signal generators, radiating elements, and so on. Many of these systems are constructed with health and performance monitors.

BEAM integration with DSN console tools is an ongoing task at JPL. In particular, we seek to allow control of an entire array of communications antennae, including the ability to "hot swap" other antennae if constraints or faults affect the system, from a single console.

Preliminary results of this effort were published in [6]. Work is ongoing to mature this system, and to provide direct communication with automated planners and other components [7].

Maintenance Cost Reduction

A third class of problem applies to domains such as piloted aircraft. In such an application, the focus is not on producing new tools for the pilot, but instead upon simplifying the job of maintaining the system.

Aircraft maintenance is a costly endeavor, often involving scheduled maintenance and inspections in addition to replacement of faulty components, which is made more costly by high false alarm rates and poor isolation. Additionally, large fleets of aircraft require relatively simple repair procedures, as system experts are usually unavailable in quantity. This differs greatly from the previous cases. We wish to take advantage of the large body of work and data existing for such aircraft, and their greater numbers, to drive a system that makes analysis and repair a routine task. Furthermore, since the fleet of aircraft is expected to age and degrade with time, we require tools to detect and characterize these new phenomena to allow meaningful prognostics.

A good example of this technology applied to a current aircraft system is given in [2]. This specific example shows how an aircraft hydraulic system can be treated using this method. The result is a diagnostic tool that is very quickly trained, enjoys excellent false-alarm rates, and exceeds the performance of current-day fault detection techniques.

Space Shuttle Main Engine

The BEAM anomaly detection system has been applied to SSME in a joint effort between JPL and the Marshall Space Flight Center (MSFC). MSFC is evaluating BEAM as an automated tool for rapid analysis of SSME ground-test data. BEAM offers a way to dramatically reduce SSME analysis time and cost while providing enhanced safety. The BEAM approach is at the cutting edge of fault detection technologies and represents a significant technological improvement over existing SSME analysis tools. Those tools include simulation and detailed fault-trees, and are themselves reasonably sophisticated, but remain labor-intensive and incomplete. BEAM automatically indicates specific time periods, signals, and features contributing to each anomaly. In the case of the SSME experiment, this represents a dataset reduction of roughly 100:1, with attendant impact on analyst effort and effectiveness. The software described here executes on a standard workstation and delivers analyses in seconds, a computing time comparable to or faster than the test duration itself. While there is an inevitable learning curve associated with proper training and interpretation of the algorithm, training algorithms themselves are also automated, and a future improvement to the software could include any standard output interface.

For the SSME application, a custom version of BEAM was built to analyze data gathered during ground tests. Since BEAM consists of modular components, a custom version can be tailored to address specific

applications and needs by mixing-and-matching components. The initial build of BEAM for the SSME focuses on signal processing and contains three components: Coherence-based Fault Detector (CFD), Dynamical Invariant Anomaly Detector (DIAD), and Symbolic Data Model (SDM).

The DIAD module was the first applied due to its particular suitability to SSME data features and the expected transient nature of SSME faults. It detects anomalies by computing coefficients of an auto-regressive model (i.e. dynamical invariants) and comparing them to expected values as extracted from previous data. DIAD is particularly sensitive to anomalies caused by faulty sensors, subtle and sudden performance shifts, and unexpected transients. DIAD was trained using sixteen nominal test firing data from Block II engine series. Each test data contained 100 channels of sensor data covering the major subsystems on the SSME. DIAD monitored 86 critical engine sensors out of the 100 sensor data. Fourteen channels were control parameters such as engine commands to identify the mode of operation. The two critical control parameters were the Engine Status Word and Commanded Main Combustion Chamber Pressure. They were used to deduce and identify the phase, mode, and power level of the engine.

DIAD was used to detect anomalies in seven different test data that contained some of the most commonly encountered anomalies in SSME testing. These include High Pressure Fuel Turbo Pump (HPFTP) blade failure, Low Pressure Fuel Turbo Pump (LPFTP) and HPFTP cavitation, High Pressure Oxidizer Turbine (HPOT) performance shift, fuel flowmeter shift, frozen sense lines, and deactivated sensors. In the seven anomalous data files, DIAD was able to detect anomalies in each file. DIAD detected all the major anomalies including HPFTP blade failure, LPFTP and HPFTP cavitation, HPOT performance shift, fuel flowmeter shift, frozen sense lines, and deactivated sensors. The frozen sense line and fuel flowmeter shift results are shown in Figures 10 and 11. There were a couple of exceptions. DIAD was not able to identify the anomaly in the HPFTP during fuel turbine pump cavitation event. It was, however, able to detect the cavitation in the LPFTP. DIAD was also able to detect some of the minor efficiency shifts in the HPOT. Overall, the DIAD was sensitive to all of the major anomalies in the seven anomalous data and detected the shift in the data characteristics.

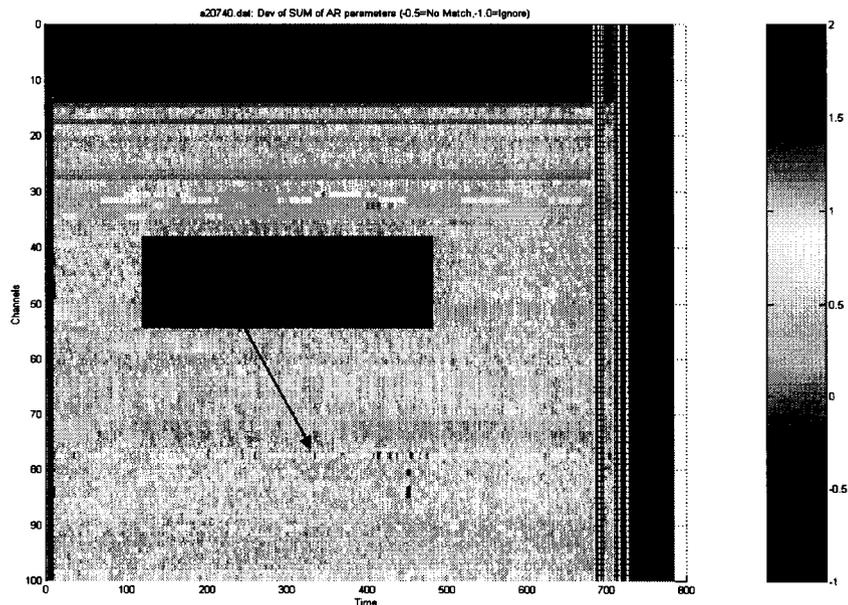


Figure 10. Overall system view plot. The x-axis is time, and the y-axis is the channel. The color indicates deviation from nominal conditions. The frozen sense line anomaly is at channel 78 as indicated by the arrow.

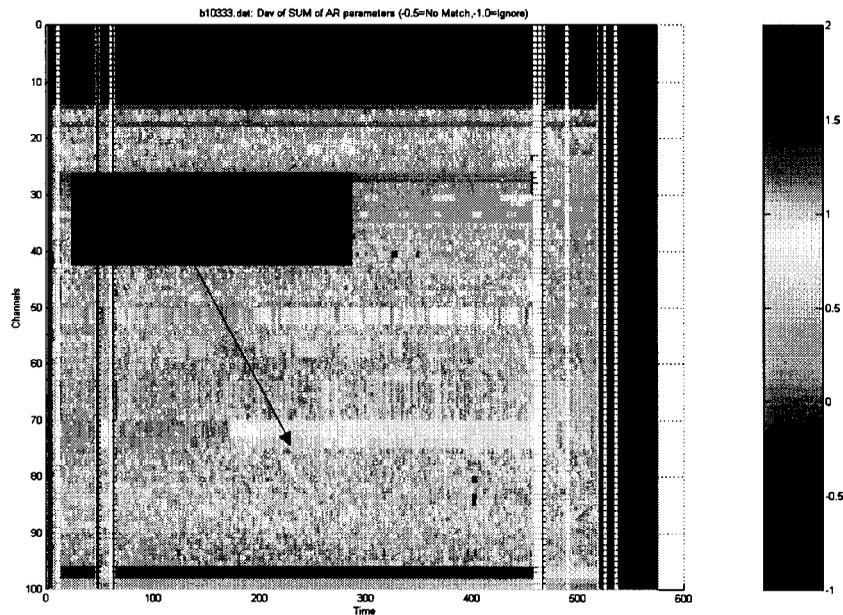


Figure 11. Overall system view plot. The x-axis is time, and the y-axis is the channel. The color indicates deviation from nominal conditions. The fuel flow meter shift in the LPFTP is indicated by the arrow.

SUMMARY AND CONCLUSIONS

Presented here is the framework for BEAM, a comprehensive self-analysis tool suitable for inclusion for virtually any monitored system. As the aerospace industry faces greater challenges in mission complexity, advanced design, aggressive cost targets, and true autonomy, so must we evolve the support structure that goes with them. In most cases, system modeling and available sensors are more than adequate for the task. Yet it remains to take full advantage of this information. Extracting knowledge from the data and domain information is an essential step in the process of self-analysis and control.

It is the opinion of the authors that a strategy for autonomous control must include all sources of information, and that this information must be fused methodically and deep within the process in order to be of greatest use. The method outlined in this paper seeks to engage all such data at the subsystem level, and from it construct simple but profound conclusions as to the operating health and capability of the system, as well as projected viability and corrective actions.

No such strategy can ignore the fundamental question of feasibility, no matter how attractive. For this reason our system seeks to mimic the logic of a human operator, and draws its training from many of the same sources. The fault detection, isolation, and prognostic conclusions are based upon physical models of arbitrary fidelity, symbolic models, example nominal data, real-time data, and architectural information such as connectivity and causal diagrams. The algorithms have also been scaled to operate comfortably on current-generation flight processors.

Such an architecture makes adaptability to numerous systems possible, but also increases the usefulness of the system in concert with more standard methods of control and analysis. Because the data products themselves are accessible at every stage of analysis, BEAM can benefit systems run at any stage of autonomy -- from the extreme of complete self-determination as part of the core flight software, to the opposite of complete human control as a highly advanced analysis tool.

This last point is more valuable than it may appear. A simple reality of advanced systems is that no testing process can ever be complete, and any new system faces considerable uncertainty in the field. It is a given that novelty will be encountered, and in order to improve upon the system, its safeguards must allow it to survive, to characterize the novelty, and to permit improvements to its software control. This process, in nearly all cases, requires human intervention. BEAM approaches this problem by incorporating physics models and studying embedded features of the operating physics of the system in order to isolate a broad class of anomalies, and applies all available rules and limits of allowed system performance. Following isolation, the results of this analysis are communicated to system experts, who may then revise the software through modifications to those same models, inclusion of new training data, or addition of new symbolic rules.

This process, begun at JPL as a method of improving spacecraft safety and operating costs, is presently under study or being applied to numerous aerospace applications, across the entire spectrum of autonomy. It is hoped that this work will contribute to the greater problem of flight software in total, and thereby simplify some of the challenges facing the next steps in vehicle autonomy.

ACKNOWLEDGEMENTS

The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

REFERENCES

Mackey, R., James, M., Park, H., Zak, M., "BEAM: Technology for Autonomous Self-Analysis," IEEE Aerospace Conference, Big Sky, Montana, March 2001.

[1] Zak, M., Park, M., "Gray Box Approach to Fault Diagnosis of Dynamical Systems," IEEE Aerospace Conference, Big Sky, Montana, March 2001.

Park, H. G., Mackey, R., James, M., Zak, M., Kynard, M., Greene, W., Sebghati, J., "Analysis of Space Shuttle Main Engine Data Using Beacon-based Exception Analysis for Multi-Missions," IEEE Aerospace Conference, Big Sky, Montana, March 2002.

[2] Mackey, R., "Generalized Cross-Signal Anomaly Detection on Aircraft Hydraulic System," IEEE Aerospace Conference, Big Sky, Montana, March 2001.

[3] James, M., Atkinson, D., "Software for Development of Expert Systems," NASA Tech Briefs, Vol. 14, No. 6, June 1990.

[4] Box, G. E. P., Jenkins, G. M., Reinsel, G. C., Time Series Analysis, New Jersey, Prentice Hall, 1994.

[5] Gulati, S., Mackey, R. "BEAM: Autonomous Diagnostics and Prognostics for Complex Spaceborne Systems," (poster session) AIAA 1998, Huntsville, Alabama, November 1998.

[6] James, M., Dubon, L., "An Autonomous Diagnostic and Prognostic Monitoring System for NASA's Deep Space Network", IEEE Aerospace Conference, Big Sky, Montana, March 2001.

[7] Fukunaga, A., Rabideau, G., Chien, S., Yan, D., "Toward an Application Framework for Automated Planning and Scheduling," Proceedings of the International Symposium on Artificial Intelligence, Robotics, and Automation for Space, Tokyo, Japan, July 1997.

[8] Colgren, R., Gulati, S., Koneck, R., "Technologies for Reliable Autonomous Control (TRAC)," IASTED '99 Control and Applications Conference, Banff, Canada, July 1999.

[9] Schaefer P., et. al., "Technologies for Reliable Autonomous Control (TRAC) of UAVs," IEEE Aerospace Conference, Big Sky, Montana, March 2001.

DISTRIBUTION STATEMENT

Approved for public release, distribution is unlimited.