UAH RESEARCH REPORT NO. 810

Software Development for Avionics
SSME Control and Diagnostics
Final Report

Prepared by

A. Choudry
Electrical & Computer Engineering Dept.
University of Alabama in Huntsville
Huntsville AL 35899

Prepared for:
NASA
Marshall Space Flight Center
Huntsville AL 35899

August 1989

Introduction

This task addresses the general problem of software development for real-time control in distributed systems. The space station and other future NASA platforms will have a large number of sensors distributed throughout the system. A comprehensive software system is needed to integrate the sensor data and make decisions for real-time control. There are several important problems that have to be solved in developing software for such a complex task. This involves;

-- Real-time multi-channel data acquisition
-- Multi-stream asynchronous data analysis
-- Data flow architecture emulation
-- Task synchronization through message passing/memory sharing
-- AND/OR parallel inferencing (mostly forward chaining)
-- Presentation of results for ready assimilation and distribution.

The scope of this project is generic enough to be applicable to various NASA systems. Instead of doing an abstract general software development we have chosen the SSME (Space Shuttle Main Engine) as a concrete example of distributed system and attempted to develop software for its real time control. A part of this problem was studied earlier under a NASA Summer Fellowship and most of that work serves as a foundation for this further development

The Space Shuttle Main Engine (SSME) based on Hydrogen-Oxygen combustion is a very complex power plant employing numerous pumps, valves and ducts. During a ground test about 500 sensors are used to monitor the state of SSME. Some of these sensors are used for the close loop control of SSME and are connected to a Computer System 'Engine Controller' To evaluate SSME performanace 1200 hot-fire ground tests have been conducted, varying in duration from 0 to 500 secs. During the test about 500 sensors are sampled every 20ms to measure the various parameters. The sensors are generally bounded by 'red-lines' so that an excursion beyond the red-line could lead to premature shutdown by the operator. In 27 tests, guided by the red-lines, it was not possible to effect an orderly premature shutdown. These tests became major incidents where serious damage to the SSME and the test stand resulted. In this study we have investigated the application of pattern recognition and allied techniques in a distributed real time system to detect trends that lead to major incidents. Based on the sensor data a set of (n) features is defined. At any time, during the test, the state of the SSME is given by a point in

the n-dimensional feature-space. The entire history of a given test can now be represented as a trajectory in the n-dimensional feature space. Portions of the 'normal' trajectories and the failed test trajectories would lie in different regions of the n-dimensional feature space. The feature space can now be partitioned into regions of normal-tests and failed tests. In this manner it is possible to examine the trajectory of a test in progress and predict if it is heading into the 'normal-region' or the 'failure-region' of the n-dimensional feature space. In this study we have developed techniques to extract features from ground test data, as supplied by Rocketdyne, and develop feature space trajectories for the tests. The initial results though looked very promising, their real time interpretation in an n-dimensional space became too cumbersome. We have developed the analysis further and reduced the n-dimensional problem to a composite 3-dimensional solid. The failure modes can, in principle, be recognized as distortions in the solid.

Data Structure

There are 3 different data acquisition systems used to collect the sensor data (1,2), namely,

-- Command and Data Simulator (CADS)

-- Facility Recording (FR), and

-- Analog High Frequency Recording (AHFR)

In Fig. 1, the salient points of these systems is shown. The engine controller uses 16 bit computations on 12-bit data words to perform close loop operation of the SSME. For the SSME Anomaly and Failure Detection (SAFD) analysis, as reported in (2), the CADS and FR data provide the bulk of the input.

In all about 1200 hot fire tests have been conducted on the SSME. In 27 tests the SSME went out of control and serious damage to the engine and the teststand resulted. A summary of some of the salient points of the ground tests is given in Table 1.

Considering that the replacement cost of an engine is ~$50M, it is highly desirable to develop some technique for detecting failure trends which would allow an orderly shutdown of the SSME and thereby preventing a major incident (3). In (2) and (3) various techniques for failure detection have been suggested including the following,

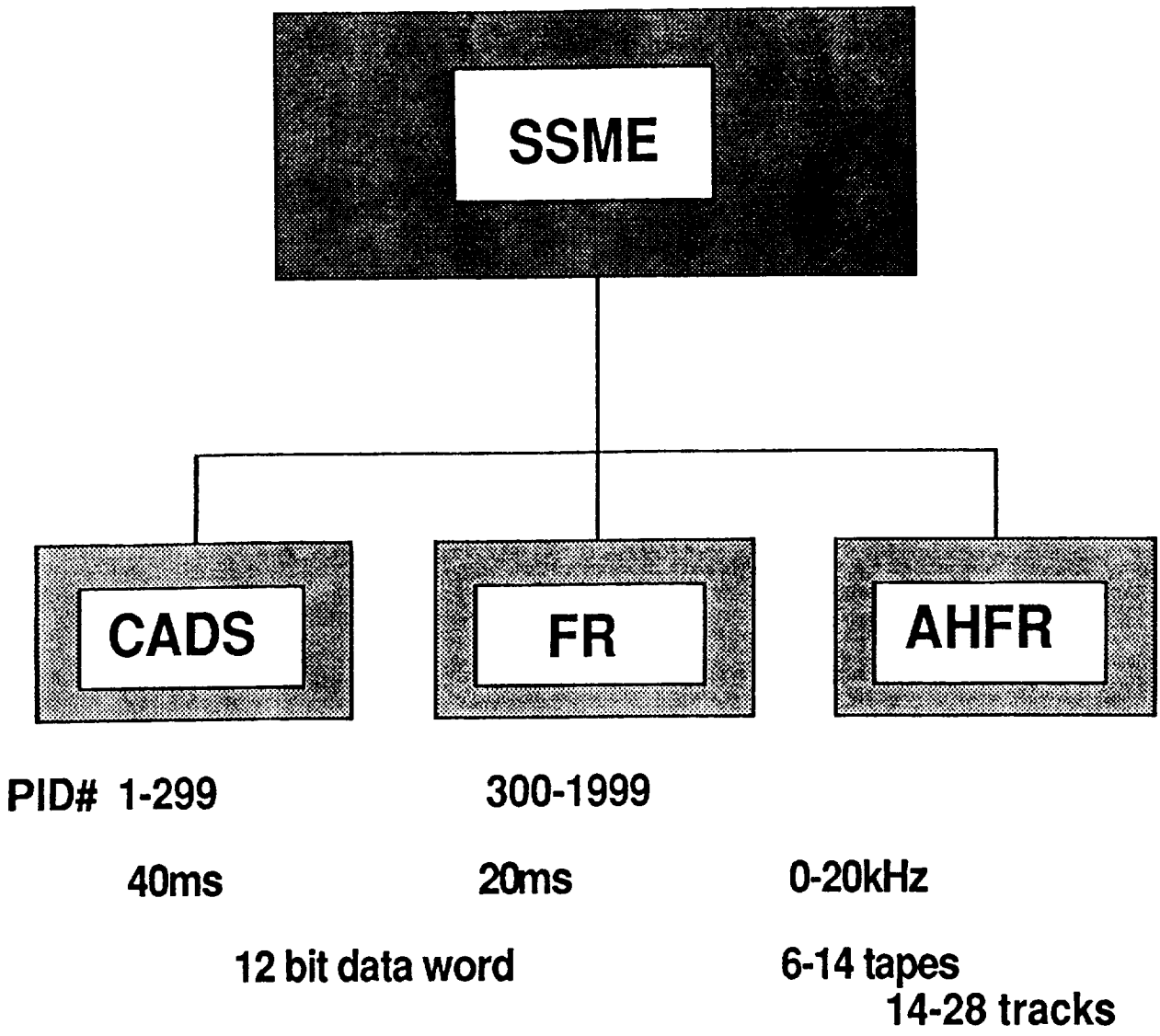


Fig. 1 SSME Data acquisition system

-- Generalized Likelihood Ratio (GLR)
-- Generalized Likelihood Test (GLT)
-- Voting
-- Confidence Region Tests
-- Kalman Filters
-- Parameter Estimation
-- Jump Processes
-- Pattern Recognition.

The success of a technique will be determined by;

-- detecting the fault fast enough to allow an orderly shutdown
-- identifying the technical nature of the fault.
-- feasibility of its implementation in software for real time execution

The last point is particulary important for this project and will determine the choice of the ultimate software.

TABLE 1. GROUND TEST SUMMARY

- -1200 HOT- FIRES
- -27 MAJOR INCIDENTS
- -TEST DURATION 0-500 SEC.
- -300-500 SENSORS MONITORED
- -SAMPLING RATE 50 Hz.
- -DATA WORD 12 bits
- -DATA TRANSFER RATE 0.5-1Mhz
- -DATA VOLUME 0.1 - 1Gbits

the basic decision process involved in SSME fround tests can best be described by the 'weighted truth-table' in Fig. 2 which shows the probability W for various actions.

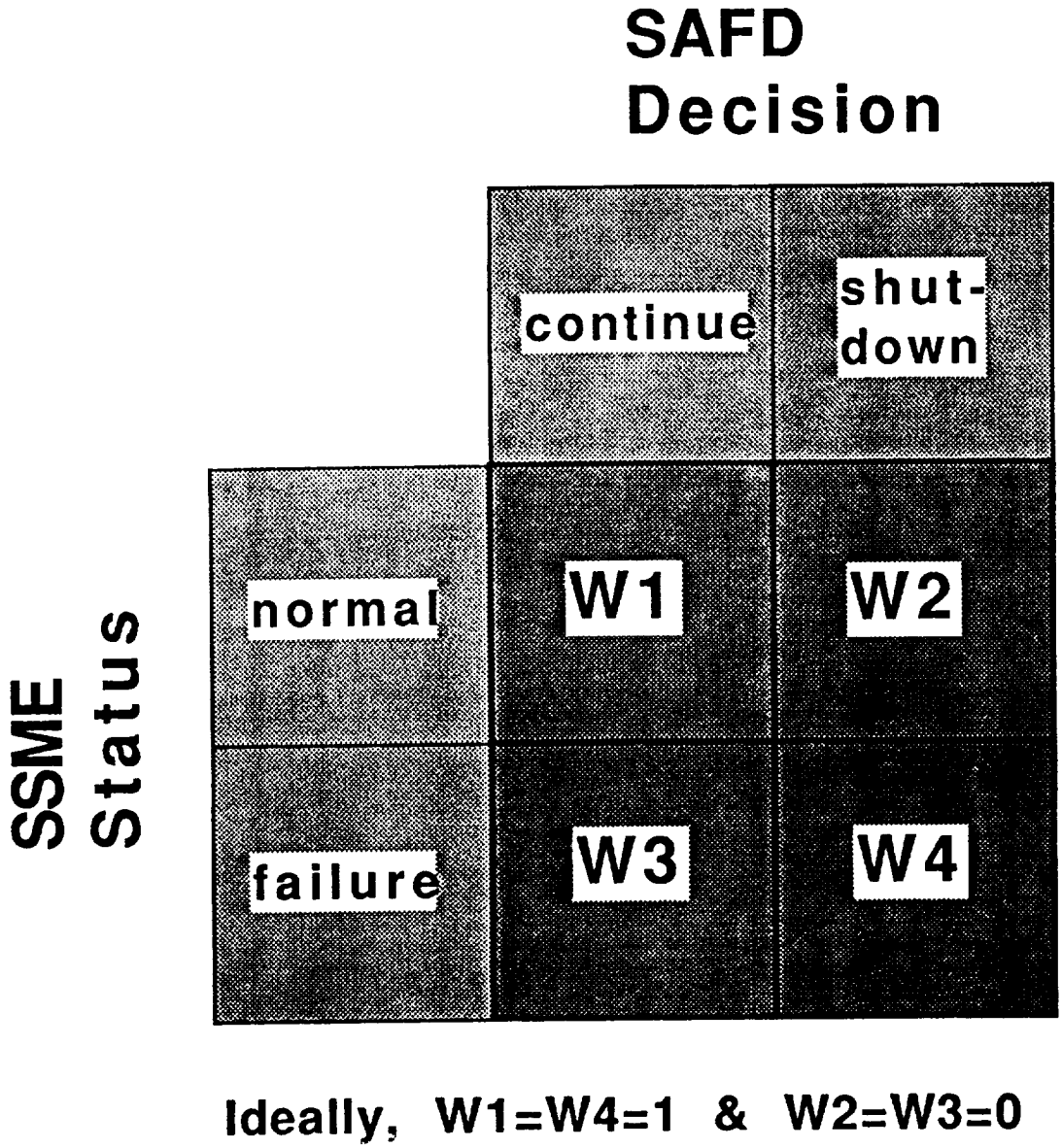


Fig. 2 SAFD performance matrix

Note that W2 being the probability of a false alarm should be zero, however, a small value ,say 1%, may be acceptable. On the other hand W3 being the probability of a miss should indeed be zero, just as W4 = 1 i. e. shutdown in failure mode.

Various alternatives have been considered for implementing such a SAFD. We shall consider the use of Pattern Recognition (PR) techniques for SAFD. It should, however, be realised that PR in general is very time consuming and real time applications of PR require special attention. We would address this point with some detail.

It should also be pointed out that much of the data processing in PR, as described below, can also be used for the other vital activities envisaged for the future systems, namely, real-time control, health assessment and condition monitoring (4,5).

Pattern Recognition (PR)

The fundamental premise for applying PR techniques is the observation that when systems fail due to internal causes there are always some warning signs that preceed the event. Furthermore, the progression of a system from normal operating mode to anomalous (failure) mode does not happen at random but follows a pattern which can be analysed and explained. The object of PR technique, described here, is to identify the patterns that have led to failures and use this knowledge to look for warning signs in future tests and predict failures well in advance of their occurence.

The current practice is based on red-lining the sensor outputs. The red-lining of n-sensors can be easily explained in terms of a polyhedron in n-dimensions as shown in Figs. 3(a,b,c). Each sensor is assigned a lower- and an upper-bound value for 'normal' operation and these define the two 'red-lines' for that sensor. For a 3-sensor case the state of the system, at a given time, can uniquely be defined by a point in the rectangular prismatic region of the S1-S2-S3 Space (S-Space), Fig. 3c. The collection of these state-points at successive times would define a trajectory in the S-Space. All the possible normal runs of the system would then be given by trajectories that lie entirely within the 'red-lined' rectangular prism as shown in Fig. 4. In principle, any trajectory that tends to approach a boundary and exit to the outside region is an indication of an imminent failure.

One can learn to detect the failure trends by examining the data of the 27 tests that resulted in failure and compare it with the normal test data. It is quite possible that the failure trajectories will reveal their different character (as compared to normal trajectories) even before coming close to the red-line polyhedron boundary as shown in Fig. 5.

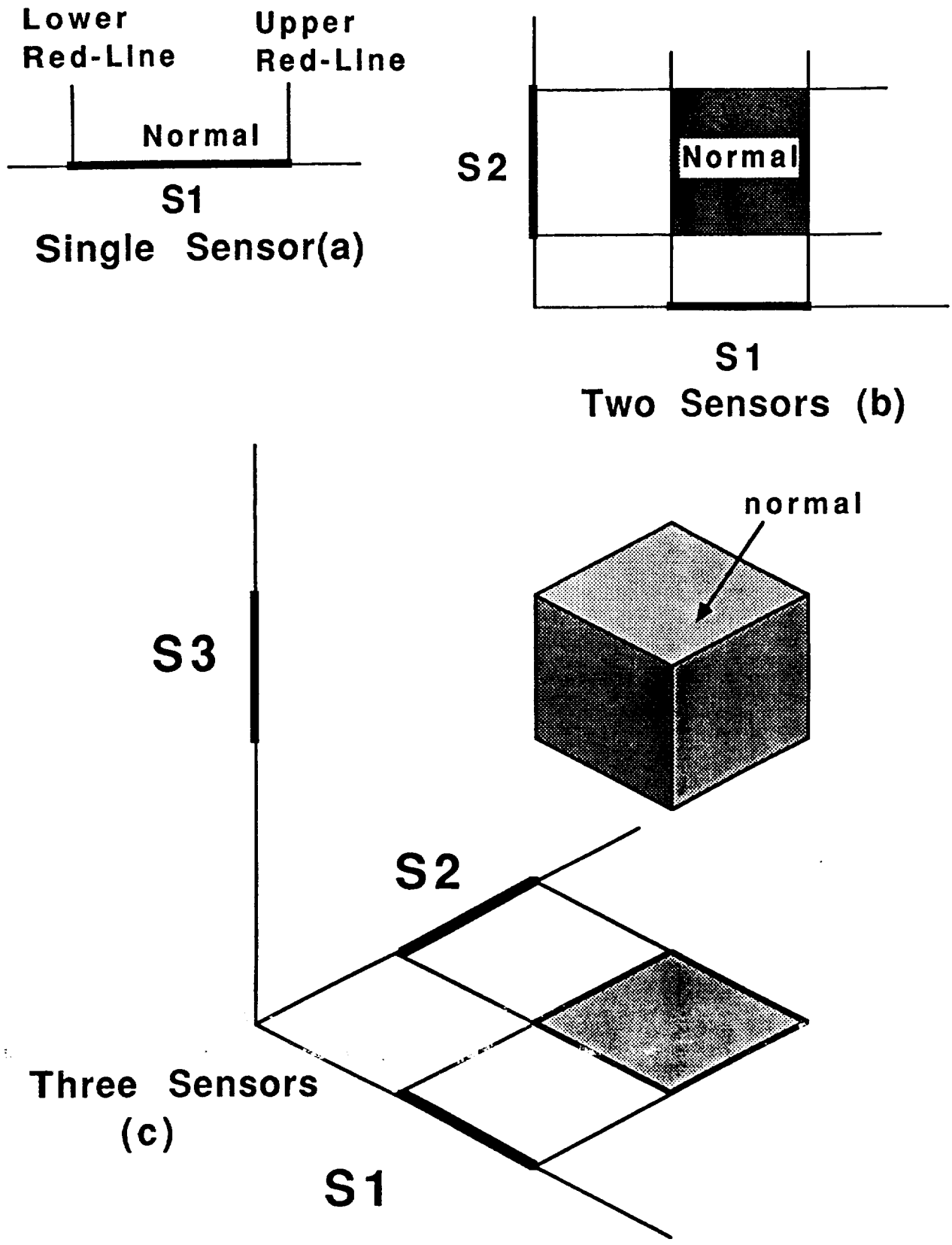


Fig. 3. Multi sensor 'red-lining'

ORIGINAL PAGE IS
OF POOR QUALITY

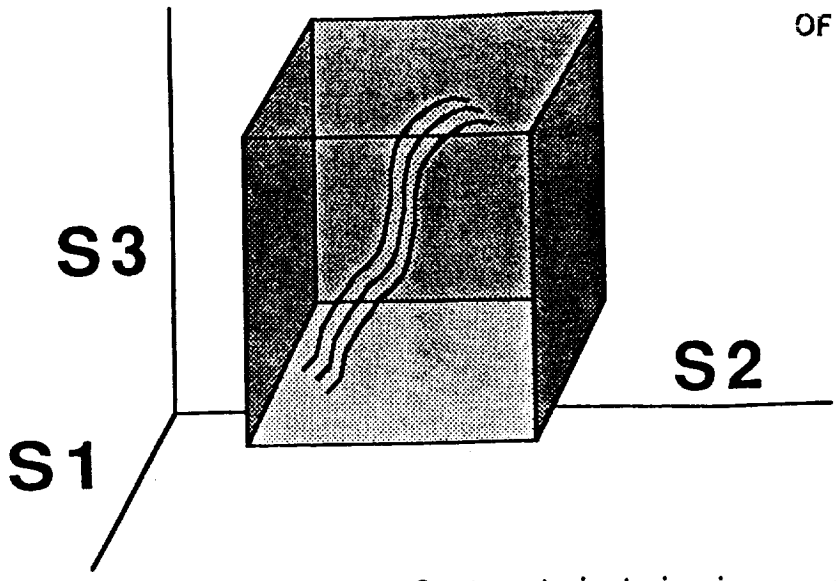


Fig. 4. System trajectories in sensor space

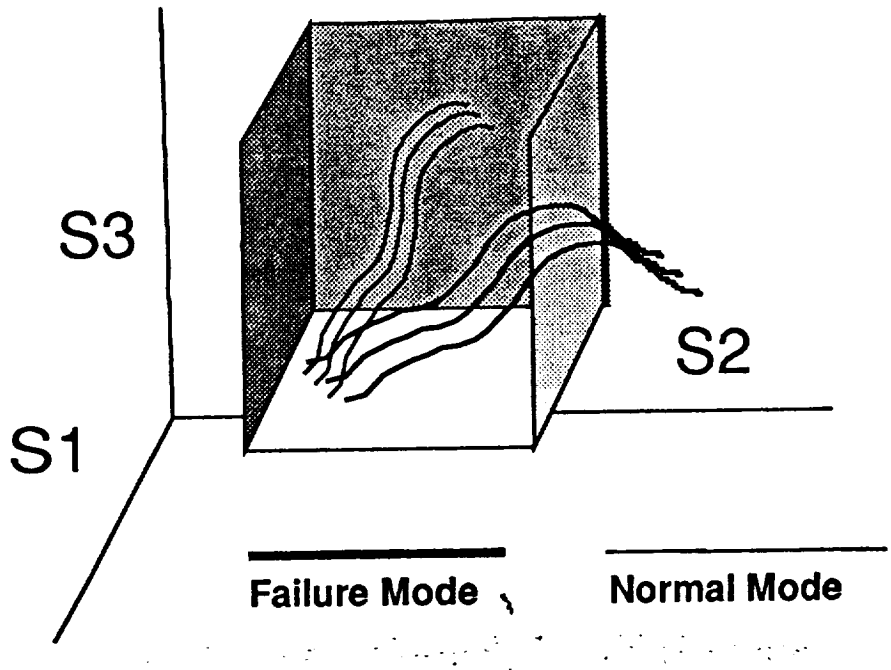


Fig. 5. Failure and Normal Mode Trajectories

There are two points that have to be considered in this context, namely,

1. The straight forward fixed red-lines for a sensor are adequate only for very special cases where no coupling among the sensors exists, i. e. the red-line for a given sensor is independent of the values of all the other parameters as measured by the rest of the sensors. Let r_k be the red-line for the kth sensor, then

$$r_k = c_k, \quad \text{where c's are constants}$$

The software red-lines can be defined by replaing c's by functions f_k so that,

$$r_k = f_k(S_1, S_2, \ldots S_n), \quad \text{where } S_k \text{ is the kth sensor reading}$$

In real-time this implies that as the test is progressing the readings S_k are used to calculate the various r_k's through f_k's. This can become not only computationally quite cumbersome but the explicit form of f_k itself has to be known perhaps from a simulation model of the system. In principle, it is simple to build the simulation model in a modular manner (6), however, the ad hoc nature of such models leads to different control and real-time simulation models. By such models it is quite possible to determine most of the f_k's, however, some crucial gaps may exist in this knowledge since not all the failure mechanisms are well understood.

2. Even if the f_k's are known and the soft red-lines can be determined, there is yet another serious problem. In principle all red-lines, soft or otherwise, are based on a single time frame of the system without considering how the system got to the state represented by the time frame. Questions of the type; has the system reached its present state through a transient, slow drift, excessive noise or under a close-loop command etc., are not considered by red-line methods. The method proposed here considers the entire system trajectory and compares it with other trajectories to detect failure prone trajectories.

The PR technique we propose to employ here has two important steps,

-- extension of the sensor-space into **Feature Space**
-- **Segmentation** of the feature-space into normal- and failure-regions

ORIGINAL PAGE IS
OF POOR QUALITY

Feature Space

The sensor space discussed above has two major drawbacks, namely,

-- For a truly multi-sensor system such as SSME the total amount of data is too large (about 100 Mbits) and can become too unwieldy for real-time processing.

On the other hand most of the data is of routine nature and a tremendous amount of data compression can be achieved by isolating and analysing only the deviations from the norm or the steady state. The norms can be defined as those values which can be calculated or predicted (assuming normal SSME operation) from a few key parameters e. g. power level, MCC pressure, throttle position etc. In the simplest case, only the deviations in sensor values, as compared to a moving average defined over a certain interval, are to be used for further analysis. This may even include deviations caused by closed- or open-loop control commands as may happen during throttling.

-- The sensor space, as based only on the sensor values, may not highlight the features important for SAFD.

This is based on the fact that the raw sensor readings, along with their red-lines, may themselves be not good indicators of impending failure. Further processing is often required to calculate features which are directly related to the failure modes. In Fig. 6 we show some of the features that can be defined for a given sensor. Starting with the raw value one can calculate first an average over a certain interval and then the deviation from it. From these one can also calculate the signal to noise ratio S/N which could be another feature. To detect drifts one can also calculate the local gradients as another feature. Similarly Fourier Transform of the signal (or the deviation), over a given time window, can be another feature, as shown in Fig. 6. One can also define 'compound' features involving data from more than one sensor. Thus, if needed, the net thermal flux, which may not be measured by a single sensor, can be calculated from the presure, flow velocity and temperature as measured by sensors in the MCC and it can be used as a feature for failure detection.

Based on the above discussion, the sensor space is replaced by a feature-time space, where a feature is defined as the deviation from the norm or steady-state as calculated from some key parameters or by averaging over a specified interval. The state of SSME, at any given time will thus be represented by a state point in the feature space. In Fig. 7 a normal SSME run is shown in a two-feature space. In a normal run, all the state points cluster around the time axis as shown, since no large deviations are encountered.

F / S	Value v	Devatn v'=v-<v>	S/N	Grad dv'/dt	FT(v')
S1					
S2					
S3					
S..					
Sn					

Fig. 6 Sensor Values & Features

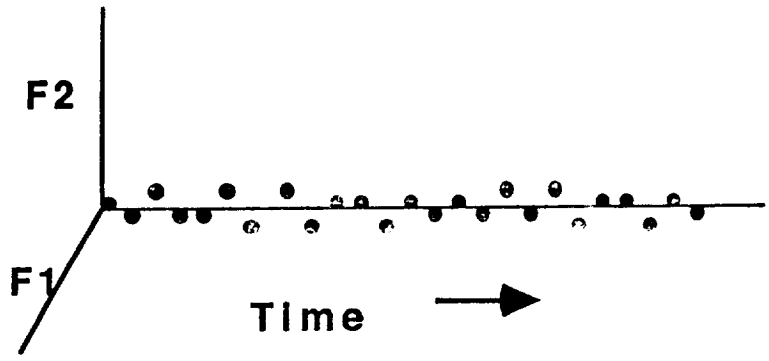


Fig. 7 Feature Space Representation of Normal SSME Test

Segmentation

Segmentation is the process of partitioning the feature space into clusters that can be identified with definite states of the system e. g. pre-failure or normal. An idealised illustration of this is shown in Fig.10 where the entire feature space has been projected along the time axis. All the points representing the normal runs should lie in a small region, cluster 1, around the origin. During normal runs there are large deviations caused by genuine excursions such as throttling etc. Such states of the system might show up as another region, cluster 2. It is anticipated that the deviations due to the failure modes will be of different nature and hopefully form another distinct region, cluster 3. An another form of the same situation is depicted in Fig.11 where an entire run is represented by a trajectory. A steady run trajectory would then lie entirely within cluster 1 whereas a controlled excursion in a run might cause the trajectory to migrate to cluste 2, but eventually return to cluster 1 after the steady state has been reached.

In practice, the situation may not be quite so clean cut, the clusters may not have so well defined boundaries and they may overlap. A number of powerful statistical techniques is available to locate cluster boundaries in such cases . It is also possible to assign to each state point, the probability of membership to a given cluster. One can also define a distance metric in the feature space to group points in clusters. In Table 2. (7) some of the commonly employed distance measures and the associated error bounds are shown. This description and the metrics have been used in an earlier study (8). It was, however, found that these techniques are still too complex for real time applications. We have attempted a simpler approach.

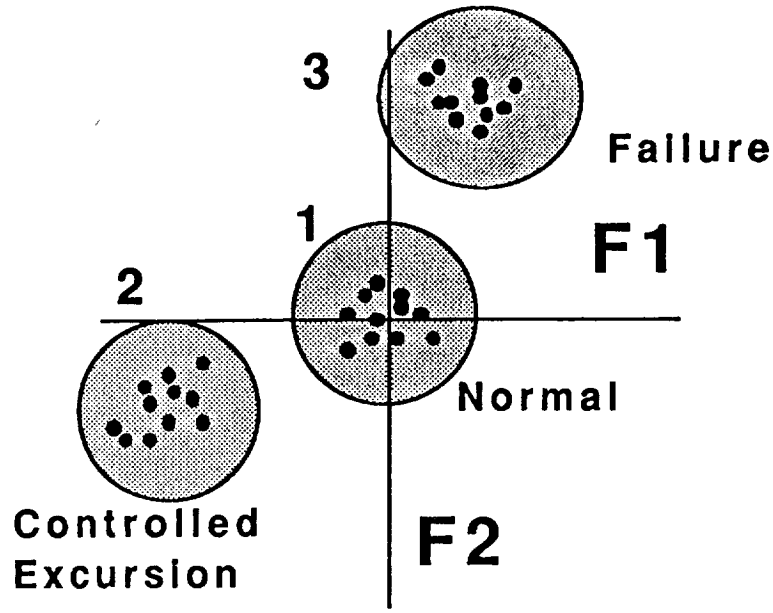


Fig. 8 Segmentation of the Feature Spac

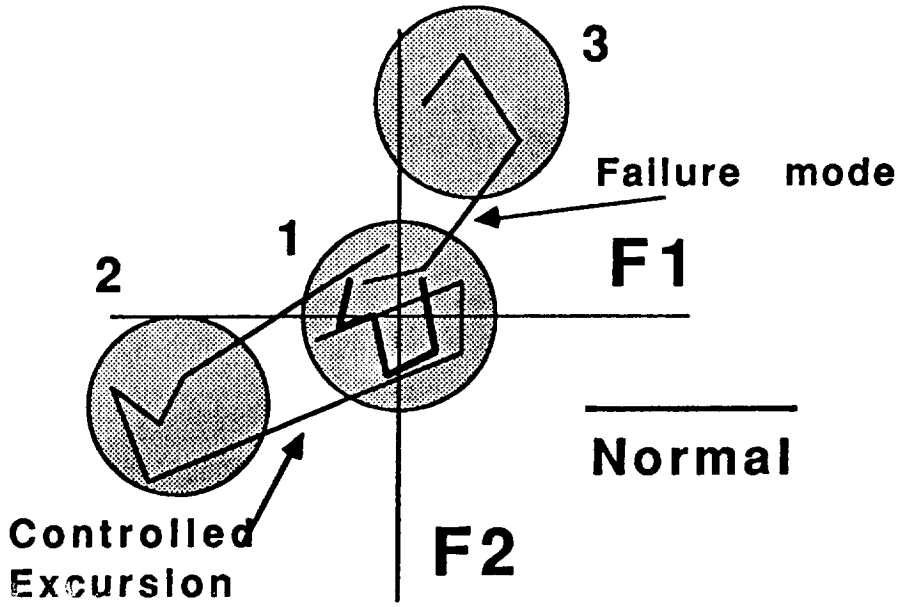


Fig. 9 Trajectories across the Clusters

ORIGINAL PAGE IS
OF POOR QUALITY

Table 2. Distance Measure and Error Bounds

Name	Expression	Relationships
Bayes error probability	$P_e = 1 - \int_{S_x} \max_i [P_i p(X\|w_i)]\, dx$	
		m Class Bounds
1) Equivocation or Shannon entropy	$H(\Omega\|X) = E\left\{-\sum_{i=1}^{m} P_x(w_i\|x) \log P_x(w_i\|x)\right\}$	$\frac{1}{2}[1 - B(\Omega\|X)]$
		$\le [1 - \sqrt{B(\Omega\|X)}] \le \frac{m-1}{m}\left[1 - \sqrt{\frac{mB(\Omega\|X) - 1}{m-1}}\right]$
2) Average conditional quadratic entropy [Vajda (1970)]	$K(\Omega\|X) = E\left\{\sum_{i=1}^{m} P_x(w_i\|x)[1 - P_x(w_i\|x)]\right\}$	$\le P_e \le [1 - B(\Omega\|X)]$
3) Bayesian distance [Devijver (1974)]	$B(\Omega\|X) = E\left\{\sum_{i=1}^{m} [P_x(w_i\|x)]^2\right\}$	$= K(\Omega\|X) = R_{\infty m} = \frac{m-1}{m} - M_0(\Omega\|X)$;
4) Minkowski measures of non-uniformity [Toussaint (1973)]	$M_s(\Omega\|X) = E\left\{\sum_{i=1}^{m} \left\|P_x(w_i\|x) - \frac{1}{m}\right\|^{s/(s+1)/s+1}\right\}$	$P_e \le \cdots \le R_{kNN} \le \cdots \le R_{1NN} \le R_{\infty m}$ [see Cover and Hart (1967) and Devijver (1974)]
		Two Class Bounds
5) Bhattacharyya bound [see Kailath (1967)]	$b(\Omega\|X) = E\{[P_x(w_1\|x) \cdot P_x(w_2\|x)]^{1/2}\}$	multicategory error: $P_e \le \sum_{i=1}^{m} \sum_{j=i+1}^{m} P_x(w_i, w_j)$;
6) Chernoff bound [see Kailath (1967)]	$C(\Omega\|X; s) = E\{[P_x(w_1\|x)^{1-s} \cdot P_x(w_2\|x)^s]\}$	$\frac{1}{2}[1 - [J_s(\Omega\|X)]^{1/s}] \le P_e \le \frac{1}{2}[1 - J_s(\Omega\|X)]$, for $s \ge 1$;
7) Kolmogorov variational distance [see Kailath (1967)]	$K(\Omega\|X) = \frac{1}{2}E\{\|P_x(w_1\|x) - P_x(w_2\|x)\|\}$	upper bound equals $\frac{1}{2}[1 - B(\Omega\|X)]$, when $s = 2$;
		$Q_{s+1} \le Q_s$; $Q_0 = 1 - B(\Omega\|X)$;
8) Generalized Kolmogorov distance [Devijver (1974), Lissack and Fu (1973)]	$J_s(\Omega\|X) = E\{\|P_x(w_1\|x) - P_x(w_2\|x)\|^s\}$, $0 < s < \infty$	
9) A family of approximating functions [Ito (1972)]	$Q_s(\Omega\|X) = \frac{1}{2} - \frac{1}{2}[E\{[P_x(w_1\|x) - P_x(w_2\|x)]^{2(s+1)/2s+1}\}]$	
10) The Matusita distance [see Kailath (1967)]	$\gamma = \left[\int_{S_x} (p(x\|w_1) - p(x\|w_2))^2\, dx\right]^{1/2}$	γ gives the same bound as $b(\Omega\|X)$;
		two-class bound relations:
		$P_e \le Q_s(\Omega\|X) \le Q_0(\Omega\|X) \le \frac{1}{2}H(\Omega\|X) \le b(\Omega\|X)$ [see Ito (1972) and Hellman and Raviv (1970)]

Notation: $\Omega = (w_i, i = 1,2 \cdots m;\ 2 \le m < \infty)$—a set of pattern classes; P_i is an *a priori* probability of class w_i; X is a n dimensional vector random variable; S_x is a sample space of X; $p(X|w_i)$ is a conditional probability density function; $P_x(w_i|X)$ is a posterior probability of class w_i conditioned on X; $f(X) = \sum_{i=1}^{m} P_i p(x|w_i)$—the mixture distribution; E is an expectation over S_x with respect to $f(X)$; $R_{\infty m}$ is an m class infinite sample nearest-neighbor risk; R_{kNN} is a k nearest-neighbor risk.

the steps employed in the above technique are;

-- definition of the features and construction of the feature space
-- plotting of ground test data (of both normal and failure tests) as trajectories in the feature space
-- segmentation of trajectories into failure and normal runs.

The technique that we have developed is based on a very simple observation that multi-dimensional single component data can be easily

represented by a set of radial lines in a polar coordinate system. Each polar line representing one sensor. The red lines for each sensor output can be normalised to two fixed values. These 'normalised red lines' in the polar system will represent two circles as shown in Fig. 10.

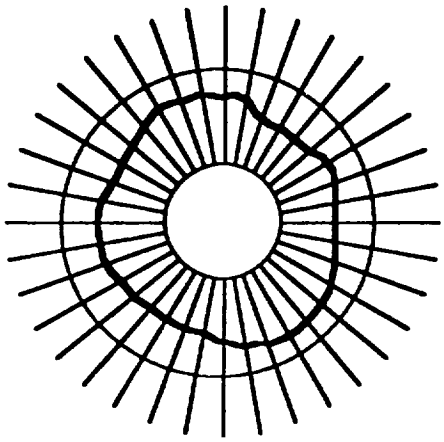


Fig. 10. Sensor data representation

The heavy line represents at some given time the sensor values normalised to the two circles as discussed above. A history of the sensor output can be made by stacking the polar graphs as shown in Fig. 11.

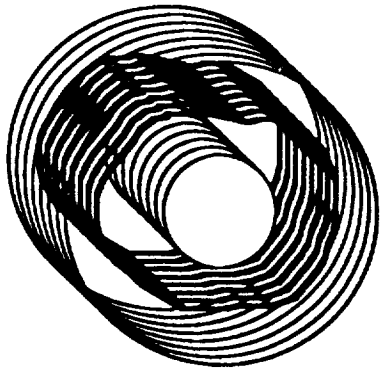


Fig. 11 Time stack of normalised sensor output

The stacked sensor output for each run would generate a solid. Each SSME run would thus generate a 'solid'.representing the entire history. It is expected that abnormal run solids will have some identifiable features which can be detected prior to failure. The computational calculations involved in this not very complex and hence it is possible to implement it in real time.

Implementation of these steps in practice is discussed nelow.

Results and Conclusions

the data from a run is stored on a number of magnetic tapes . the data for a short interval (10-100 sec.) and from a few imortant ssrs is combined into a single tape file. This tape file is read into a disk file which can be accessed by application programs. Fig. 12 shows the header, or the Run-Log, of the disk data file. This data, as can be seen from the first line, is for the time period 320 to 392 secs. of the run #901-364 which resulted in a failure. It also shows the ssr PID#, the engineering unit used and the SSME component mnemonic.

```
9010364R*11          320.00000        392.15000
       367  AP    HCC H.G. INJ PR
       940  GP    HPFP CLNT LN PR
       395  GP    HCC OX INJ PR
       410  AP    FPB PC NFD
       480  GP    OPB PC
       459  AP    HPFP DS PR NFD
       764  RH    HPFP SPD NFD
       854  GP    FAC OX FM DS PR
       858  GP    ENG OX IN PR 1
       878  GP    HX INT PR
       879  IC    HX INT T
       883  DP    HX VENT DP
```

Fig. 12 Data File Header

An interactive, menu driven program has been written to process the data and extract the features. In Fig. 13 a beginning MENU of the program is shown. Various types of operations are available by choosing the appropriate code. these operations include both recursuve and non-recursive filters, data compression, Logical Operations, FFT, Look-Up-Tables etc.

```
 3          395  GP    HCC OX INJ PR
 4          410  AP    FPB PC NFD
 5          480  GP    OPB PC
 6          459  AP    HPFP DS PR NFD
 7          764  RM    HPFP SPD NFD
 8          854  GP    FAC OX FM DS PR
 9          858  GP    ENG OX IN PR 1
10          878  GP    HX INT PR
11          879  IC    HX INT T
12          883  DP    HX VENT DP

 * * TYPE SEQ#'S OF COMPONENTS, END WITH 0
1
0
TYPE # OF DATA POINTS TO READ (LT.3000). 0=EXIT
3300
# # # # CHOOSE OPTION BY TYPING # # # #
 DATA COMPRESS * * * * * * =1
 NON-REC FLTR  * * * * * * =2
 RECURSIVE FILTER  * * * * =3
 NRM-OPRNS * * * * * * * * =4
 FFT * * * * * * * * * * * =5
 EXIT          * * * * * * =0
```

Fig. 13 Operations MENU

The first step is to read the raw data for a ssr by choosing the appropriate PID#. In Fig. 14 the data for PID#367, for the entire duration of 320 to 392 seconds is shown

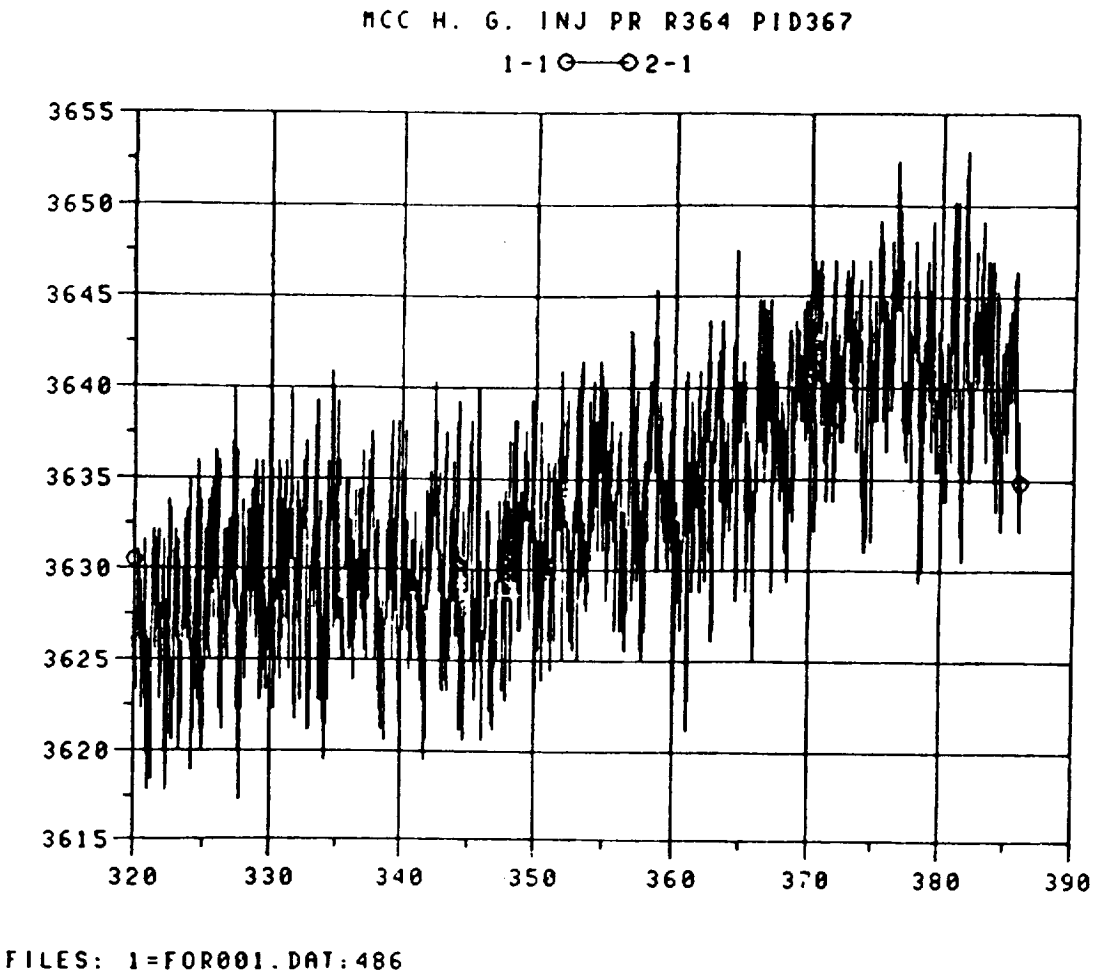


Fig. 14. MCC H. G. INJ PR, PID #367, RUN 901-364

From the above figure it is clear that the data has some structure in the form of some distinct features, however, the noise level is fairly high to mask them. the first step we have taken is to reduce the 'observational

sampling rate' through moving average. This is done in the following three steps;

1. Select a window size N = 0, 1, 2 . . . .

Let M = 2N + 1

2. Form signal averages $\underline{S_k}$ from the raw signal S_i

$$\underline{S_k} = \Sigma\, S_i/M\ ;$$

where k = N+1, 2N+1, 3N+1, . . .2rN+1, . . . and

i = k-N, k-N+1, . . .k+N

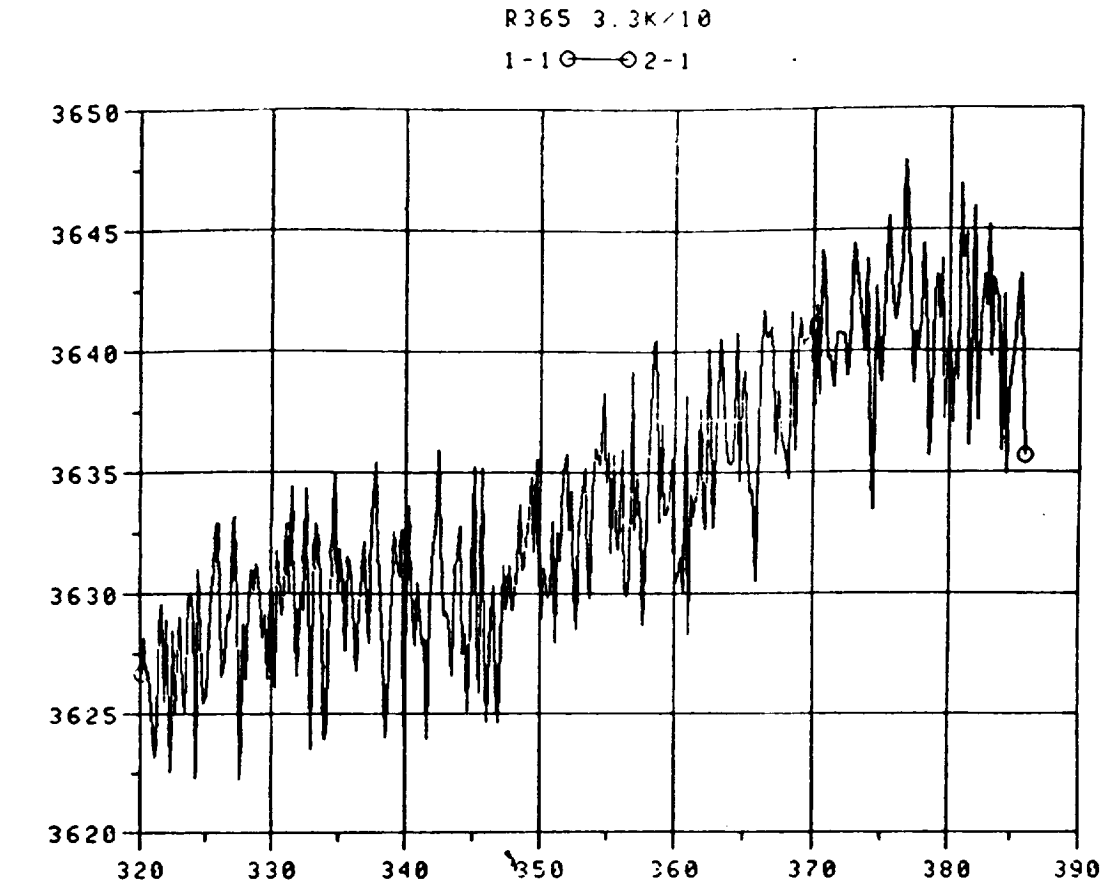
3. Replace the original signal S_i by $\underline{S_k}$. the sampling rate in the new signal, $\underline{S_k}$, is reduced by a factor of N.

In Fig. 15 the sampling rate of the data in Fig. 14 has been reduced by N=9, or compressed by a factor of 9. the program allows an interactive choice of N. the data in Fig. 15 still seems to have some noise which can be removed by various filtering techniques. As an illustration Fig. 16 shows the result of applying a non-recursive to the data of Fig. 15.

the data in Fig. 16 seems to have two distinct features, namely, a predominant frequency and a 'drifting background'. To separate these two components one can determine local averages over an interval larger than the hi-freq. wavelength as shown by the background line in Fig. 17. from this one can determine the zero-crossing points. A smooth curve can be fitted to these points to determine the background, as shown in Fig. 18.

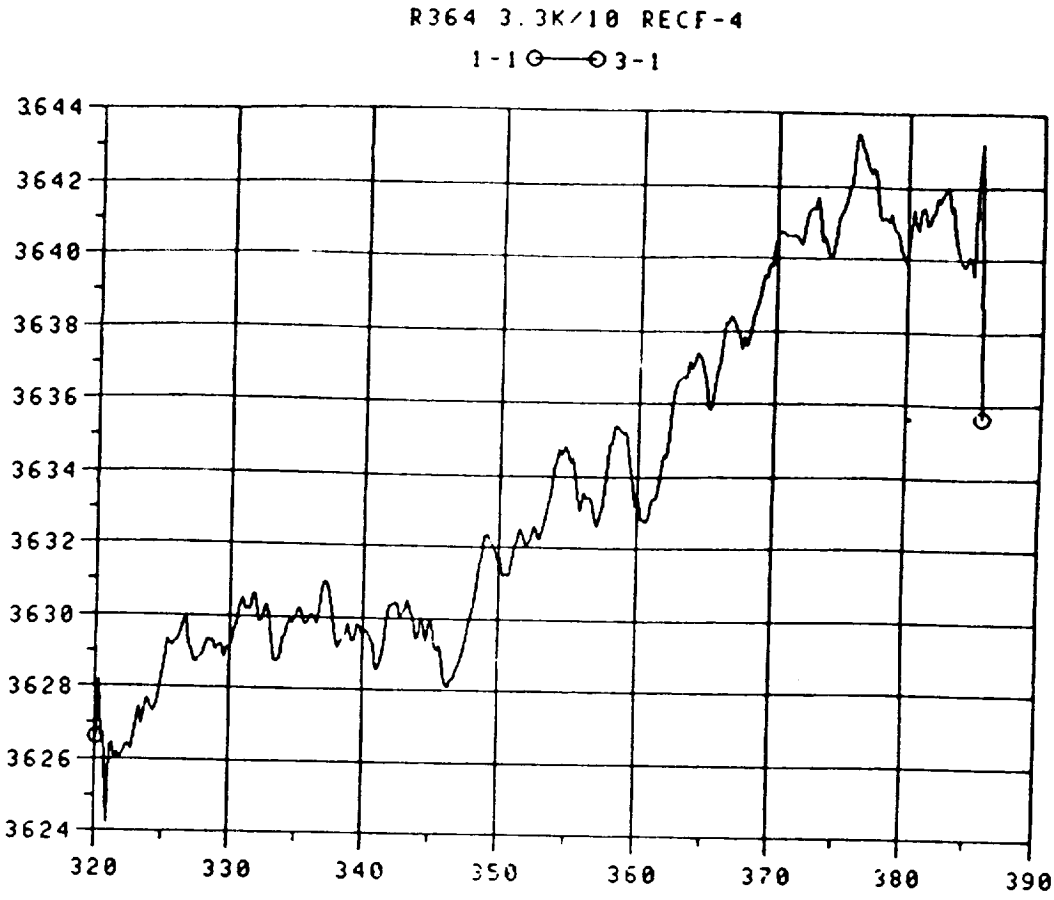
the background level, as found in Fig. 18 can now be subtracted from Fig.16 to give the hi-freq. component of the signal, as shown in Fig. 19. This signal can further be 'smoothed' to yield a 'cleaner' hi-freq. signal, as shown in Fig. 20.

Fig. 15 Reduced Sampling Rate Data, PID#367



FILES: 1=FOR001.DAT:495

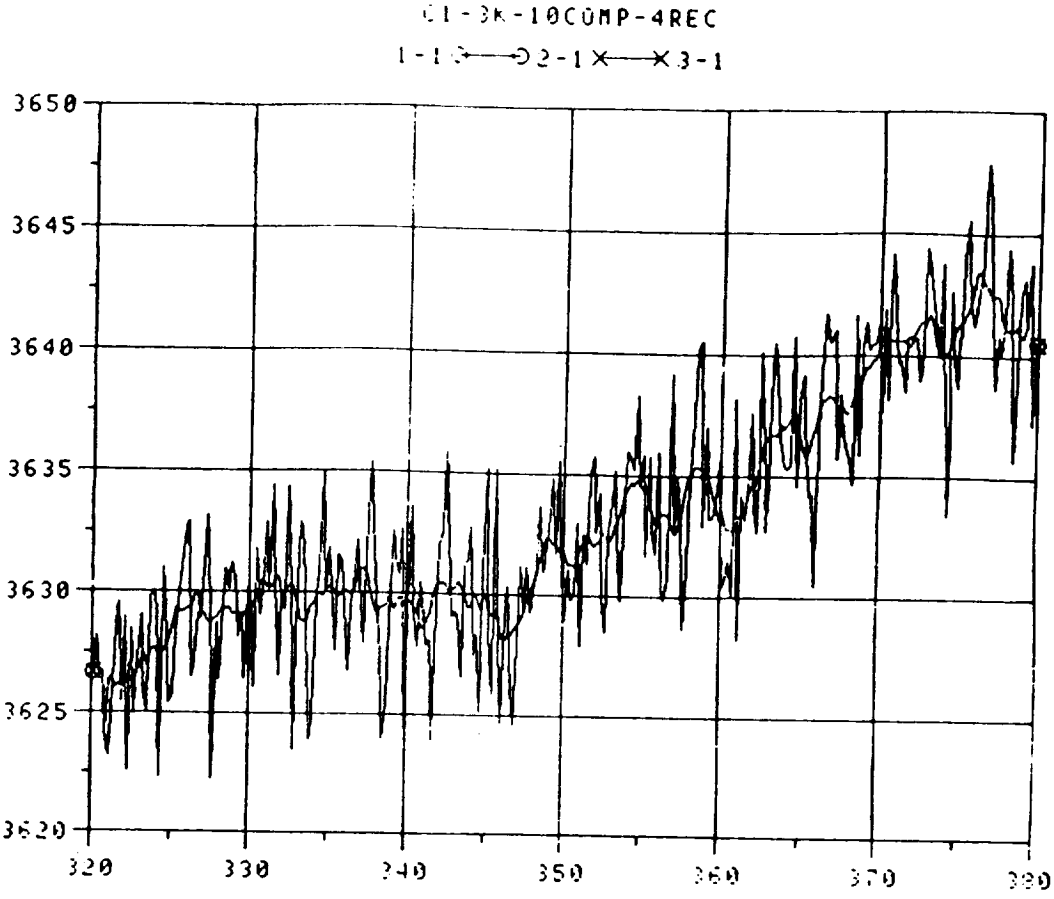
Fig. 16 High-Pass Filtered Data, PID#367



FILES: 1=FOR001.DAT;495

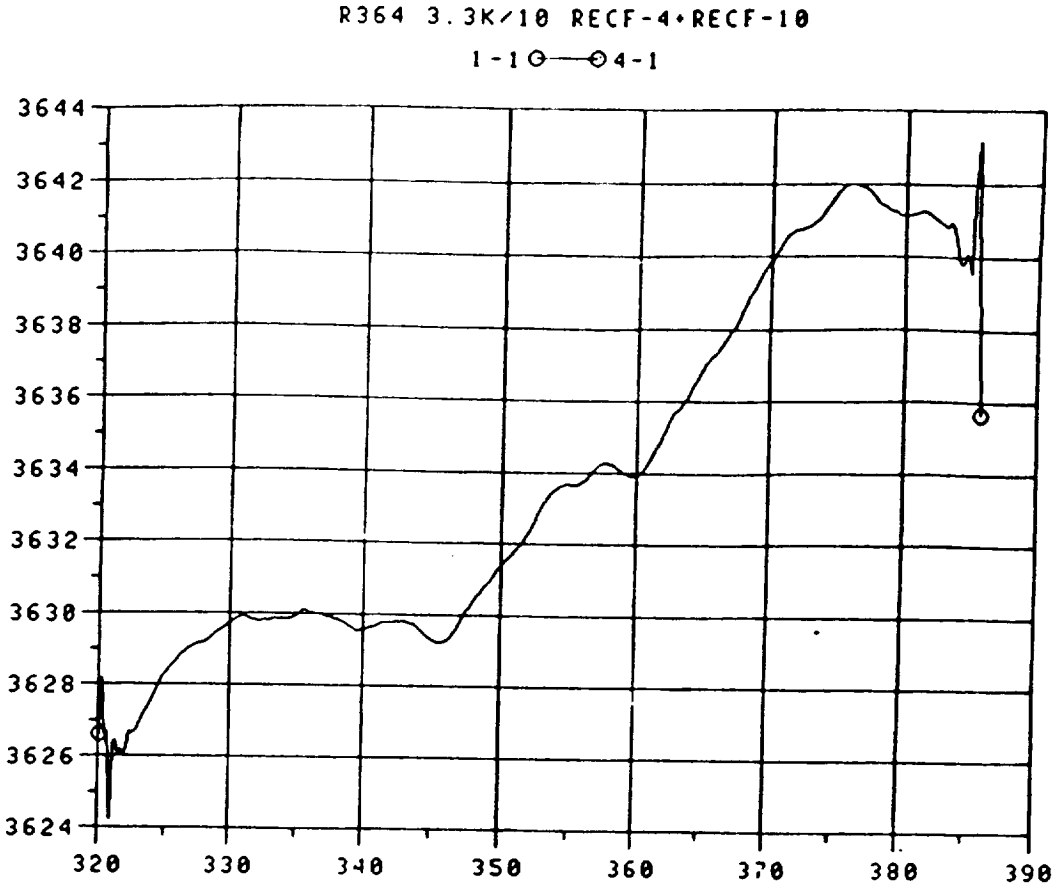
ORIGINAL PAGE IS
OF POOR QUALITY

Fig. 17 Zero Crossing Points, PID#367



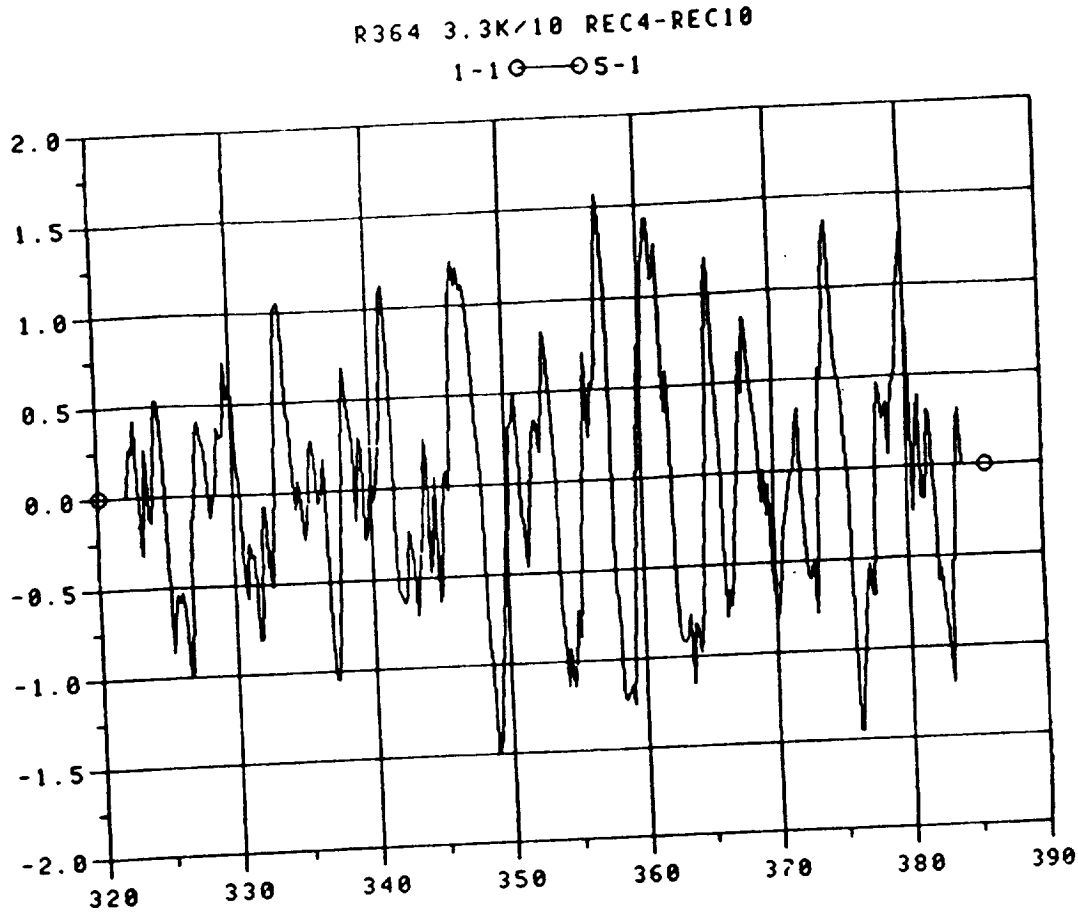
FILES: 1=FOR001.DAT;493

Fig. 18 Background Trend, PID#367



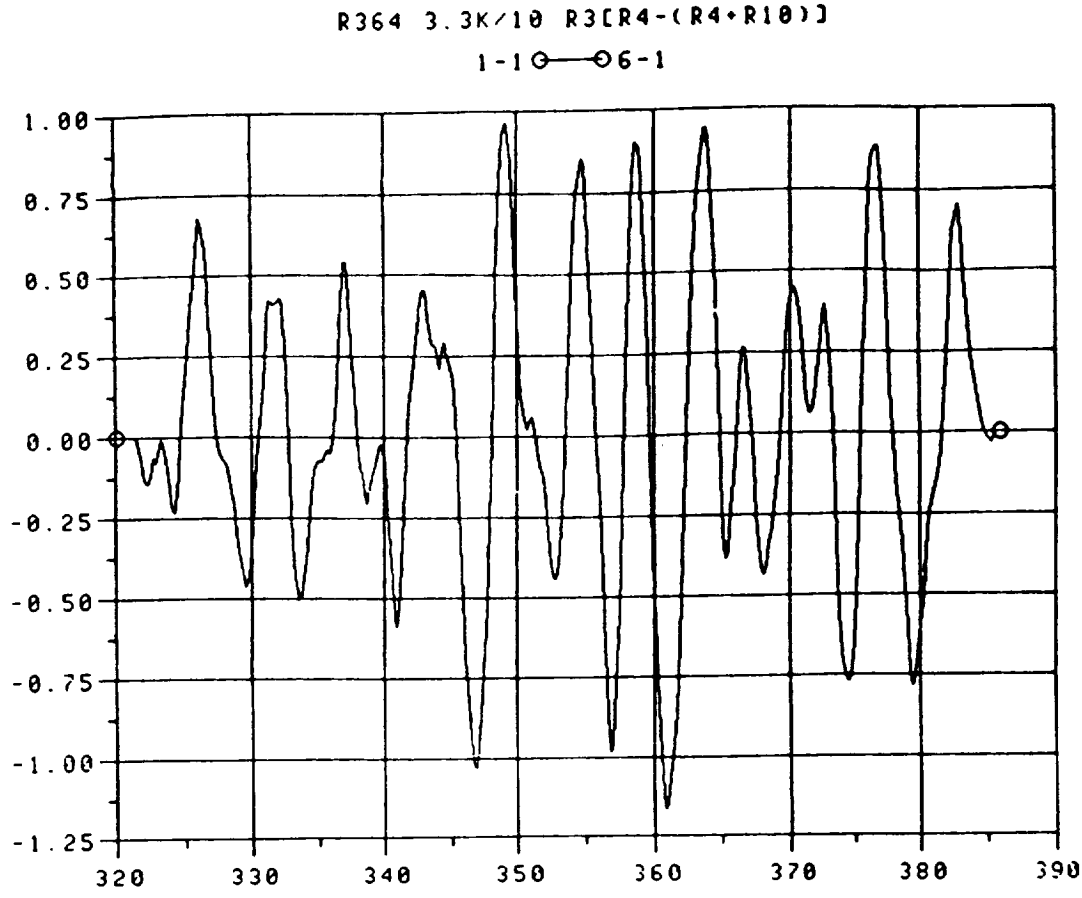
FILES: 1=FOR001.DAT:495

Fig. 19 High Frequency Data, PID#367



ORIGINAL PAGE IS
OF POOR QUALITY

Fig. 20 Smoothed Hi-Freq Data, PID#367



FILES: 1=FOR001.DAT:497

the original signal of Fig. 14 can now be said to have two distinct features as represented by Figs. 18 & 20. the former is a slow drift with with plateaus, whereas the latter is a high frequency jitter, which, if needed, can be Fourier analysed. This drift and the high frequency can now be taken as features for representation in the feature space as discussed earlier.

This technique, though developed earlier (8) and outlined here for the sake of completeness, was not tested for real time applications. We have now made repeated attempts to test its feasibility as a real time code. it has not been possible to realise fast execution, as compared to the data collection time 20-40 ms.

Fig. 21, the 'stacked data representation of a simulated run on a Supercomputer Ardent/Titan. shows the evolution of a run from time t = 0 to some final value. Variations in sensor output as also the collective behaviour of all the sensors can be seen at a glance. This technique is much faster than the other two developed earlier and discussed here. The real data could not be used for it since the tape data was in a different Format and could not be accessed by the Supercomputer

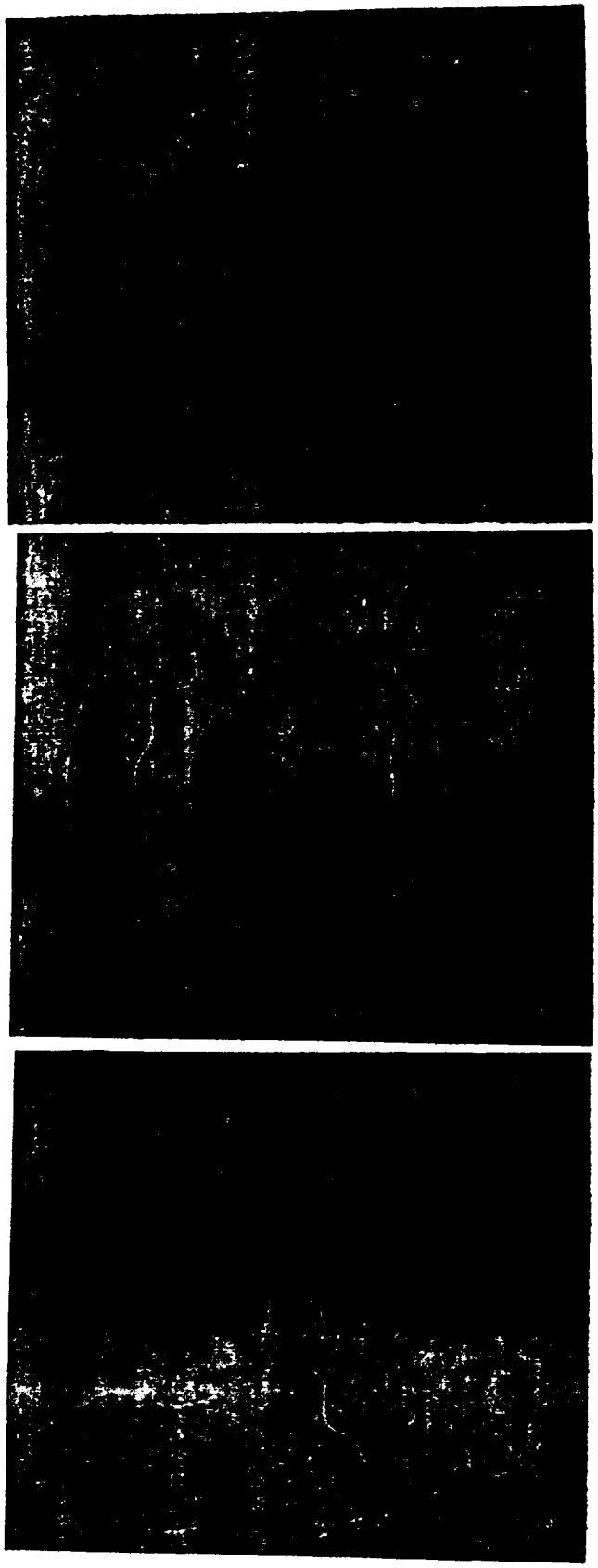


Fig. 21 Evolution of 'data stack' in time.

ORIGINAL PAGE IS
OF POOR QUALITY

Conclusions

From this study we conclude that a real-time system for the analysis of SSME data, based on the data visualisation scheme introduced here can be realised.

-- A feature space description of the SSME ground test data has been realised.

-- Two different approaches have been attempted, one of which is selected.

-- A real time implementation of the selected approach has been made.

-- Simulated data tests give very encouraging results.

-- A design of a more comprehensive program has been made to;
A. Survey a large number of normal runs (about 50), and
B. Survey all the failed runs (27) and compare them with the above.

-- Considering that an overall comprehensive review of neither the normal nor the failed runs exists it is highly recommended that an analysis environment of the type discussed above, should be implemented.

References

1. SSME Controller
R. E. Mattox and J. B. White
GSFC/NASA Technical Paper. 1932 (1981)

2. Failure Control Techniques for the SSME.
Final Report, NAS8-36305, Rocketdyne Division.

3. 'SSME failure characteristics with regard to failure detection'
T. C. Evatt, L. R. Iwanicki, M. H. Taniguchi and H. A. Cikanek III
Adv. Earth-to-Orbit Propulsion Technology, NASA Conf. Proc. 2436
MSFC/NASA May 13-15, 1986

4. SSME Failure Detection
H. A. Cikanek III
Proc. Am. Control Conf. June 19-21(1985)282

5. Controls, Health Assessment and Condition Monitoring
H. A. Cikanek III
Proc. Am. Control Conf. June 18-20(1986)1943

6. 'DEAN: A program for dynamic engine analysis'
G. C. Sadler and K. J. Melcher
NASA TM 87033, July 1985

7. 'Patterns in Pattern Recognition'
L. Kanal
IEEE Trans IT-20(1974)697

8. SSME Control & Diagnostics
A. Choudry
NASA/MSFC Summer Faculty Fellowship Report 1987