https://ntrs.nasa.gov/search.jsp?R=19840003440 2020-03-21T01:46:32+00:00Z

TR-10-83

N84-11508

# A CONCEPTUAL STUDY OF AUTOMATIC AND SEMI-AUTOMATIC QUALITY ASSURANCE TECHNIQUES FOR GROUND IMAGE PROCESSING

Prepared for:

National Aeronautics and Space Administration
Goddard Space Flight Center
Greenbelt, Maryland

Under Contract Number
NAS 5-27513

by:

Engineering and Economics Research, Inc.
1951 Kidwell Drive
Vienna, Virginia

September 30, 1983

## FOREWORD

This report summarizes the results of a study conducted by Engineering and Economics Research (EER), Inc. under NASA Contract Number NAS 5-27513. The study involved the development of preliminary concepts for automatic and semi-automatic quality assurance (QA) techniques for ground image processing. EER was supported by MRJ Incorporated, as subcontractor, in this study and we acknowledge the valuable contributions of Dr. Edward McMahon in this effort.

EER acknowledges the valuable assistance of Mr. Joseph Heinig of NASA/GSFC, the contract technical officer. In addition, helpful comments, constructive criticism, and useful guidance provided by Mr. Paul Heffner, Mr. Fred McCaleb, and Mr. Gerald Grebowsky of NASA/GSFC during the course of this study were very useful.

TABLE OF CONTENTS

TABLE OF CONTENTS (Concluded)

LIST OF FIGURES

LIST OF TABLES

## 1.0 SCOPE

This report addresses automation of quality assurance and quality assessment techniques in a scientific sensor data processing system. A distinction is made between quality assessment and the more comprehensive quality assurance which includes decision making and system feedback control in response to quality assessment.

The philosophy of automated quality assessment (QA) is the main subject of this study. Some examples of automated QA are given, but they are speculative. It is difficult to give attractive and feasible techniques without concentrating on a specific system design; however, the principles espoused herein as philosophies need to be recognized at the beginning of the design of future systems which attempt to incorporate automated QA.

### 1.1 Summary

An automated QA system should be:

- o Designed integrally with the processing system. Access of the QA function to data, knowledge of input data quality, and having test sequences available are requirements for the QA function design.

- o Easily managed. Trends, failures, and status of changes to the system to improve quality or correct failures must be logged and tracked through a management information system.

- Maintained and modified. the QA function should accept new algorithms and modifications as quality problems emerge from a maturing system. Having access to all data paths and a library of simple measures will allow "artificial intelligence" concepts such as learning to be studied.

There are obviously different levels of QA and different amounts of automation which may be used to implement QA.

The minimum level of QA is output product inspection. With this level, little more can be done than to reject bad products, request regeneration, and report the error to some maintenance or repair function. Failures of a production line are detected by operators or by lack of output and are handled the same way. Higher levels of QA are inspection of intermediate products, trend analyses on product quality parameters, and running and analyzing test data. As with failure reports, warning reports are generated for the maintenance and repair function.

Automating these QA functions requires:

1. Sensing the parameters used by a QA analyst.
2. Emulating the algorithm used by the QA analyst to reach a decision.
3. Generating failure or warning reports; tracking progress and closeout of outstanding reports.

Steps 1 and 3 can be accomplished easily within the state-of-the-art.

Step 3 is a management information system which is not difficult to implement (or purchase and modify).

Step 1 requires that the QA function have access to the pertinent data. This may be difficult in certain cases, but should be feasible if considered in the initial design of a system.

The major difference between a human analyst QA function and an automated function is in Step 2. A human analyst is adaptive.

An improvement can be made to some existing QA functions simply by providing automated Steps 1 and 3, but it is also possible to program known, planned QA functions such as checks on data bounds, format, data counts, trends, statistics (destriping), etc. In addition, as new QA evaluation functions are discovered ( as a result of, say, failures or degradation of some system component), they may be added to a repertoire of QA algorithms.

It is difficult to program a general adaptive process. One approach to an adaptive process is a general learning program, which is "taught" or trained by an experienced analyst as to what is "good" and what is "bad" output. Such a program would have to maintain many statistics on each data flow. Candidate statistics could be max, min, mean, mode, second and third central moments, correlation between data items, and autocorrelation function over a fixed length sample.

Under training of a "bad" output, a learning program would correlate differences in statistics between good and bad data to the definition of "bad." It appears more cost effective at this time to plan for assisted

automated QA rather than attempting (and having to rely on) adaptive fully automated QA functions.

Figure 1-1 presents the concept of an integrated quality assessment system in a generic data flow process. The pertinent features are that the quality assessment function is not done piecemeal throughout the system, it has access to all data, and it has input to the process control function to enable quality assurance.

The presentation of the QA function as a single function does not preclude implementation of various subfunctions in distributed processors, perhaps coresident in the processors implementing various main-line functions in the system, but any QA processes must communicate to each other or to a supervisory QA function and a QA management information system.

Access to the data is obviously necessary, and the processing system must be designed with this access and overhead considered. The access does not have to be constant, and should be under the control of the QA process. The amount of data needed to be processed by the QA function will vary depending on system status and history. (For example, a low sample rate would be appropriate when the system has a history of nominal behavior). Therefore, the ability to supply data should be designed in the processing system; the ability to select data for assessment should be built into the QA function.

The most fruitful area for automated QA is in trends analysis on data from many points in the data flow. Catastrophic failures are simple to

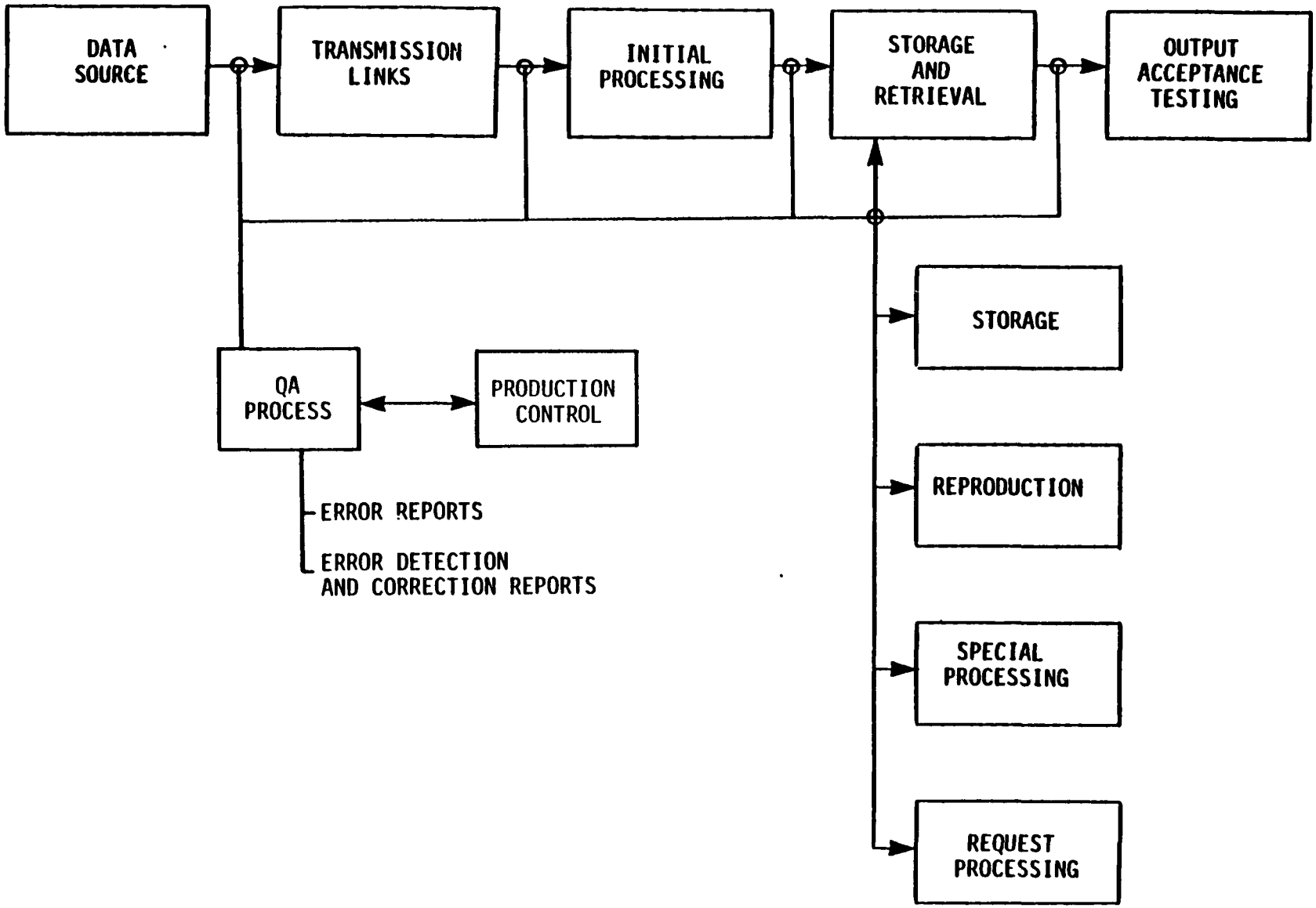


FIGURE 1-1. GENERIC DATA FLOW PROCESS WITH CENTRAL QA FUNCTION

detect, and random unimportant bit errors can be tolerated ( and are expensive to detect). Day-to-day QA requires algorithms to detect when quality measures cross some threshold between acceptable and unacceptable.

When quality measures are computed regularly, it is easy to perform trend analyses, and to anticipate or predict when some quality measure may be approaching a threshold of unacceptability. Appropriate action may then be taken to correct the cause of the trend.

The algorithm used to compute quality measures may be based on knowledge of the process and quality measures, or may have to be general statistical computations from which an adaptive function can "learn" differences in acceptable and unacceptable products.

Knowledge-based algorithms require more design and analysis, and less computing time than adaptive functions. Knowledge-based algorithms reliably give answers according to the rules specified (which may be incomplete or incorrect); adaptive functions can be "trained" to follow changing standards of quality (at some lag in time), but give the desired answer with some probability less than one. Crop recognition algorithms are good examples of adaptive algorithms.

The concept of automated quality acceptance and assurance presented here is treated as a system-level function. The interfaces between QA and the Data Processing Functions are complex, and QA would benefit by being designed in conjunction with the Data Processing Functions. However, the utility of an automated QA system depends on the use made from its

output by the Production Control Function, so there is cause to include QA in Production Control. The treatment of the QA process ultimately depends on the philosophy of the system sponsor, and, in any disposition, relies on system engineering to properly integrate and trade QA system costs and benefits.

## 2.0 DEFINITIONS

### 2.1 Quality Assessment and Quality Assurance

The term Quality Assurance is generally accepted to mean the program within a system which ensures that the quality of the product meets minimum standards. Implied in this term is the measurement of quality of the product, the decision to act or not to change the system, and a means of changing the system to adjust the quality of the product. These three components of Quality Assurance will be referred to as:

1. Quality Assessment
2. Production Management Responses
3. Corrective Action

Quality Assessment is the kernal of the problem being studied. It can be a manpower intensive effort if capabilities to automatically monitor product quality are not in place.

Production Management Responses to a measure of product quality are usually constrained by policy, and production management is expected to be able to recognize or discover what faults cause unacceptable quality and what adjustments, corrections, or repairs are necessary to restore quality output. Production management may have the option of deciding to ship or reprocess marginal quality products based on cost, time to reprocess, and knowledge of customer requirements.

The corrective actions which may be taken to adjust the quality of the output are system specific. Such actions as repair of failed components, realignment and recalibration, cleaning, and special processing are included in this class of actions in addition to actions such as control parameter adjustments. Software errors and operator mistakes require corrective action.

Corrective action is generally not under control of Quality Assurance even though Quality Assurance requests corrective actions. Corrective Actions may require an engineering change request, software modification with configuration control approval, repair, and maintenance action, procedural change, or personnel training. Other (usual) corrective action requires simple parameter adjustment. This is considered to be feedback control and must be included in the specific system implementation. Quality assurance includes feedback control; quality assessment does not.

This study will concentrate on automation of quality assessment and will use the acronym QA to refer to quality assessment.

## 2.2 Automated QA

To distinguish between automated and semi-automated QA, automated QA is defined to encompass all QA done in the system without operator intervention. Thus, the assessment of quality determined by an algorithm leads to product acceptance and rejection, reprocessing orders, and repair orders. This is a reasonable goal but the state-of-the-art in artificial intelligence will not now reasonably support this goal for a new system. The problem

lies in the probability that a new system will experience unexpected errors and the difficulty in generating a complete set of product acceptance criteria. The need to automatically control and correct the process for unanticipated errors requires an adaptable, learning algorithm. But for a stable, well understood process, a reasonably complete set of product acceptance bounds can be established and simple production management decision rules can be programmed, closing the loop to the process controls and extending automated quality assessment to automated quality assurance.

An assumption in the above paragraph is acceptance of failures when unexpected errors occur (but the likelihood of such errors is small because the system is stable and well understood). Thus, some supervision is needed. Extending the concept of supervision leads to the definition of semi-automated QA.

## 2.3 Semi-Automated QA

Semi-Automated QA is defined to be operator assisted automated computation of quality measures rather than a direct measure of quality. This is certainly part of automated QA and may included estimates of quality. The intent in this definition is to emphasize the participation of an operator or production control personnel in the process of QA as well as in production management, and to allow the easy addition of new assessment algorithms. A semi-automated QA system has more flexibility to adapt to a changing or maturing system and more ability to tolerate a middle ground of QA - where estimates of quality meet neither criteria for acceptance nor for rejection, and judgement is required.

## 2.4 QA of System Components

The focus of a QA system is the output product, and it is appealing to consider a QA scheme that only measures observables in the output product to judge whether or not an output product is acceptable. This is too narrow, and presumes a completeness of knowledge of output product observables which is probably not the case. Moreover, given a quality failure, no indication of the cause of the error is known. Therefore, QA of system components and intermediate products should be performed.

### 2.4.1 QA of Hardware Components

It is presumed that all hardware in a system has been accepted by tests against a specification, and QA of hardware components may be performed by exercising certain of the acceptance tests, modified as appropriate to work in a production environment. QA of hardware components (in addition to obvious failure determination) will support trend analyses of parameters such as processing time, operator interactions, feedback control parameter values, power, and expendable usage (if appropriate). Study of the trends or correlations of trends with failures may lead to prediction capabilities to avoid failures.

### 2.4.2 QA of Software Components

Software does not degrade over time as does hardware. There are no situations where software works properly at one time and fails later, but there are various conditions where it may appear that the software is acting

in a random fashion (in error). Typical cases occur when input data or parameters are outside expected bounds and no limit checking is performed. Another condition may be produced by combinations or sequences of data values or events not anticipated in specifying, designing, or building the software. Still, the software is deterministic and will give repeatable results. There are hardware faults that may make it appear as if the software failed (such as a bit error, memory failure, or I/O error). To protect the software itself against hardware errors, it is prudent to have copies of software and to periodically verify -- not just recopy -- the operational software against an archived copy to detect any errors in the operational code.

Quality failures in software, "bugs", are hitherto undiscovered errors. These errors exist due to failures in specifying the software or inadequate acceptance testing the software. (There is an argument that inadequate testing may be more cost effective than complete testing if the QA and maintenance processes are inexpensive, or schedule is a driving function.) Both types of failures can occur on the initial build of the software or can result from changes made to the software (fixing one bug may uncover others, or the fix might ignore critical interactions elsewhere in the process). It is obviously important to have adequate acceptance testing of changes to software.

Failures in specifying the software can result from the specifications being incorrect or incomplete. Incompleteness may result from an unforeseen combination of events or from presuming (or needing) the software to compensate for failures or tolerances in other components. It may be argued that

the latter cases are not specification failures, and no specification can afford to be complete regarding every possible failure mode and combination of conditions but, in any case, a change to the specification must be made to resolve these issues if they occur.

At the output product level it may not be possible to distinguish between errors caused by hardware or those caused by software errors.

## 2.5 QA of the Process

The quality of the output product, given the design of the processing system, is a function of the hardware and software components discussed above, the data and the control parameters. The data will be discussed in detail in later sections, but it is necessary here to state that the quality of the input data must be known if proper quality assessment and assurance is to be performed. Input data quality must be known to assess quality failures, to order reprocessing or to accept poor output as the best possible, ot to adjust control parameters to optimize processing.

The data contains calibration or reference data which is used to compute control parameters. Usually, calibration data is designed to be usable even at poor signal to noise ratios or high bit error rates (e.g., step functions with many samples per step, linear ramp functions, etc.). Separate checking should be performed on calibration data as recieved to assess the consistency and reliability of the calibration source.

other control parameters are set by the operator or by calculations

performed on the data (including feedback from measures of output data). The proper use of data -- sensors, telemetry and calibration data -- depends on the hardware and software, leaving errors by the operator to be discussed.

All control parameters set by the operator should be recorded, not to assess blame, although such records will locate causes of quality errors, but to be analyzed to determine if changes in procedures or training are necessary.

Analysis of operator controls will also reveal which controls could be easily automated.

## 2.6 Measures of Quality Assessment

The fundamental measure of quality is whether or not the output product meets specifications. The fundamental measure of QA is, therefore, what fraction of product sent to users meets specification, and what is the cost of performing the QA.

Cost can be measured in various ways, such as system availability, throughput, average time in process, and overhead (poor quality product, test, maintenance, calibration). The appropriate measure depends on the mission of the processing system. A system designed for rapid response runs to a different criteria than a system designed for bulk throughput, or one designed for custom processing.

User demands and system guarantees set the percentage of output product which meets specifications.

## 3.0 GENERIC SYSTEM

### 3.1 System Description

Figure 3-1 presents a block diagram for a general sensor processing system. It is principally a serial processing system for some set of standard products. These standard products are generated according to a specification on some schedule based on receipt of data. In addition to the standard product, an archival record of the data is kept and is used as a source to fill custom requests for special processed data.

The data source in the block diagram represents the sensor and all processing to format the data, and includes calibration and telemetry data. Transmission links include all processes up to the receipt of the data (sensor, calibration and telemetry) at the processing facility. It is at this point, the input to the initial processing, that a measure of the quality of the data should be made. Some indication of quality may be available from the transmission links subsystem to augment or identify sources of any error, but a quality measure of the input data is necessary as a reference for later error identification.

The initial processing block in the diagram contains all standard product processing. The result of initial processing is an archival copy of the data, which may also be the standard output product.

Typical initial processing which would be included in an image processing system would include calibration, reformatting, geometric and radiometric

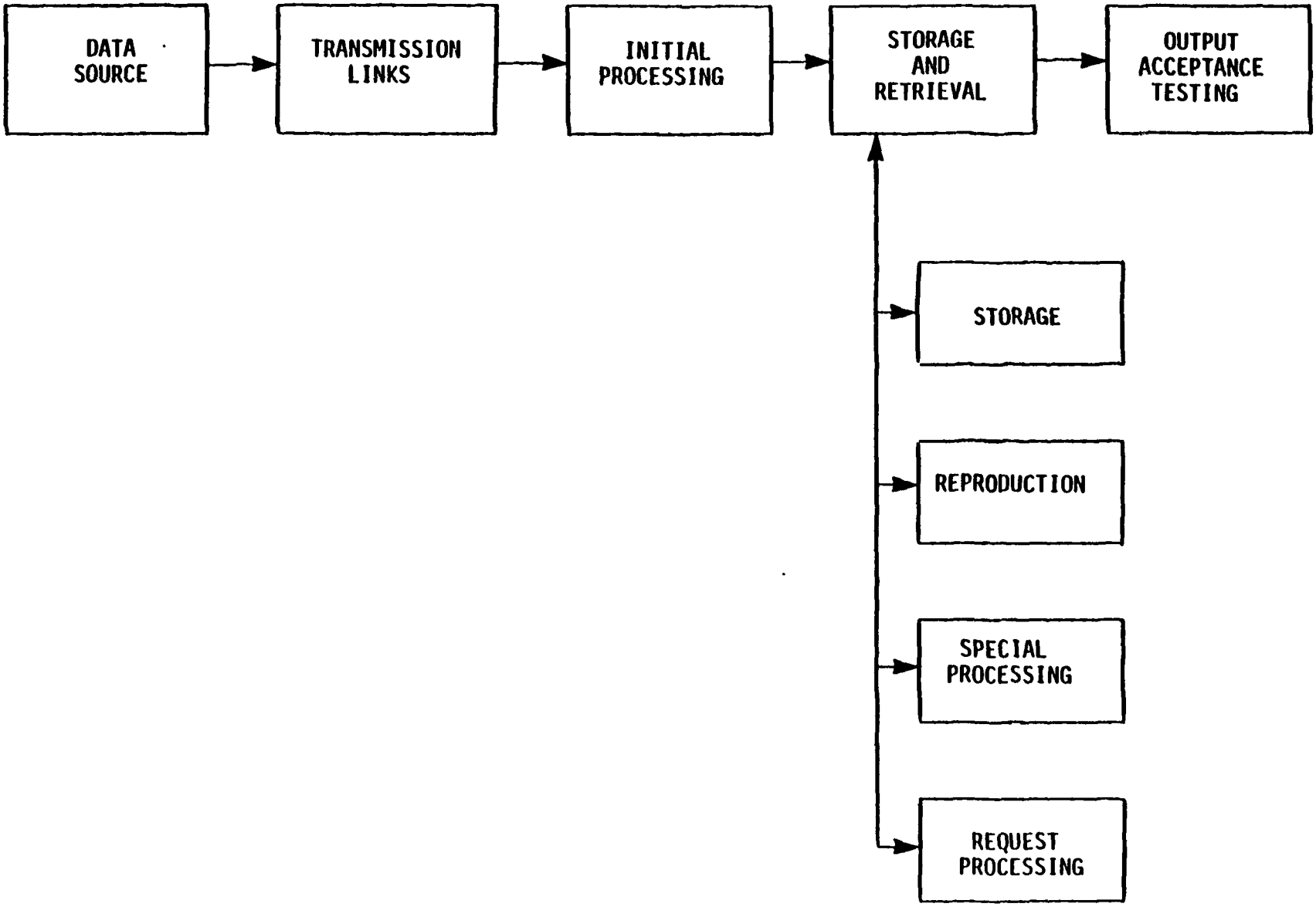


FIGURE 3-1 GENERIC DATA FLOW PROCESS

correction, and annotation or association of ancillary data with sensor data. A more complex process might also include registration to some standard projection, and destriping and missing data estimation. Note that regiatration (which involves resampling), destriping and missing data estimation (say, for failed detectors or scan line length variations) is a form of error correction or quality assurance. If these functions are included as image processing functions, then they are not considered quality assurance processes as addressed in this report.

Special or custom processing requires identification of the necessary processing, and retrieval, processing and reproduction of the data. Note that this chain includes the processing of the request as well as the processing of the data.

## 3.2 Sources of Error

### 3.2.1 Data Source

All sensors operate in an environment which produces a signal-to-noise ratio (SNR) which is a measure of sensor data quality. Estimates of the SNR at the input processor may be made by a QA process not only to know the input data quality but to be used in trend analysis of the sensor performance. SNR estimates may also control later processing and may be used as feedback control to command sensor system parameters (such as gains, filters, data compression, etc.).

Since most data appears random at first glance, and since the data

passed through a transmission system, estimating the SNR may be difficult. One means is to compare the power spectrum of the input data to expected power spectra. Transmission system performance can be estimated by analysis of calibration and other fixed format data (sync patterns, fill data, etc.).

Failures in sensor channels should be recognizable from trend analyses. However, conceiving an automated response to every imagined sensor system failure does not appear cost-effective. Automated measure of trends and alerting abrupt changes or threshold crossings certainly is feasible and recommended.

### 3.2.2 Transmission Links

Most transmission links provide error detection and (at least some) error correction. These give measures of system performance. As mentioned above, calibration data and fixed format data can be used to estimate BERs and data drop-outs, but these errors may not be attributable to the transmission links, as errors could arise in the sensor system. Schemes such as retransmission of data from a remote receiving site or transmission of test data can resolve some of these issues.

### 3.2.3 Initial Processing

This process contains the main processes in the system. The output is the archival data and usually the "standard" product. The processes mentioned in 3.1 above will be addressed in turn.

### 3.2.3.1 Calibration

The calibration data must be identified and associated with the correct sensor data. the quality of the calibration data should be assessed and trend analysis performed. Sudden changes in calibration data is cause to suspect the process or the sensor.

Post-calibration data analysis (averages and variances) should be sufficient to assess the quality of the calibration process. Low frequency sampling appears capable of maintaining QA on the process if trend analysis is done on calibration data.

### 3.2.3.2 Destriping

The need for destriping indicates incorrect or insufficient calibration. It is, in fact, a means of calibration.

If destriping is done (in lieu of calibration per se) trends should be maintained on the correction needed in each channel as if it were calibration data. Sudden changes or threshold crossings are cause for notification of system error.

QA of the destriping process is probably best done with test data. An alternate is a second pass of destriped data which should suffer no change. This, however, is implementation algorithm dependent.

### 3.2.3.3 Geometric and Radiometric Correction; Registration and Reformatting

These processes are variations of resampling of data. (Radiometric correction may be considered calibration or destriping.) There are two major aspects to resampling: calculating the resampling parameters by analyzing the data and performing resampling.

Calculation of resampling parameters may be done from ancillary data (telemetry, stored parameters from independent analyses, required output formats) in an open-loop fashion or from analysis of data content (ground control points, other correlation). These calculations can be checked by "reasonableness" checks and by sampled analysis of the processed data. For example, ground control point residuals can be checked; correlation of registered data with the registration base can be made and correlation values analyzed.

The actual resampling process can be monitored by processing test data and comparing output to analytically derived results. Example test data are constant data, linearly varying data, and fixed frequency data.

### 3.2.3.4 Annotation

Computation of annotation from telemetry or registration processes is difficult to verify. Trend analyses or sampled checking by an analyst may be the only reasonable choices. Indications of annotation error may be found by other processes - especially those in 3.2.3.3 that depend on the content of data also used for annotation.

### 3.2.4 Storage and Retrieval System

#### 3.2.4.1 Storage System

Typical errors in a storage system are data deterioration and lost data (misplaced, mislabeled or not in storage). The degree of data deterioration may be estimated from existing studies or by including fixed test data in the stored data. Redundant coding and storage techniques could be used if this is seen as a problem.

Lost data is usually an operator problem and should be minimized by a good library management system. Data identification should be well distributed in the data to prevent misidentification.

#### 3.2.4.2 Reproduction

Errors in archival masters are copied to duplication masters. These in turn are copied to production data, and bit errors are added at each duplication step. Sepcific system constraints will dictate the reproduction process. QA measures of errors requires measures of errors in the master and independent measures of errors in the product. Test strips and fixed format data are useful in monitoring this process, but it is important that the test data be such that it will be treated identically to actual sensor data.

QA of the reproduction process could be the most valuable QA in the system, and should be relatively easy to automate.

### 3.2.4.3 Special Processing

Special processing is system dependent. If a special process is offered as a standard service to be requested, it should be treated as the initial processes for QA, except cost-effectiveness values of automated QA are different from those in initial processing. If a special process is a customized assemblage of available processes or algorithms, then the QA will probably be an individual assessment perforce.

### 3.2.4.4 Request Processing

Proper record keeping and a good data management system should hold errors in requests, custom processing parameters, and data identification to a minimum. Independent checks against order copies and verification or orders to customers are means to identify errors in request processing. QA of this process would probably be a manual process, and complete records of such errors should be kept to determine if changes in procedures are needed.

## 3.3 Quality Assessment System

Automating a quality assessment function requires:

1. Sensing the parameter used by a QA analyst.
2. Emulating the algorithms used by the QA analyst to reach a decision.
3. Generating failure or warning reports; tracking progress and close-out of outstanding reports.

The most difficult step, QA algorithms, will be addressed first.

### 3.3.1 QA Algorithms

Algorithms for QA can be classified as those required to assess a known quality parameter (such as striping between detectors or deviation of calibration data from nominal values) and those which adapt to evaluate quality measures not anticipated in the design phase of the system. Both classes of algorithms are covered in some detail in Section 4, and both classes of algorithms fit under the broad umbrella of "Artificial Intelligence" (AI) in common usage. AI successes at present are in knowledge based systems (or expert systems) in which a body of knowledge is codified as a number of IF-THEN statements in a fixed hierarchical structure. An example of such a construct in QA could be:

IF (calibration bias - normal is less than e) THEN (set variable A to TRUE) ELSE (set variable A to FALSE)

IF (estimated bit error rate is between TABLE (I,1) and TABLE (I,2) THEN (set variable B to TABLE (I,C)).

.

.

.

IF (variable A is TRUE and variable B is greater than x) THEN (issue quality warning flag 1).

This example makes the point that the QA algorithms, called AI or not, must be designed for a specific system. The knowledge coded into the algorithm is quite specific.

Adaptive algorithms produce values for clauses or terms in the above IF-THEN statements. For instance, a system might "learn" in a training mode what values are appropriate as bounds in a table as in the second IF-THEN statement in the above example. Such a system will have mistakes in QA analysis unless the bounds are very stable, and until the bounds are "learned."

As a system matures, additional tests and algorithms can be implemented in the QA process. These would reflect additional knowledge gained about the operation of the system.

### 3.3.2 Sensing parameters for QA

It is generally impossible to state completely which parameters or what data is needed for QA by the principle that known sources of error can be avoided; the unanticipated sources of error will cause problems. For that reason, we state that access to all data should be provided for the QA process during system design, and the QA process has the choice of whether to read a specific data element or not. While it is easy to make such a statement, it is not easy to justify excessive costs to make high volume or high rate data continuously available for QA. As in any system design, trades must be made between cost and amount of data access, but QA access to data must be taken into account during such trades. This is further addressed in Section 5.

It is presumed that parameters observable by a QA analyst can be computed from available data and control parameters. This may not be simple,, and can be viewed as "feature extraction" (also addressed in Section 4). For instance, "zipper" in an image produced from digitial data when bit synchronization is lost for part of the line is obvious to an operator but is not trivial to find by computation.

It is strongly recommended that access to data and control parameters throughout the system be included in the system design.

### 3.3.3 Failure and Warning Reports

In addiition to the obvious process management function supported by the issuance and trackings of QA failure reports, the QA process should use history of failure to condition data processing until the fault is corrected. Knowledge of faults in a process should be used to modify processing schedules and maintenance. Knowledge of faults in data should be used to request reprocessing, or retransmission, or to issue disclaimers on output quality from flawed source data.

### 3.3.4 Example QA Process

Figure 3.3-1 presents an example implementation of QA process in a scientific data processing system. Only part of the system is shown; additional stages of processing may be to the left and right of the portion shown.

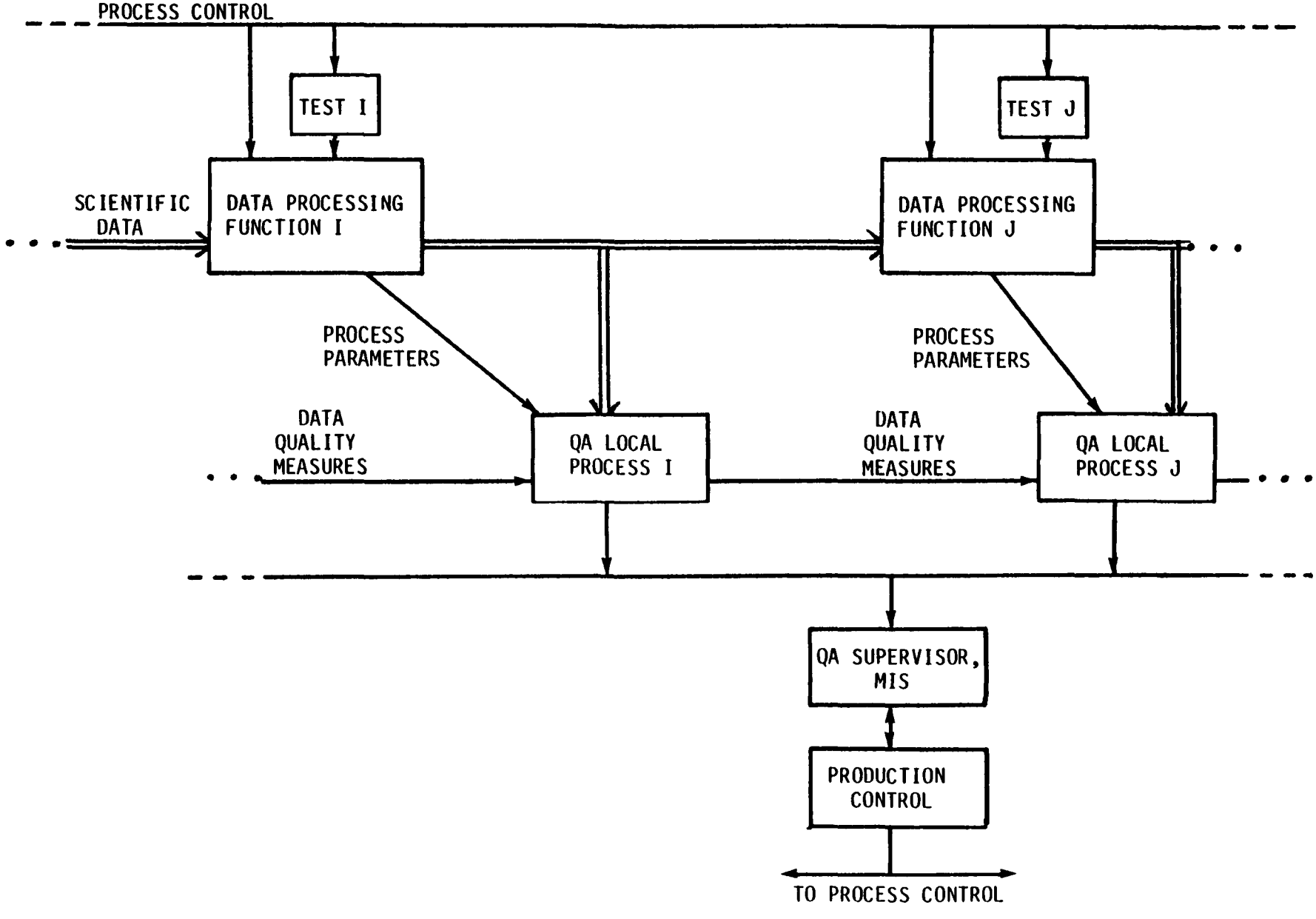


FIGURE 3.3-1. Example QA Process in a Generic System Fragment

The QA process consists of a number of local processes and a QA supervisor process, shown implemented with a management information system. The QA processing shown as being separate might be implemented in one central processor or the local processes might be integrated with their associated data processing functions. The diagram emphasizes that local processes (which may implement the same or similar algorithms) are applied to various data paths emphasizes that the quality measures computed by the local QA processes are fed forward for use in computing later data quality and for use in assessing performance of the data processing functions, and the quality measures are fed to the central or supervisory QA function. The design allows tests to be run in response to requests by the QA function through production control when testing is necessary to evaluate the performance of a data processing function.

The results of the QA process are available for use by production control in scheduling and controlling the processing of data.

## 3.4 Data Types

Data subject to QA in a scientific sensor processing system can be categorized as sensor data, test data, calibration data, support data and management data. The latter, management data, is concerned with ordering data, requesting special processing, shipping, controlling inventory, etc. The QA of management data will be left to a MIS or review by appropriate personnel. Each of the four technical data types is discussed below.

### 3.4.1 Sensor Data

Sensor data is, by its nature, random, but important statistics of the data must be known for the processing system to be designed and built. Such statistics include maximum values, max variations, precision, variance, and number of smaples per operating sequence. These statistics can be computed for input data and, in some designs are necessary to know to process the data. For instance, mean and maximum values and the variance can be used to destripe image data. The same measurements should be maintained to calculate calibration system performance. Some scaling of these parameters may be invariant through processing, and calculation of these parameters would provide a QA check on the output product. An example would be the normalized power spectrum of each input sequence in an image processing system. The transfer function of the processing system should be known (some MTF compensation might be implemented) and the spectrum of the output should bear a known relationship with the input.

The variance of the data normalized to the dynamic range could be another effective monitoring measure. It is strongly recommended that trends of the statistics be maintained to monitor the sensor input and the product output. Sampling the statistics would keep the volume of trend data manageable.

It is necessary to know the quality of the input data in order to determine the achievable quality of the output product. Without this, it will not be known whether the processing system degraded the data, or if special processing or reprocessing could improve the data.

Since the input data abears random, the quality of the input data cannot be directly measured (except for obvious measures such as data counts). The statistics may yield estimation of the quality of the input data but other sources such as calibration data, bit sync performance and error detection and correction codes can yield good measures.

Imbedding reference data in sensor data should be considered in a system design. Calibration data, synchronizing sequences, controlled data during retrace or border scanning in a scanning system or the like can be used for quality checks on the path up to the input of the processing system in addition to the purposes for which such data is generated.

Continuing the same concept, it can be worthwhile to process such data as border data and calibration data as equivalent test sequences through the processing system. It is important, of course, that such data not interfere with sensor data processing by skewing the statistics and impacting control parameters.

### 3.4.2 Test Data

Test data is constructed to see if specifications of a system are being met. Test data usually stresses a system, and usually includes data which are out of the expected bounds for a system. Results of processing test data, in an entire test sequence, should be unambiguous. Specific failures should point to specific processes. Subsets of test data should be used periodically to check system performance. Trends of the results should

be monitored so system performance can be predicted, and system performance can be calibrated after a repair or upgrade.

Lacking other measures, test sequences would have to be run on a frequent schedule to provide QA measures. This presents an overhead to the system throughput which must be considered in system design.

Another design consideration is the necessity of providing injection points for test data throughout the system. It is generally unsatisfactory to have to rely on a single end-to-end test sequence to isolate a quality problem.

### 3.4.3 Calibration Data

In this context, calibration data will encompass all fixed and known format data. This includes the synchronizing sequences, null words and similar data discussed in 3.4.1. Within some limits, this data is known. For example, calibration data may consist of 100 "black" samples, a ramp of fixed rate to "white", and "white" samples to produce a total of 1000 samples. The QA process should calculate and record for trend analysis the level and noise on the black and white levels, the slope (length) of the ramp, the variation of the ramp from nominal (e.g., straight line or logarithmic), and data counts. It is not difficult to compute adaptive quality thresholds for some statistics of the calibration data a given confidence level. Then, as long as the statistics remain within these threshold bounds (and the correct statistics are used), the input calibration data can be considered of good quality. As the example presented in Figure 3.4-1,

the x% confidence thresholds can be tighter than the specified thresholds. Whether they are or not, if the statistics cross the threshold, a quality warning should be generated. This should be done even if the data is still within specifications, since it warns of a trend which should be monitored and explained.

This same general technique should be applied to all statistics for which trends are maintained.

### 3.4.4 Support Data

Support data includes telemetry and operator-set parameters. Processing parameters and annotation data will likely be derived from support data, so the quality of the support data is important. Another example of support data is ground control points used to generate coordinates for matching points in imagery. Poor quality ground control points will lead to poor quality geometric corrections.

Each kind of support data must be examined to determine the properties that reflect quality. With this, "reasonableness" checks, trend analyses and statistical measures may apply. The spacecraft operational control functions should not be overlooked as sources of quality measures of telemetry data as they are responsible for knowing the quality of telemetry data for purposes of spacecraft control. It may be cost effective to have that operation control function monitor sensor as well as spacecraft telemetry.

The sensor data processing function uses spacecraft data such as attitude and attitude rates to annotate and sometimes process sensor data. Knowledge of "pure spacecraft" data (such as thermal and electrical status) should be available for correlation with data quality. Presumably, necessary telemetry points will be requested by those responsible for monitoring sensor status, but the potential use of these and other telemetry points in QA analyses should be recognized when a limited choice of telemetry points are chosen for the system.

Misalignment of support data with sensor data is obvious enough not to be overlooked in system design. Unique tagging both sensor and support data is usually simple. In cases where the alignment must be calibrated, it should be monitored for correctness.

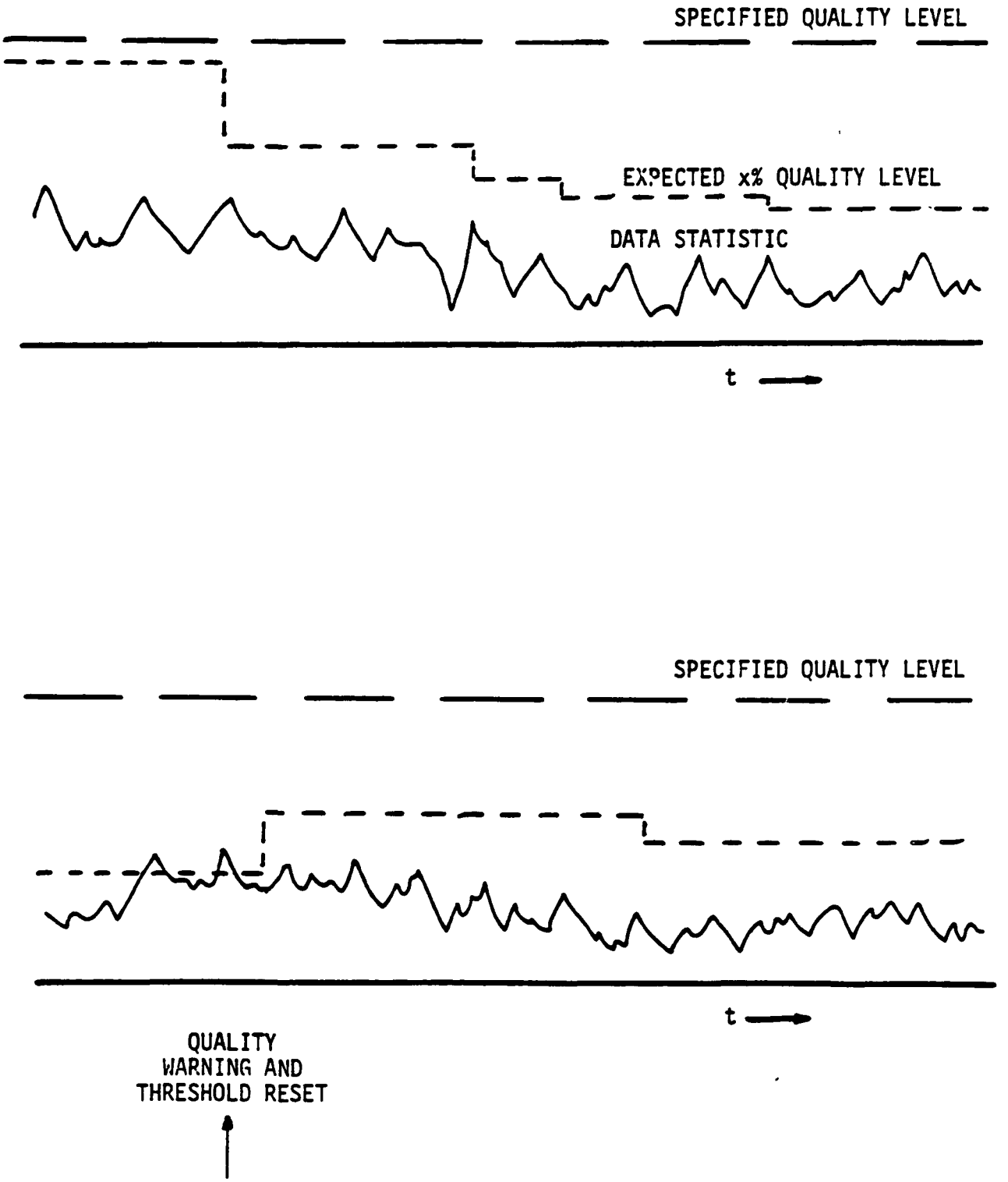


Figure 3.4-1. Example of Adaptive Threshold Monitoring

## 4.0 AUTOMATED QUALITY ASSURANCE

The first step toward automated quality assurance is assisted (or semi-automated) quality assessment. The components of either system are:

- o Data access
- o MIS
- o Automated measure of known functions
- o Ability to adapt (or be adapted) to new functions

### 4.1 Data Access

It is obvious that without data, there can be no quality assessment. The kinds of data are discussed in Section 3.4

Quality errors or faults can be considered as random errors, total failures or "out of calibration" performance. Failures are generally easy to detect with a few samples from a data stream (either the data stops, the data is a constant, or the data is noise with no relation to the correct statistics). Random failures are difficult to detect. As mentioned above, catastrophic random errors need to be detected, and reprocessing will usually solve the issue, but infrequent data value errors should be tolerated to a limited extent. One reason to tolerate inconsequential errors is the extreme difficulty in detecting them. Errors causing catastrophies are similar in their randomness, but are detectable (by definition -- a catastrophy must be easily observable). Specific systems and specific definitions

of catastrophies will determine the level of data processing needed for detection of such errors.

The remaining class of errors are "out-of-spec" or "out-of-calibration" errors. These, which can be avoided by adjustment or maintenance, are the standard fare for a QA process, and cost trade-offs should be done primarily with the control of this type error in mind. To control this error, trend analyses and prediction of error conditions should be designed.

To be cost-effective, sampling is recommended. This is discussed in Section 5.1. Enough data must be sampled to calculate reasonably confident projections and from enough paths in the system so that fault isolation or identification can be done.

## 4.2 MIS for Reporting and Tracking

A typical ground image processing system has three types of information components within it.

- o image data
- o support data
- o production control messages from management

Efficient flow of management information (e.g., work orders, work schedules, error reports, log entries, report listing, etc.) is critical since it has an obvious impact on the processing, movement, tracking, and quality of the image data as well as support data. Clearly, the efficient

operation of the production processor depends on the efficient management communications and implementation of a central QA function in the data flow process. Additionally, such a QA function must have access to data to derive, extract, or compute measures of quality which, in case of bad output products, will be provided to the production management for it to take appropriate corrective action (see Figure 4.2.1).

Based upon state-of-the-art, it is difficult to design a general adaptive QA process. Consequently, only some QA functions can be fully automated while others may have to only semi-automated or largely manual. Furthermore, it is imperative that a sufficiently long history (or knowledge) be maintained and made available to the QA function for the purpose of 'training' those QA functions which are to be automated.

Sensing of parameters to be used by the QA functions can be easily automated. Such parameters may include, among various others, checks on data bounds, formats, data counts, trends, statistics such as mean, variance, max, min, etc., calibration parameters, geometric correction parameters (resampling parameters). These and other useful parameters should be properly tagged and stored on-line for quick access on an as needed basis. It is noted that these parameters are sensed at various stages during production processing. Furthermore, it is important that these parameters be sensed in manner without interferring with the normal data flow or data processing in the production processor.

The task of automating the QA algorithms themselves is a complex one since such algorithms will have to be adaptive just like human QA analysts.

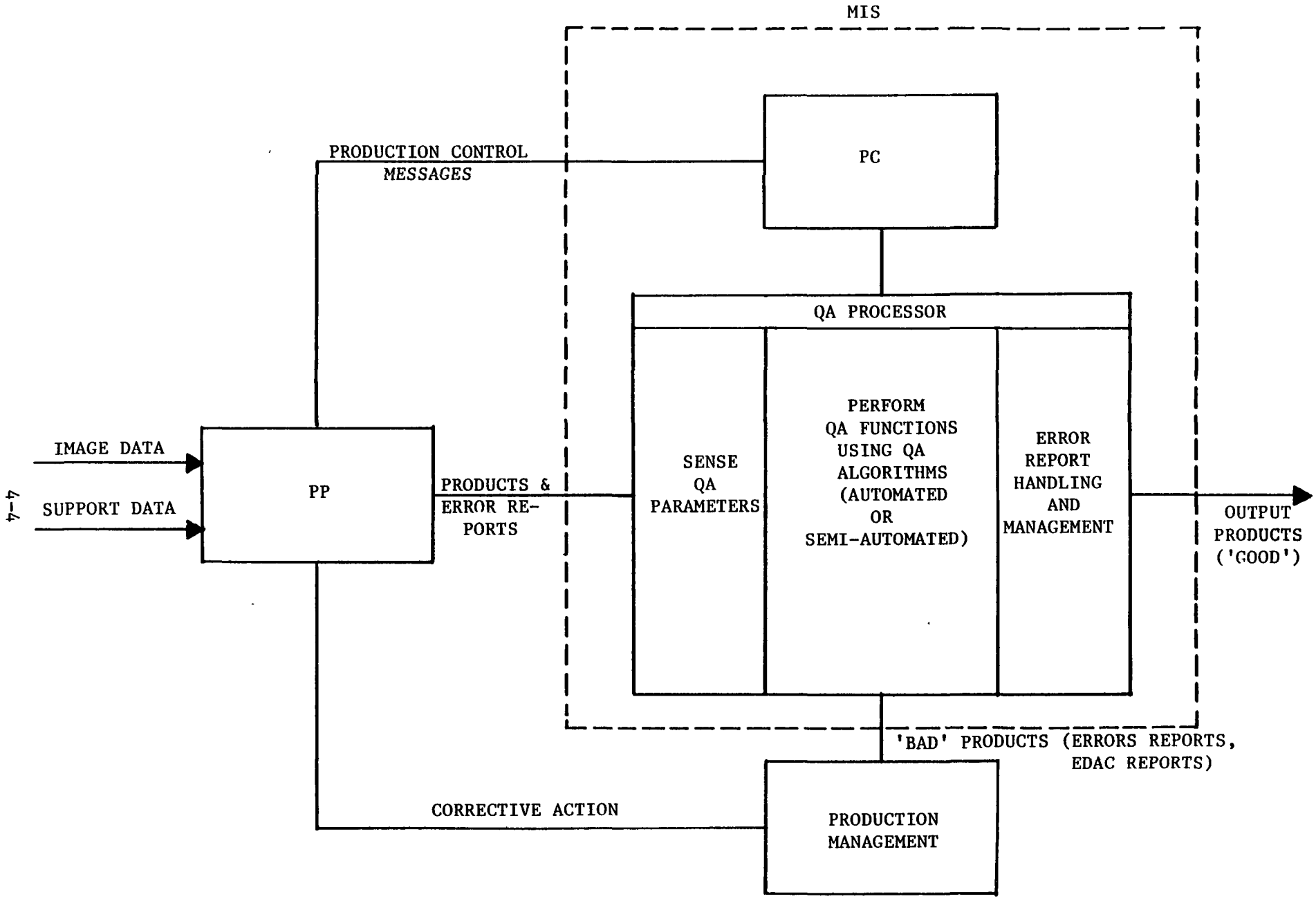


FIGURE 4.2.1 A MIS FOR GENERIC IMAGE PROCESSING SYSTEM

Simple QA algorithms such as checking data bounds, data counts, calibration, BER measurements, etc. can be automated. Furthermore, these QA algorithms can be implemented in near real-time without interrupting the normal production processing.

However, more complex QA algorithms (like the ones to be discussed in Section 4.3.2) are based upon artificial intelligence (AI) or pattern recognition (PR) techniques. Most of those AI/PR algorithms can be automated. The performance (success or failure) of these will depend largely upon:

- availability of a large sample for each QA parameter
- selection criteria used in extracting each QA parameter

Extreme care would be necessary in implementing such QA functions to avoid interferring with or impacting the performance of the production process itself. Additionally, some of these QA functions may be highly process intensive thus making implementation in near real-time unlikely. Consequently, some of these QA functions may have to be performed off-line.

Another important function of an auotmated QA processor is to generate error reports (including failure and warning reports), tracking progress, and closing out of outstanding error reports. From past experience with systems such as Landsat Image Processing Facility (IPF) it is clear that error reports represent a large portion of the management message volume. Error reports are usually generated by operators on various subsystems and then sent to respective subsystem production control groups within the IPF. It is a very people-intensive effort from generation to correction/

resolution. As many as 1000 error reports/month are generated during some months and as many as 700 may be unresolved at any given time. The effect of this on system throughput is significiant since the travel of error reports through the system is less than efficient.

In brief, considering the points/problems mentioned above in manual handling of error reports, it is absolutely essential that future image processing systems use on automated on-line management information system for generation, collection, evaluation and resolution of error reports. Streamlining of the error reports related functions will undoubtedly result in less processing delays thus yielding a much improved total system output. System-aided resolution tracking would also result in better control of outstanding error reports and, in addition, it may allow efficient planning around erroneous data by the production controller during scheduling. This would reduce the error-scatter effect. Resolution tracking would also permit rapid information on previous dispositions for identical or similar error reports.

A management information system (MIS) consisting of a local management network (such as 'Ethernet'), a job tracking and error report handling computer system, and applications software will prove to be essential for future ground image processing systems. It is possible to incorporate other functions such as sensing of QA parameters and application of QA algorithms in the MIS mentioned above and shown in Figure 4.2.1. In the final analysis, however, the design constraints and cost-effectiveness should determine the optimal design of MIS.

## 4.3 Pertinent Functions of AQA

### 4.3.1 Known Functions

These functions are known; specifications exist for which tests and evaluation algorithms can be designed. Some examples of Automated QA on known functions are given in the following paragraphs.

#### 4.3.1.1 Calibration

Calibration data is used to transform sensor data to some fixed reference. To Use calibration data, some processing is done to calculate reference parameters (black level, gain factors, linearity, etc.) to high accuracy and precision, even in the presence of many data errors. It is likely that calibration data is smoothed (or filtered or averaged) in this process. With very little extra effort, the calibration processing function can produce measures of the smoothing (variance, extremes, residuals, etc.) for use by the QA process. Thus, QA can have measures of the quality of the calibration data, and trend analysis will determine if a particular set of cal data is anomolous or not, and if slow degradation of the cal process is occurring QA is extended to quality assurance when processing parameters or decisions to reprocess with different or differently estimated cal data are determined based on the QA of the cal data. Errors in calibration data can be due to the calibration source or the data path.

If large varying corrections to a specific data channel are being performed, the quality of the data in that channel should be suspect.

Later processing may be improved by ignoring or deweighting contributions from this channel to processing parameter computations.

4.3.1.2 BER

Bit error rates provide a measure of system performance. It can be measured if known data is used as a reference. In an operational situation, it may not be possible to presume that data such as sync words or calibration levels are, in fact, known. A composite error can be determined and, by observing the random properties, this error may be allocated between the data source and components in the transmission path.

Test data and error detection and correction codes are more reliable and accurate sources for BER measurement.

4.3.1.3 Destriping

Destriping is a specific example of the potential use of quality assessment.

Images may be produced from data collected from a number of detectors designed to scan adjacent strips in some field of view. That is, parallel scan lines in the reconstructed image come from different (adjacent) detectors. Differences in responsivity of the detectors will cause the reconstructed image to appear striped, so the differences in responsivity are removed by calibration (having the detectors view a known or common source). If

the responsivity model is not accurate enough or if the calibration scheme does not work as aniticipated, detectable stripes will remain in the output image. Destriping is a method of removing relative differences in adjacent lines by assuming that statistics of the data (mean and variance ) in adjacent channels should be identical.

Quality assessment should measure striping if there is a specification regarding line-to-line variation. The existence of stripes can be estimated by computing the same statistics used to correct striping, or by some other method such as a power spectrum analysis. (Computing the power spectrum of an image in the direction orthogonal to the scan direction, say by a fast Fourier transform, would show if a spike of energy existed at the scan frequency. This should be attributed to striping if the data used has remained in the digital domain. Data scanned from an image could show the scan frequency due to printing spot size errors or line spacing variations.) If the power spectrum measure is used for QA, no data is available for correction striping but an independent measure is known. If the same statistics are used to measure the quality of destriping as were used to destripe, then the QA process measures the implementation of the destriping process and must rely on the analytical correctness of the statistical destriping process to actually remove stripes to specification levels.

### 4.3.2 Adaptive Functions

A pattern recognition system is generally composed of the following elements:

- Input pattern
- Environment
- Feature extraction
- Decision (classification) algorithm
- Adaptive or Learning mechanism

The word "adaptive" in pattern recognition is generally used to mean the strategy of feature extraction and classification algorithms which can be changed flexibly according to the state of the input patterns and its environment, with the additional function of learning.

The simplest approach for pattern recognition is probably the method of "template-matching" where a set of templates or prototypes, usually one for each pattern class, is stored in the machine. The input pattern (with unknown classification) is compared with the template of each pattern class and the classification is based on a pre-selected matching criterion or similarity function. In other words, the input pattern is assigned to the pattern class whose template it matches the best.

The main disadvantage of the template-matching approach is that it is somtimes difficult to select a good template for each class and also to define a proper matching criterion. This difficulty is especially remarkable when large variations and distortions are expected in all patterns belonging to one class.

Consequently, a more suitable approach is to classify based upon selected measurements extracted from the input pattern (see Figure 4.3.1). These

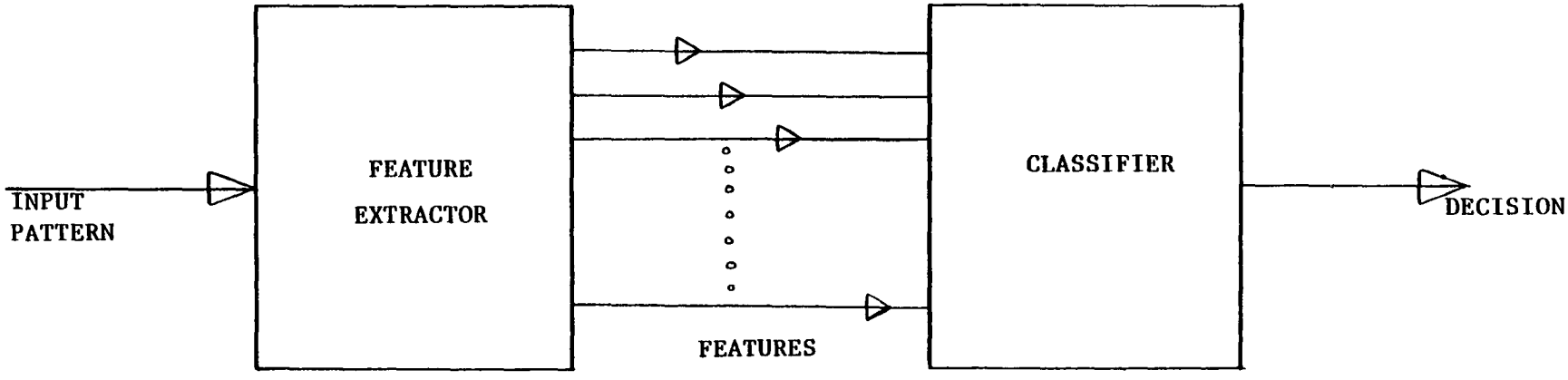


FIG. 4.3.1. A PATTERN RECOGNITION SYSTEM

selected measurements are called features and are supposed to be invariant or less sensitive with respect to commonly encountered variations and distortions. Under this approach, pattern recognition can be considered as consisting of 3 subproblems.

1. What to measure? That is, what primitive measurements should be represented in the input pattern?

2. What measurements (features) to extract from the input pattern and how?, i.e., Feature Extraction.

3. What pattern classification technique to use to make a class assignment to the input patterns based upon selected features?, i.e., Pattern classification algorithm.

Before operating a pattern classifier, one must first decide which measurements to use as the input pattern. Unfortunately, there is very little theory to guide in selection of measurements. At worst this selection process may be guided solely by the designer's intuitive ideas about which measurements play an important role in the classification at hand. At best the process can make use of known information about some measurements that are certain to be important.

For QA in image processing systems, such measurements may include, among various others, BER measurements, signal to noise ratio, statistical parameters such as mean, variance, etc. We will henceforth assume that a sufficient (large) number of measurements yielding the pattern to be

classified have been selected wisely remembering that the pattern classifier cannot itself compensate for a careless selection of measurements. Usually the decision of what to measure is rather subjective and higly depedent on practical considerations such as the availability of measurements, cost of measurements, etc. For instance, a certain measurement for QA may be known to contain extremely useful and important information. Yet, the cost of making that measurement may be prohibitive, thus making it impractical.

## Feature Extraction

As mentioned earlier, each of these measurements may carry a small amount of information about the sample or pattern to be classified. This high dimensionality makes many pattern recognition problems difficult. Obviously, as the number of measurements (or inputs) for the classifiers increases, the design of the classifier becomes more difficult. In order to simplify the problem, we should find some way to extract/select important features from the measured patterns. This problem is called 'feature extraction' and is a key problem in pattern recognition.

Feature selection is generally a process of mapping the original measurements into more effective features. If the mapping is linear, the mapping function is well defined and our task thus reduces to finding the coefficients of the linear functions so as to maximize or minimize a criterion function. Consequently, if we have the proper criterion for evaluating the effectiveness of features we can use well-developed techniques of linear algebra or apply optimizing techniques to determine these mapping coefficients, e.g., Principle Component Analysis.

Unfortunately, in many applications, there are important features which are not linear functions of original measurements. So, the basic problem translates into finding a proper non-linear mapping function for the given data. Since we don't have any general theory to generate such mapping funcions systematically and to find the optimum one, the selection of effective features becomes very much problem oriented.

#### 4.3.2.1 Pattern Classification Techniques

The concept of pattern classification may be expressed in terms of a mapping from feature space to the decision space. Figure 4.3.2 shows a generic articifical intelligence (AI) oriented system approach to quality assessment (QA) in an operational image processing system. It is presumed that the production processor is being monitored at N different stages and that at each of these stages one is able to make sufficiently large number of measurements which are reasonably useful for QA. Thus, $K_I$ measurements made at stage I can be represented by a vector P(I) representing the input pattern for stage I. The feature extractor for stage I would then yield a feature vector, p(I), containing $k_I$ ($<<K_I$) important features.

The problem of pattern classification can now be restated as follows: "Formulate a classification algorithm to assign each possible feature vector p(I) to proper pattern class, i.e., class 1 or class 2, where class 1 contains all input patterns of acceptable quality and class 2 contains input patterns that are not of acceptable quality. Mathematically, this problem can be formulated in terms of "discriminant functions", $D_i[p(I)]$, i=1,2.

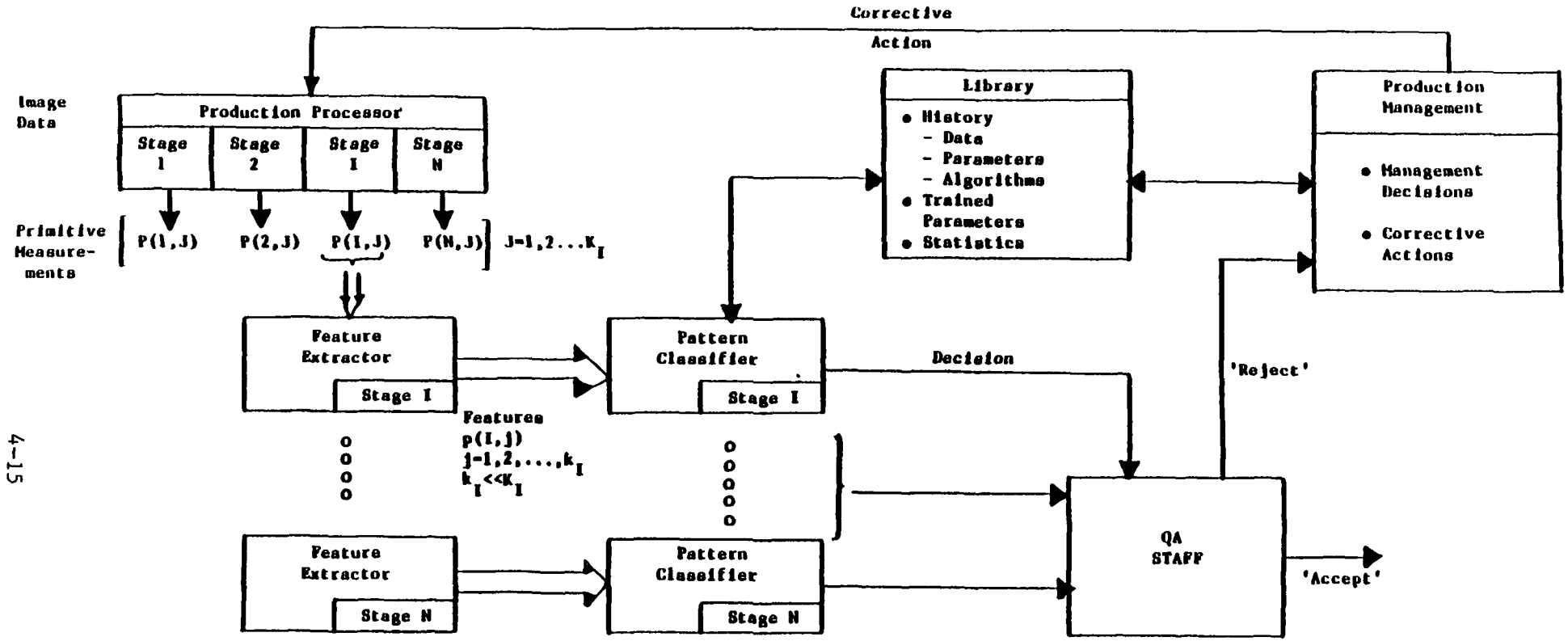


FIG. 4.3.2. AI APPROACH TO QA

The decision rule is given by:

if

$D_1\ [p(I)] > D_2\ [p(I)]$ p(I) belongs to class 1 (i.e., p(I) is of acceptable quality)

$D_1\ [p(I)] < D_2\ [p(I)]$ p(I) belongs to class 2 (i.e., p(I) is not of acceptable quality)

And, the decision boudnary (i.e., boundary of partition between class 1 and class 2 in the feature space) is expressed by the following equation,

$$D_1\ [p(I)] = D_2\ [p(I)]$$

or,

$$D_1\ [p(I)] - D_2\ [p(I)] = 0$$

A general block diagram for such a classifier is shown in Figure 4.3.3 while Figure 4.3.4 depicts a 2-dimensional illustration of the decision boundary. A wide variety of discriminant functions (e.g., linear, piecewise linear, minnimum-distance, quadratic, polynominal, etc.) are described in literature. For sake of simlicity, however, only linear discriminant functions will be discussed here. It should be pointed out that classifiers that use linear discriminant functions are called "linear classifiers".

A linear discriminant function is a linear combination of feature measurements, i.e.,

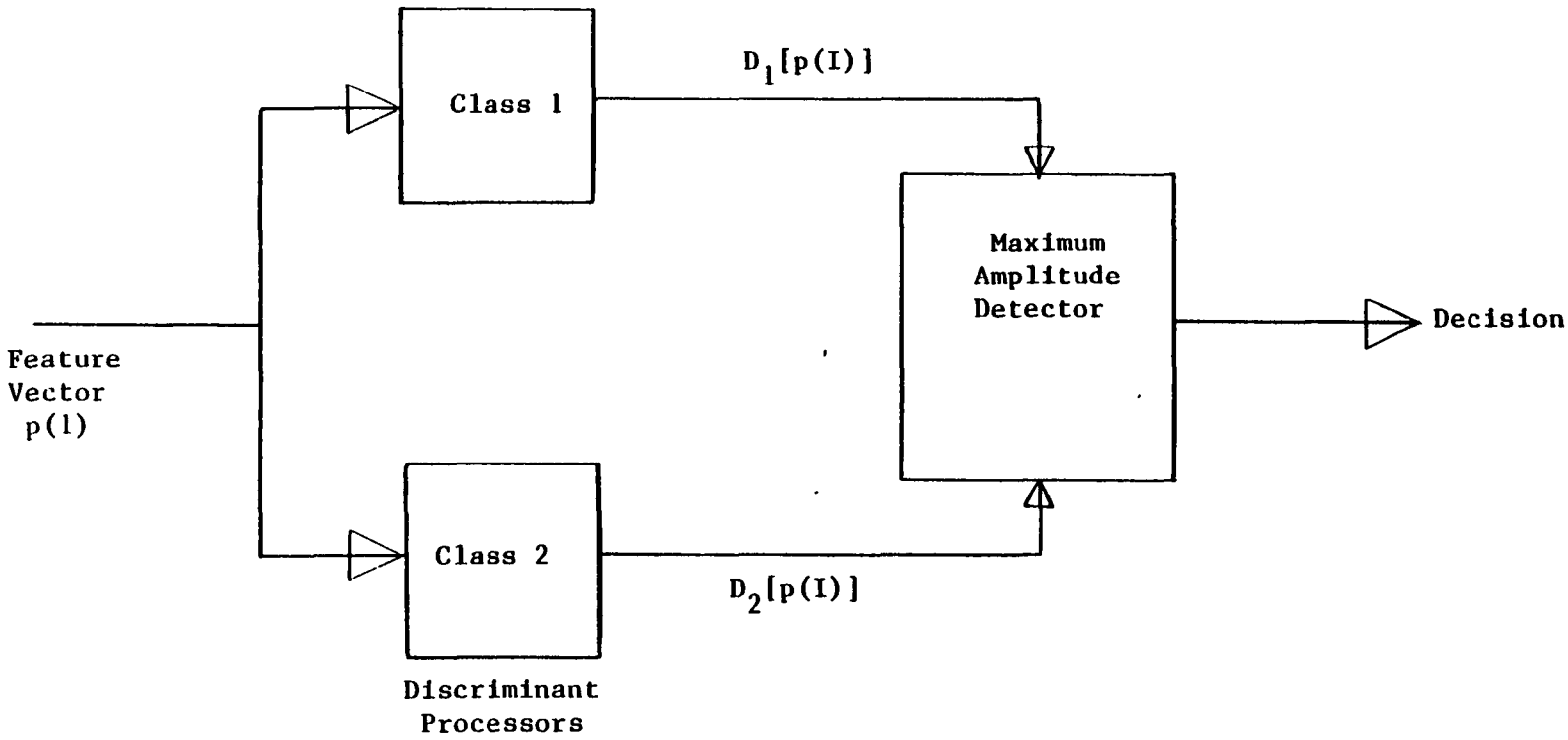


Figure 4.3.3

A PATTERN CLASSIFIER

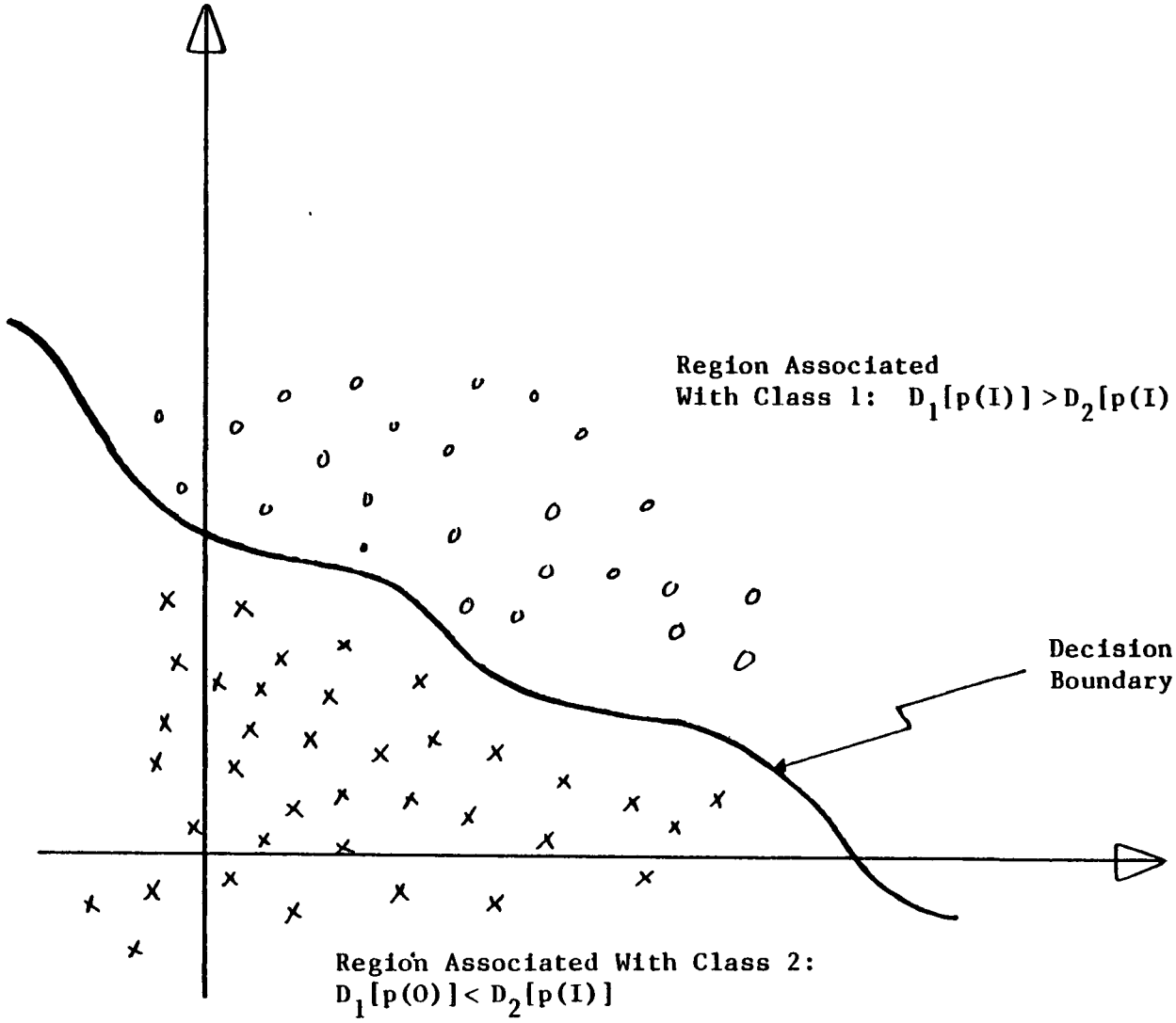


Figure 4.3.4

A DECISION BOUNDARY IN A 2-DIMENSIONAL FEATURE SPACE

$$D_i\ [p(I)] = W_i(I) \cdot p(I) + W_i(0) \quad , i=1, 2$$

$$= \overline{W}_i(I) \cdot \overline{p}(I)$$

where,

$\overline{W}_i(I)$ is the weight vector and $\overline{p}(I)$ is the augmented feature vector.

Let,

$$D[p(I)] = D_1[p(I)] - D_2[p(I)] = \overline{W}(I) \cdot \overline{p}(I)$$

The decision rule is given by:

if

$D[p(I)] > 0$, then $p(I)$ is acceptable

and if

$D[p(I)] < 0$, then $p(I)$ is not acceptable.

This decision rule, if necessary can be easily extended to multi-class situations so that feature vector $p(I)$ whould be assigned to the pattern class i with largest value of $D_i[p(I)]$.

4.3.2.2 Learning in Linear Classifiers

The implementation of the linear classification technique described above requires that proper values of the "weights" be available. However, in practice, the correct values of the weights are not known and, therefore, the classifier should be designed to have the capability of estimating the best values of weights from feature vectors. By observing the feature vectors with known classifications, the classifier should be able to automatically adjust the weights in order to acheive correct recognitions. And, the performance of the classifier should gradually improve as more and more patterns are observed. This process is called "training" or "learning" while the patterns used as inputs are called "training patterns"

For the sake of simplicity, it can be assumed that the augmented training patterns or feature vectors belonging to the two pattern classes are linearly separable (can be separated by a hyperplane in the feature space). This means that a weight vector W (I) exists such that

$\bar{W}(I) \cdot \bar{p}(I) > 0$ for each training pattern $\bar{p}(I)$ in class 1

or, $\bar{W}(I) \cdot \bar{p}(I) < 0$ for each training pattern $\bar{p}(I)$ in class 2

The "error-correction" training procedure can be summmarized as follows;

For any training pattern in class 1, the above product (i.e. $\bar{W}(I) \cdot \bar{p}(I)$ must be positive. If the output of the classifier is erroneous (i.e., product $<0$) or undefined (i.e., product $=0$), the weight vector should be adjusted to yield a new weight vector $\bar{W}'(I) = \bar{W}(I) + a.\bar{p}(I)$, where $a>0$ is called the correction increment.

On the other hand, for any training pattern in class 2, this product must be negative. Else, the weight vector should be adjusted to give $\bar{W}'(I) = \bar{W}(I) - a.\bar{p}(I)$.

Prior to training, the weight vector can be initialized to any convenient value. Some rules to make a proper selection of the correction increment (a) are given below.

(i) Fixed increment rule: a is any fixed positive number.

(ii) Absolute correction rule: a is chosen to be the smallest integer such that the product $\bar{W}(I) \cdot \bar{p}(I) > 0$.

(iii) Fractional correction rule:

$$a = b \cdot \frac{\bar{W}(I).\bar{p}(I)}{\bar{p}(I).\bar{p}(I)} \qquad 0<b\leq 2$$

Each of these correction rules is known to converge to yield a solution for weight vector in a finite number of training iterations.

4.3.2.3 Statistical Decision Techniques

In the preceding sections it was assumed that the feature measurements, p(I), are deterministic quantitite. However, in many applications such as image processing, this is not always true since noise effects in making these measurements cannot be neglected. This is because the input patterns in one class may have large variations.

One approach is to consider the feature vector, p(I), multi-variate random variable having known probability density function and known probability

of occurrences of each pattern class. Based upon this a priori information, the function of a pattern classifier is to perform the classification task for minimizing probability of misrecognition. The optimal decision rule which minimizes the average loss* is called the "Bayes Decision Rule" and a classifier that implements this rule is called a "Bayes Classifier."

Perhaps an example would help one understand the above and also reduce the mathematical complexities associated in formulating such decision rules. Assume that parameters such as "gain" (G) and "bias" (B) are found to be important in performing QA on radiometric correction process and a sufficiently large sample of these features is available for training.

Further, let

$f_1$ = probability of occurrence of Class 1

$f_2$ = probability of occurrence of Class 2

$F_1$ = Probability density function for all samples [gain, bias] belonging to Class 1

$F_2$ = Probability density function for all samples [gain, bias] belonging to Class 2

---

*Loss incurred by the classifier when it misrecognized. For the (0,1) loss function, the average loss is essentially same as the probability of misrecognition.

The Bayes Decision Rule will render the following as decision boundary between Classes 1 and 2.

$$f_1 \cdot F_1 - f_2 \cdot F_2 = 0$$

If both parameters, i.e., gain and bias, have gaussian density function within each pattern (which is realistic to assume), then the above decision boundary is a hyperquadric of the form $a.G^2 + b.B^2 + c.G.B+d = 0$. The coefficients of this equation are functions of mean and variance of gain and bias in each pattern class.

The decision rule for an incoming test pattern having gain G and bias B will be as follows;

if G and B are such that the pattern falls above the decision boundary, then it belongs to Class 1. Otherwise, the pattern belongs to Class 2.

Special Case: When covariance matrices of both pattern classes are equal and it is an unit matrix (or can be transformed into a unit matrix by performing a whitening transformation), the Bayes Classifier discussed above takes a much simpler form and becomes a distance classifier and the decision boundary is the perpendicular bisector of the line joining the mean values of gain and bias for the respective classes (see Figure 4.3.5).

It is believed that a classification technique similar to the one above might also prove to be useful for performing QA on parameters like bit-error-rate (BER) an signal-to=noise ration (SNR).

### 4.3.2.4 Sequential Decision Techniques

In the statistical classification system described in section 4.3.2.3 all the $k_I$ features are observed by the classifier at one stage. Additionally, the cost of making feature measurements was not taken into consideration. Usually an insufficient number of feature measurements would not result in satisfactory levels of correct classification. On the other hand, an arbitrarily large number of feature measurements is impractical. The problem

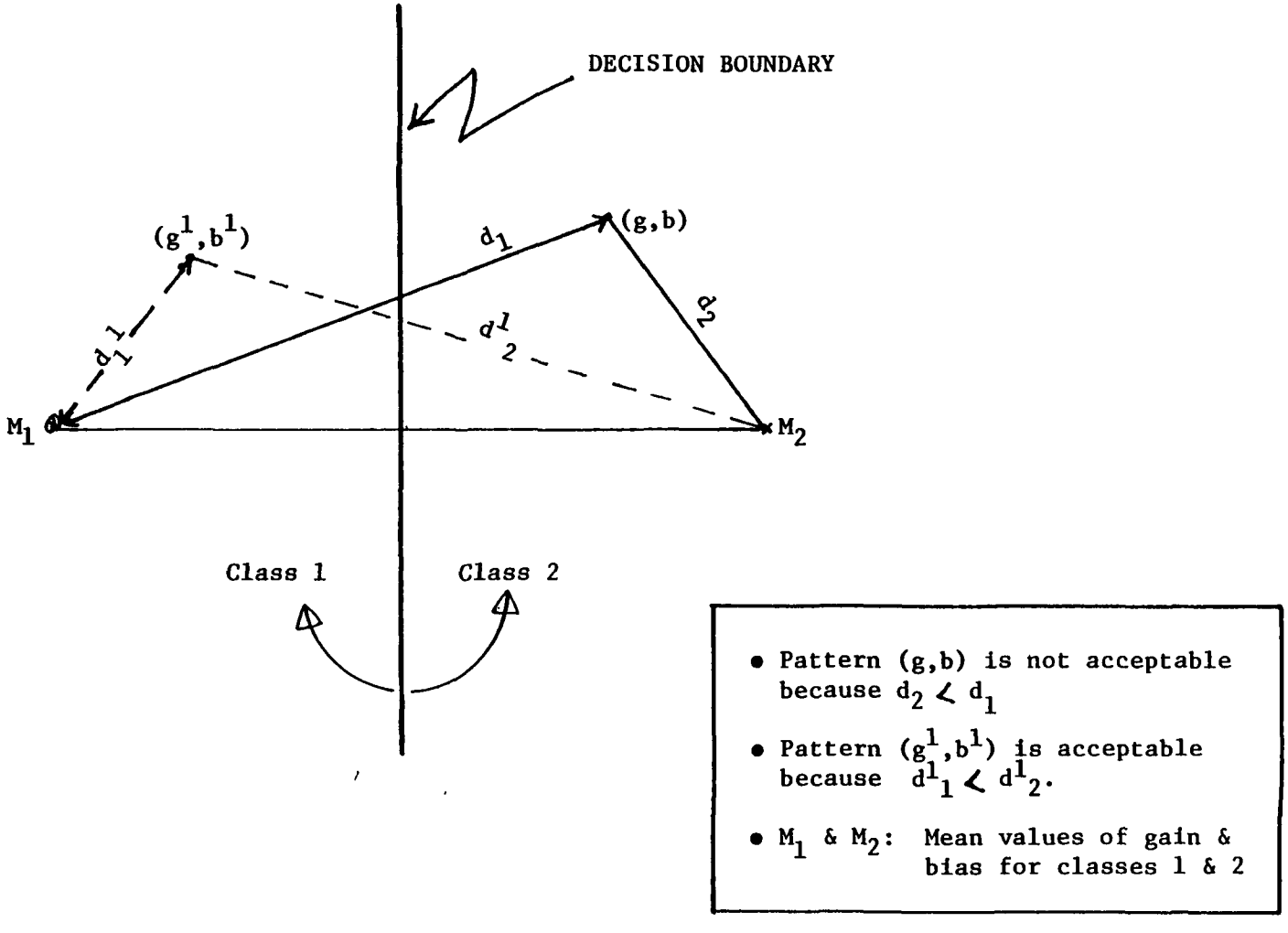


FIGURE 4.3.5 A DISTANCE CLASSIFIER

is especially pertinent when the cost of making a feature measurement is high. For example, if the measurement requires that the production process be interrupted or completely stopped, or if elaborate equipment, excessive times, or complicated operations are required to perform the measurement, then these factors may limit or even prohibit the use of such a feature. In such instances, sequential decision techniques provide a necessary balance between usefulness of a feature measurement and the cost of making that measurement. A trade-off between the error (misrecognition) and the number of features to be measured can be obtained by making feature measurement sequentially and terminating the sequential process (i.e., making a decision) when a sufficient/desirable accuracy of classification has been achieved).

Since the feature measurements are to be made sequentially, the order of features to be measured becomes important. The feature ordering scheme should be such that the measurements taken in that order will cause the terminal decision earlier. As a result, the problem of feature ordering is very important in sequential recognition systems.

Wald's sequential probability ratio test (SPRT) is one of the best sequential procedures known. At the $i^{th}$ stage of the sequential process, i.e., after the $i^{th}$ feature measurement is taken, the classifier computes the sequential probability ratio, R(i)

$$R(i) = \frac{F1(i)}{F_2(i)}$$

where $F_1$ and F2 are the probability density functions as defined in Section 4.3.2.3. This value of R is then compared with two stopping boundaries -- $S_1$ and $S_2$. The decision rule then becomes,

If $R \geq S_1$ then pattern p(I) belongs to Class 1, and if

$R \leq S_2$ then pattern p(I) belongs to Class 2.

On the other hand, if $S_2 < R < S_1$, then an additional feature measurement should be taken and the decision process proceeds to stage i+1. The stopping boundaries are related to the error (misrecognition) probabilities in the following manner;

$$S_1 = \frac{1 - e_{21}}{e_{12}}$$

and

$$S_2 = \frac{e_{21}}{1 - e_{12}}$$

where $e_{ij}$ is the probability of deciding that p(I) belongs to Class i when actually p(I) truly belongs to Class j [i, j = 1, 2]. It has been shown that Wald's SPRT is optimal, that is, for given values of $e_{12}$ and $e_{21}$ there is not other procedure with at least as low error probabilities or expected risk and with shorter length of average number of feature measurements.

It should be noted that the Wald's SPRT results in two decision boundaries which partition the feature space into three regions:

1. The region associated with Class 1
2. The region associated with Class 2
3. The region of indifference (null region)

The region between the two boundaries is the region of indifference in which no terminal decision is made. It is obvious, but important to note, that the decision boundaries in a sequential process vary with the number of feature measurements. For this reason, it is highly likely that such a process will be extremely useful while performing QA of the geometric

correction process based upon, for example, resampling parameters. For example, if $x_1, x_2, x_3 \ldots$ are independent measurements during resampling process, then assuming gaussion density functions (having means $m_1$ and $m_2$ and variance v for two classes), then sequential probability ratio R(i) can be computed numerically.*

After the first parameter $x_1$ is measured, R(1) is given by

$$R(1) = \frac{(m_1 - m_2)\, x_1 - 1/2\, (m_1^2 - m_2^2)}{v}$$

and, the decision boundaries are given by

if $x_1 \geq \frac{v}{m_1 - m_2}$ Log $S_1$ + 1/2 $(m_1+m_2)$, then pattern belongs to Class 1

if $x_1 \leq \frac{v}{m_1 - m_2}$ Log $S_2$ + 1/2 $(m_1+m_2)$, then pattern belongs to Class 2

and if, $\frac{v}{m_1 - m_2}$ Log $S_2$ + 1/2 $(m_1+m_2) < x_1 < \frac{v}{m_1 - m_2}$ Log $S_1$ + 1/2$(m_1+m_2)$

then next resampling parameter ($X_2$) is observed and the sequential decision process proceeds to Stage 2.

After measuring $x_2$, one can compute R(2) as follows,

$$R(2) = \frac{(m_1 - m_2)}{v} \quad [x_1 + x_2 - (m_1 + m_2)].$$

Proceeding as before, the decision boundaries are given by:

*For simplicity of computation, instead of R(i), Log (R(i)) is computed.

if $x_1+x_2 \geq \frac{v}{m_1-m_2}$ Log $S_1$ + $(m_1+m_2)$, then pattern belongs to Class 1

and if,

$$\frac{v}{m_1-m_2} \text{ Log } S_2+(m_1+m_2) < x_1+x_2 < \frac{v}{(m_1-m_2)} \text{ Log } S_1 + (m_1+m_2)$$

then next resampling parameter $x_3$ will be observed and the decision process will proceed to stage 3. This process may continue for several more stages. In general, the sequential classification procedure becomes such that

if $\sum_{i=1}^{n} x_i \geq \frac{v}{m_1-m_2}$ Log $S_1 + \frac{n}{2} (m_1 + m_2)$ then pattern belongs to Class 1

if $\sum_{i=1}^{n} x_i \leq \frac{v}{m_1+m_2}$ Log $S_2 + \frac{n}{2} (m_1 + m_2)$ then pattern belongs to Class 2

and if

$$\frac{v}{m_1\ m_2} \text{ Log } S_2 + \frac{n}{2} (m_1 + m_2) < \sum_{i=1}^{n} x_i < \frac{v}{m_1-m_2} \text{ Log } S_1 + \frac{n}{2} (m_1 + m_2), \text{ then}$$

the process continues to next stage (i + 1)

The width of the region of indifference is proportional to $\frac{v}{(m_1-m_2)}$ and, hence for given or assigned values of error probabilities $e_{12}$ and $e_{21}$, the average number of feature measurements for termination of this sequential process depends directly on variance v and inversely on $(m_1-m_2)$

It has been proven, in literature, that the Wald's SPRT

1. terminates with probability = 1

2. minimizes the average number of observations to achieve a given set of error probability values
3. is optimal

It should be noted that there exists a trade-off between the number of feature measurements that can be tolerated and selection of values for probabilities $e_{12}$ and $e_{21}$.

#### 4.3.2.5 Learning in Sequential Pattern Recognition Systems

In the previous section, all the information relevant to the statistical characteristics of patterns in each class is assumed to be completely known. However, in practical situations, this information is only partially known. One approach is to design a pattern recognition system which has the capability of learning the unknown information during its operation. The decisions (feature selections and classifications) are then made on the basis of learned information. If the learned information gradually approaches the true information, then the decisions based upon the learned information will eventually approach the optimal decisions as if all the information required were known. Therefore, during the system's operation, the performance and the knowledge of the system are gradually improved. The process which acquires necessary information for decision during system operation and which improves system performance is usually called "learning" or "adapting."

During the operation of a pattern recognition system, the system learns (estimates) the necessary information about each pattern class by actually observing various patterns. In other words, the unknown information is

obtained from these observed patterns. Depending upon whether the correct classifications of the input patterns observed are known or not, the learning process performed by the system can be classified into "learning with a teacher" or "supervised learning," and "learning without a teacher" or "nonsuperviced learning." In the case of supervised learning, Bayesian estimation and stochastic approximation can be used to successively estimate (learn) unknown parameters in a given form of feature distributions of each class. The successive estimation of continuous conditional probabilities of each pattern class can be performed by applying the potential function method or the stochastic approximation. The similarities between certain Bayesian estimation schemes and the generalized stochastic approximation algorithm have been demonstrated. It has also been shown that certain learning algorithms of the potential function method belong to the class of stochastic approximation algorithms. In nonsupervised learning (or clustering), the correct classifications of the observed patterns are not available and the problem of learning is often reduced to a process of successive estimation of some unknown parameters in either a mixture distribution of all possible pattern classes or of a known decision boundary.

One property of SPRT which can be used to improve the accuracy of classification is to reduce the error (misrecognition) probability by varying stopping boundaries. It has been shown that in SPRT if the upper stopping boundary $S_1$ is increased and the lower stopping boundary $S_2$ is decreased, then at least one of the error probabilities, $e_{12}$ and $e_{21}$, decreases.

#### 4.3.2.6 Summary

It is necessary to emphasize again that the selection of feature is an important problem in pattern recognition and it is closely related to the performance of classification. Furthermore, in sequential pattern recognition systems, the ordering of features for successive measurements is very important. The purpose of feature ordering is to provide, at successive stages of sequential classification process, a feature which is most "informative" among all possible choices of features for the next measurement so that the decision process can be terminated as early as possible.

## 5.0 USE OF QUALITY ASSESSMENT IN QUALITY ASSURANCE

### 5.1 Sampling as a Tool

In a well-designed system, it should be unnecessary and is probably overly expensive to test every piece of data at every stage in the process. Moreover, it is statistically certain that a number of errors will be in the data. QA must be concerned with the errors which are catastrophic to the data (such as loss of sync) and somewhat tolerant of simple data value errors (such as radiance errors). It should be clear that catastrophic errors are easier to detect and, if caused by some random phenomenon, can be eliminated by reprocessing. (It is not obvious how a radiance error could be detected after, say, a resampling process. Majority voting on three runs is an expensive possibility.)

In a system where a certain small number of detectable errors may be permitted but many errors cannot be permitted, sampling provides a means to estimate the number of errors without exhaustive testing. This may be the case wherein proper operation of a system produces only a few statistically generated errors (as from BER), but system failure produces many errors. As will be shown, sampling does not aid the case where a single error (or two or three) errors are intolerable.

Sampling can provide estimates of errors (bad data) in a population where trend analysis of threshold monitoring is being performed. In a population of N items with n errors, the probability that a sample of size k will contain x errors is given by the hypergeometric distribution

$$P(x) = \frac{\binom{n}{x}\binom{N-n}{k-x}}{N} = \frac{\binom{k}{x}\binom{N-k}{n-x}}{N}$$

$$P(x) = \frac{\binom{n}{x}\binom{N-n}{k-x}}{\binom{N}{k}} = \frac{\binom{k}{x}\binom{N-k}{n-x}}{\binom{N}{n}}$$

where

$$\binom{a}{b} = \frac{a!}{b!(a-b)!}$$

Given that a sample of size k is taken and that it does contain x errors, the maximum likelihood estimator of n, the number of errors in the total population, is

$$\hat{n} = \text{greatest integer not exeeding} \quad \frac{x\,(N-1)}{k}$$

This, simply, says the proportion of errors in the population is most likely the same as the proportion of errors in the sample. The variance in $\hat{n}$ is

$$\text{var}(n) = \frac{(N+1)^2}{k\,(N-1)}\,\frac{(N-k)}{N}\,\frac{n}{N}\,\left(1-\frac{n}{N}\right) \approx \frac{(N-k)}{k}\,n\,\left(1-\frac{n}{N}\right)$$

from which the "goodness" of the estimate can be known in a non-rigourous fashion (n is not known).

Consider the following argument for confidence estimation. A sample k contains x errors. It is most likely that the population N contains n = INT[x(N-1)/k] errors. The probability that the population contains more than some limit n' given that a sample of k contained x errors can be estimated as follows.

$$\text{Prob(acutal } n>n') = \frac{\text{Ways that x errors in k can come from all } n > n'}{\text{ways that x errors in k can come from any n}}$$

$$= \frac{\sum_{i = n'+1}^{N-(k-x)} \frac{\binom{i}{x}\binom{N-i}{k-x}}{\binom{N}{k}}}{\sum_{i = x}^{N-(k-x)} \frac{\binom{i}{x}\binom{N-i}{k-x}}{\binom{N}{k}}}$$

$$= \frac{\sum_{i = n'+1}^{N-(k-x)} \binom{i}{x}\binom{N-i}{k-x}}{\sum_{i = x}^{N-(k-x)} \binom{i}{x}\binom{N-i}{k-x}}$$

$$= \left[ 1 + \frac{\sum_{i = x}^{n'} \binom{i}{x}\binom{N-1}{k-x}}{\sum_{i=n'+1}^{N-(k-x)} \binom{i}{x}\binom{N-i}{k-x}} \right]^{-1}$$

The only unspecified parameter in the above (given the results of a sampling) is n', so the probability -- the confidence in this case -- can be computed as a function of an upper limit on the number of errors in the population.

In the special case where no errors are observed in the sample, the equation reduces to

$$P_r(n > o) = \left[ 1 + \frac{\binom{N}{K}}{\sum_{i=1}^{N-k} \binom{N-i}{k}} \right]^{-1}$$

For reasonable values of k, this is approximately 1 - k/N as intuition tells us.

Table 1 gives the probability of obtaining x errors in a sample of size k for a population of 1000 for various numbers of errors in the population. The column for one error in the population (n=1) is intuitive. If there is one error, the probability of observing it equals the fraction of the population sampled.

Table 2 gives the probabilities of n errors existing in the population when 0, 1, or 2 errors are observed in samples. The population is 1000 and sample sizes of 200, 500, and 900 are shown. Suppose 1/2% errors (5 in 1000) could be tolerated. Then if zero errors were observed in a sample of size 500, the probability is 0.9847 that the number of errors in the population is 5 or fewer (by summing from n=0 to n=5). If one error was observed, the confidence would be 0.8917, and if two errors were observed, the confidence would drop to 0.6577.

| x, ERRORS IN SAMPLE | n, ERRORS IN POPULATION | | | | | | | | k, SAMPLE SIZE |
|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
| 0 | 1 | 0.9 | 0.8099 | 0.7288 | 0.6557 | 0.5898 | 0.5306 | 0.4772 | 100 |
| | 1 | 0.8 | 0.6398 | 0.5116 | 0.4090 | 0.3269 | 0.2612 | 0.2086 | 200 |
| | 1 | 0.5 | 0.2497 | 0.1246 | 0.0621 | 0.0309 | 0.0154 | 0.0076 | 500 |
| | 1 | 0.1 | 0.0099 | + | + | + | + | + | 900 |
| 1 | 0 | 0.1 | 0.1802 | 0.2435 | 0.2924 | 0.3291 | 0.3557 | 0.3736 | 100 |
| | 0 | 0.2 | 0.3203 | 0.0385 | 0.4105 | 0.4106 | 0.3942 | 0.3678 | 200 |
| | 0 | 0.5 | 0.5005 | 0.3754 | 0.2500 | 0.1559 | 0.0933 | 0.0542 | 500 |
| | 0 | 0.9 | 0.1802 | 0.0268 | 0.0035 | + | + | + | 900 |
| 2 | 0 | 0 | 0.0099 | 0.0268 | 0.4840 | 0.0727 | 0.0982 | 0.1240 | 100 |
| | 0 | 0 | 0.0398 | 0.0958 | 0.1536 | 0.2051 | 0.2464 | 0.2762 | 200 |
| | 0 | 0 | 0.2497 | 0.3754 | 0.3758 | 0.3131 | 0.2346 | 0.1639 | 500 |
| | 0 | 0 | 0.8099 | 0.0235 | 0.0484 | 0.0079 | 0.0012 | + | 900 |
| 3 | 0 | 0 | 0 | + | 0.0030 | 0.0079 | 0.0143 | 0.0226 | 100 |
| | 0 | 0 | 0 | 0.0079 | 0.0254 | 0.0509 | 0.0816 | 0.1145 | 200 |
| | 0 | 0 | 0 | 0.1246 | 0.2500 | 0.3131 | 0.3134 | 0.2743 | 500 |
| | 0 | 0 | 0 | 0.7288 | 0.2924 | 0.0727 | 0.0143 | 0.0024 | 900 |

TABLE 1 PROBABILITY OF X ERRORS IN SAMPLE GIVEN N ERRORS IN POPULATION

Population, N, = 1000
Sample Size = 100, 200, 500, 900

| ERROR IN POPULATION | OBSERVED ERROR IN SAMPLE = 0 | | | 1 | | | 2 | | |
|---|---|---|---|---|---|---|---|---|---|
| | SAMPLE SIZE | | | | | | | | |
| | 200 | 500 | 900 | 200 | 500 | 900 | 200 | 500 | 900 |
| 0 | .2010 | .5005 | .9001 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | .1608 | .2502 | .0900 | .0405 | .2502 | .8101 | 0 | 0 | 0 |
| 2 | .1286 | .1250 | .0089 | .0648 | .2505 | .1622 | .0083 | .1250 | .7290 |
| 3 | .1028 | .0624 | .0009 | .0778 | .1879 | .0241 | .0199 | .1879 | .2191 |
| 4 | .0822 | .0311 | .00008 | .0831 | .1251 | .0031 | .0319 | .1881 | .0435 |
| 5 | .0657 | .0155 | 0 | .0831 | .0780 | .0004 | .0426 | .1567 | .0071 |
| 6 | .0525 | .0077 | 0 | .0798 | .0467 | .00004 | .0512 | .1174 | .0010 |
| 7 | .0419 | .0038 | 0 | .0744 | .0271 | 0 | .0575 | .0820 | .0001 |
| 8 | .0335 | .0019 | 0 | .0680 | .0154 | 0 | .0613 | .0545 | .00001 |
| 9 | .0267 | .0009 | 0 | .0612 | .0086 | 0 | .0631 | .0349 | 0 |
| 10 | .0213 | .0005 | 0 | .0543 | .0048 | 0 | .0631 | .0217 | 0 |

TABLE 2 PROBABILITY OF N ERRORS IN POPULATION GIVEN X ERRORS IN SAMPLE (X=0,1,2)

Population, N, = 1000
Sample Size = 200, 500, 900

### 5.1.1 Adaptive Sampling

In an automated QA system operating at, say, 90% confidence, a sample in the above example showing one error would be below the confidence threshold. However, 89% confidence is still good. Resampling (or sampling the next trial of 1000) at a higher sampling rate is recommended to tighten the variance on the estimate. Suppose that 90% sampling were performed. Then, two observed errors would yield 99.87% confidence that five or fewer errors were in the population. If more errors were observed (say 4 or 5), then 100% sampling and a quality warning would be indicated. It there is high confidence that quality is being maintained, the sampling rate would drop back to smaller levels.

Such a sampling scheme is recommended whenever a small number of errors can be tolerated and measuring errors is time consuming or expensive.

## 5.2 Other Parameters

### 5.2.1 Costs

The breakpoint for a QA system is where costs due to having products (final and intermediate) passed on which do not meet specifications balance the cost of the QA system. System engineering would determine balancing QA costs and processing system improvements, considering throughput loading due to the need to reprocess data. Costs due to unsatisfied errors are difficult to know, but result from having no product, a product not meeting

specification, and late products. This trade-off is presented in Figure 5-1.

Costs can be reduced if the QA system gives not only indication of quality failures, but provides a measure of urgency of required remedial action. A total failure requires immediate action. A quality failure due, probably, to random failure (a statistical excess of errors in one product) requires no action but reprocessing (and a logged report of reprocessing -- an indication of system status itself). A trend analysis crossing a warning threshold requires, perhaps, preventive maintenance or a test sequence to be scheduled at the end of an operational shift.

### 5.2.2 Other Measures

There are benefits accruing to a QA system which reduce costs to other system components or provide capabilities beyond QA.

A proper QA system with its MIS contains data for use in analyzing and recommending changes in preventive maintenance scheduling, spares inventory policies, operator training and, ultimately, processing system design changes.

In reviewing the costs for an automated QA system, these benefits should receive accounting.

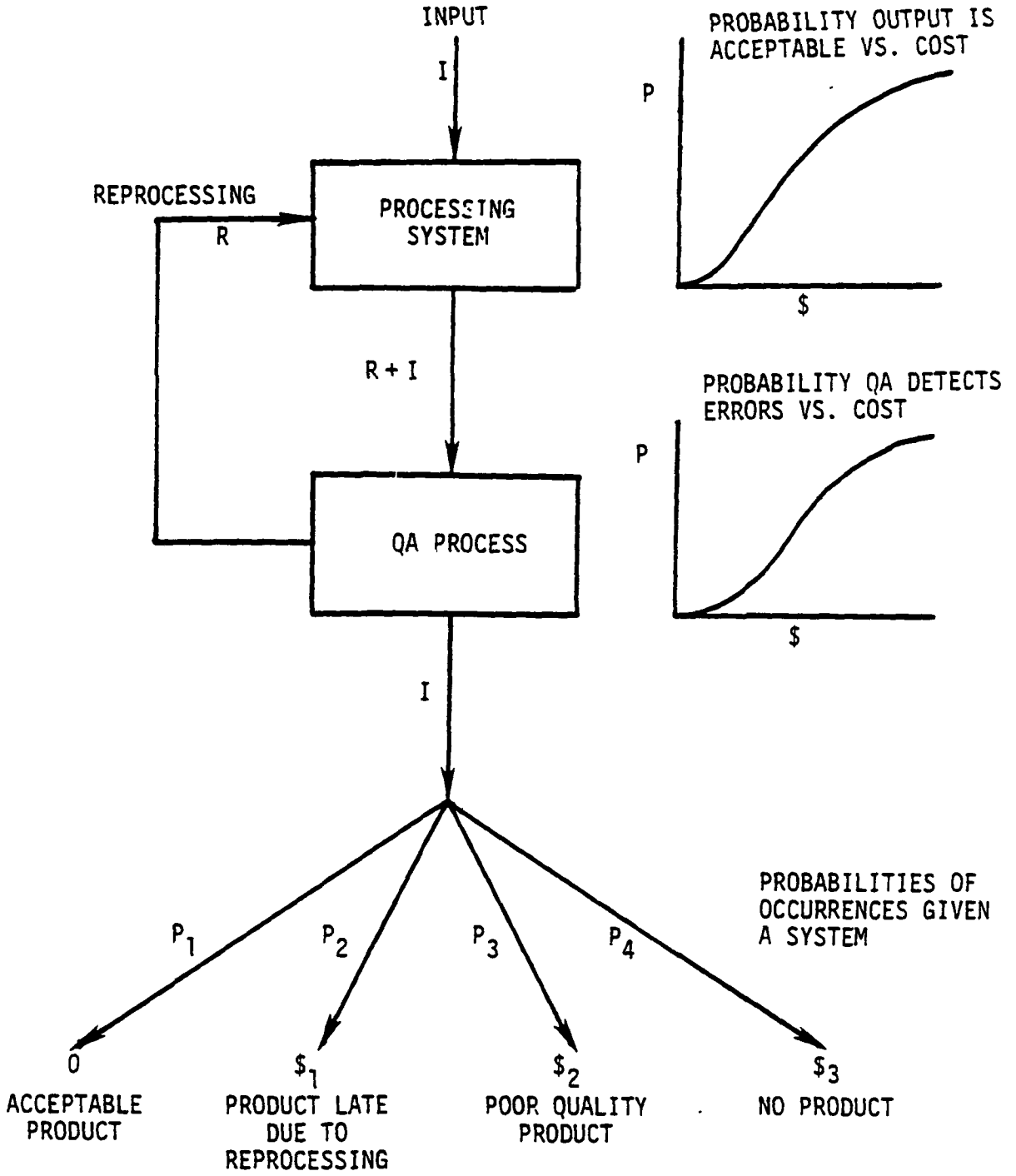


FIGURE 5-1. Cost Trade-Off Model

## 6.0 SURVEY OF STATE-OF-ART

### 6.1 Literature Survey

A fairly comprehensive search of existing literature was conducted in an effort to identify available techniques and existing systems using such techniques for purposes of performing QA (automated and/or semi-automated) of image data during ground processing. The search included, among others, numerous on-line queries on National Technical Information Services's (NTIS) databases, many trips to local scientific/technical libraries, and a thorough screening of various IEEE publications during past 4 or 5 years. Unfortunately, the results have not been encouraging. In fact, no technique(s) or system(s) could be identified to assist us in simplifying the QA problem at hand.

A brief description of each of the relevant articles uncovered during literature search is presented below.

Antikidis [1] has attempted to show how important the needs for image quality are in the definition of an image-taking satellite system and the associated on-board and ground processing facilities. Some measures of image quality have been defined in the framework of future European Space Agency (ESA) sensing system.

Leberel and Kropatsch [2] have conducted experiments with part of a digital Landsat-image of Southern Germany to show that automatic location of features in a digital image is feasible if recognition is supported by

a digital map database. The authors have recognized 13 features in the test scene and reported that resulting image rectification left residual point errors of less than $\pm$ 1 pixel.

Tsuchiya and Arai [3] have suggested an approach to geometric correction processing. Removal of geometric errors in Landsat MSS imageries in precision processing is made using GCP's (Ground Control Point). Thus selection of GCP's affects the geometric accuracy of the processed imageries. Based on 2 years Landsat MSS imageries data, effects of the feature of GCP matching success rate and cross correlation of the two imageries which should be registered were studied together with the relationship between time lapse of two imageries and success rate of GCP matching. It was found that the best GCP's in the automatic matching are island, wharf and break waters, and the best GCP's in the manual matching are break water, highway intersection and wharf. Furthermore, it was also found that break water and wharf indicate high cross correlation coefficient in the automatic GCP matching. There was a periodical tendency in the success rate of GCP matching with the prevailing period of 21 months. Between two imageries of time lapse ranging from 8 to 17 months, a symmetric tendency was found in GCP matching success rate with the maximum of 12 months.

Williams, Siebert, and Gunn [4] have described an image analysis system known as KARS. The Kansas Applied Remote Sensing (KARS) program and Department of Geography-Meteorology have developed an interactive digital image processing program package that runs on the University of Kansas central computer. The module form and simple Fortran programming of the package has allowed easy and rapid upgrades and extensions of its capabilities.

The package is comprised of subimage extraction and rectification, image display and enhancement, and both supervised and unsupervised classification routines. It has been used in both instructional and research settings at the University.

A classification of multi-sensor imagery from the sensor's point of view is advanced by Casasent and Munoz [5]. From this treatment, the statistical and deterministic contributions to a multi-sensor image correlation process are more clearly seen. The optimum preprocessing operation for several cases of multi-sensor image pattern recognition are noted and the use of weighted matched spatial filter synthesis as a one step optical pattern recognition correlator is described. Theoretical formulation and experimental verification of the result that edge enhancement preprocessing is not always optimum in a multi-sensor optical image pattern recognition system are presented.

Aggarwal and Panda [6] have described a system developed by Honeywell for analyzing the imagery automatically and detecting tactical as well as strategic targets in the image. The main features of the image recognition system are sequential frame processing, symbolic image segmentation, context-dependent syntactic recognition, and recognition of multi-component objects and conflict removal.

# 7.0 CONCLUSIONS

## 7.1 Recommendations

The primary recommendation from this study is that Quality Assurance be considered in the system design from the earliest point, and that access to the data be provided in the design for QA. This is conceptually easier to do for modular, serial-processing systems than for highly integrated parallel-processing design. In the latter, QA should be addressed on whichever level provides access to data and to whichever level fault isolation is desired. This may be a fairly low subfunction level.

Another important recommendation is that, whatever the level of automation, some supervision of the QA process by an analyst is required. Known quality measures can be programmed from the start as a "knowledge based system (a structured set of IF-THEN statements), and, with access to the data, additional quality measures can be added as they are discovered by analysts. It is not cost effective to insure against every conceivable failure; many failures in existing systems were certainly not foreseen and would have been assigned an extremely low probability a priori had they been considered.

Should NASA wish to pursue even more automation of QA in future systems, an adaptive "learning" process is recommended. Again, with access to the data, simple statistics and trend analyses can be calculated inexpensively. Analysis of the trends and development of a classification algorithm for QA may prove worthwhile.

Specific recommendations are given individually in the following.

- QA must be a system level function, composed of central QA and local QA functions which may be distributed throughout the system.

- Quality should be measured/monitored at the level of satellite design, checkout, in-flight control as well as on-board and ground processing. That is, QA must become an important element of end-to-end satellite system design since the question of image quality is no longer just an instrumental concept.

System Design Impacts

- The ground system by design must be required to provide access to all data by the central QA function or process. Such access may be provided via numerous taps into the production processing system.

- QA must have strong interface with Production Control.

- Cost-effective studies shall account for overall QA process.

QA Process

- Central QA should control local QA functions, local QA functions determine and select data to be analyzed as data progresses through various stages of production process.

- Contains or has access to a MIS to track system history and status of repair and maintenance.

- Has access to quality of input data.

QA Algorithms

- Some algorithms may be shared by local functions:
  - Statistics
  - Trend Analyses
  - Sampling Algorithms

- Known measures be assessed by specific calculations ("AI" type IF-THEN calculations).

- Adaptive algorithms may be included for unforeseen problems or growth in analysis.

- QA should be structured so new known algorithms can be added easily.

- The Central QA function must provide quality indicating measures which may later be appended to all output products before their dissemination to the user community.

- All QA algorithms/parameters must be stored for TBD years (perhaps, life of mission) for quick retrieval to aid in future analyses.

- Simple QA functions such as checking data bounds, calibration, etc. can and should be made adaptive at a reasonable increase in system costs initially. Yet, in the long run, this should result in cost savings. Such QA parameters can be computed in real-time or near real-time.

- More complex QA functions such as those needed to perform QA of the geometric corrections processing may also be made adaptive. However, implementation of corresponding QA algorithms will probably not be in real-time or near real-time. Additionally, cost of their implementation would, in all likelihood, far exceed the resulting benefit.

- Certain types of QA functions (such as detecting a "zipper") will best be performed by a human analyst since no simple/known algorithms exist to even detect such deficiencies by means of computations.

## 8.0 BIBLIOGRAPHY


1. Antikidis, J. P., "Introduction to Image Quality Definition and Requirements for Remote Sensing Satellites." 14th Congress of the International Society of Photogrammetry, Hamburg, West Germany, July 1980, p. 416-425.

2. Leberl, F. and Kropatsch, W., "Experiments With Automatic Feature Analysis Using Maps and Images," 14th Congress Of the International Society of Photogrammetry, July 1980, p. 451-457.

3. Tsuchiya, K. and Arai, B., "Some Effects On the GCP Success Rate," 7th Canadian Symposium on Remote Sensing, Winnipeg, Canada, Sept. 1981, p. 497-502.

4. Williams, T. H. Lee, Siebert, J., and Gunn, C., "The KARS Image Analysis System: A Low Cost Interactive System For Instruction and Research," *Machine Processing of Remotely Sensed Data Symposium*, West Lafayette, Indiana, June 1981, p. 178-180.

5. Casasent, D. and Munoz, D., "Statistical and Deterministic Aspects of Mulit-sensor Optical Image Pattern Recognition," Society of Photo-Optical Instrumentation Engineers, Vol. 201, 1979, p. 58-64.

6. Aggarwal, R. K. and Panda, D. P., "Context Dependent Automatic Image Screening System," Society of Photo-Optical Instrumentation Engineers, Vol. 205, 2980, 85-89.