NASA CONTRACTOR REPORT

NASA CR-999

NASA CR-999

RESEARCH ON THE UTILIZATION OF PATTERN RECOGNITION TECHNIQUES TO IDENTIFY AND CLASSIFY OBJECTS IN VIDEO DATA

Prepared by
DOUGLAS AIRCRAFT COMPANY
Newport Beach, Calif.
for Electronics Research Center

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • MARCH 1968

NASA CR-999

RESEARCH ON THE UTILIZATION OF PATTERN RECOGNITION TECHNIQUES TO IDENTIFY AND CLASSIFY OBJECTS IN VIDEO DATA

Issued by Originator as Report No. SM-48464-F

Prepared under Contract No. NAS 12-30 by
DOUGLAS AIRCRAFT COMPANY
Newport Beach, Calif.

for Electronics Research Center

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

For sale by the Clearinghouse for Federal Scientific and Technical Information
Springfield, Virginia 22151 – CFSTI price $3.00

FOREWORD

This report was prepared by Astropower Laboratory, Douglas Aircraft Company, Newport Beach, California, in fulfillment of NASA Contract NAS 12-30. The work was sponsored by the NASA Electronics Research Center, Cambridge, Massachusetts. Mr. Eugene M. Darling, Jr. was the Technical Officer for this office.

The studies began on June 18, 1965. This final report is the sixth technical report, and encompasses the material contained in the earlier documents.

Work on the contract was performed by the Electronics Department of Astropower Laboratory. The major contributors were R. D. Joseph (Project Leader), C. C. Kesler, W. B. Martin, J. N. Medick, A. G. Ostensoe, R. G. Runge, and S. S. Viglione (Project Manager).

ABSTRACT

This report covers the research accomplished in a two-year period on NASA Contract NAS 12-30. These studies are concerned with the feasibility designing of a spaceborne pattern recognition system.

The study was conducted in three parts: (1) A literature survey to select a number of pattern recognition techniques for testing; (2) A series of controlled recognition experiments to test the performance of the selected techniques on five specimen classification tasks; (3) A hardware feasibility study culminating in three potential designs for a spaceborne recognition system. These designs were formulated in sufficient detail to define such parameters as size, weight, power, processing time and storage for each of the three system configurations.

Six techniques for generating property filters and eight algorithms for designing decision functions were tested. Two data samples were used in these experiments: (1) photographs of lunar terrain; (2) NIMBUS satellite pictures of cloud patterns. Decision networks were designed to perform three recognition tasks on the lunar data and two tasks on the NIMBUS data. Performance was measured in terms of percent correction decisions on an independent sample of patterns. The best results obtained on these five recognition tasks ranged from 84.5 to 99.5% correct.

Three possible mechanizations of a spaceborne recognition system were designed: (1) parallel/analog, (2) sequential/hybrid, (3) sequential/digital. Common to each is an input device consisting of an image dissector tube which views the scene and provides scanning of the sensed image through a small aperture. The parallel/analog system implements all property filters separately in analog form, resulting in a heavy system (50 lbs) which consumes considerable power (180 w). The sequential/hybrid system consists of a single property filter, used repetitively, with a diode matrix containing the weighting elements for each computed property. The resulting system weighs 32 pounds and comsumes 37 watts. The sequential/digital system uses a compact core storage unit and an arithmetic unit with a combination of hardwired program and software program to implement the required

property filters. Not only does this system have the smallest weight (24 lbs) and least power consumption (35 watts) of the three, but it also affords the added advantage of flexibility in logic mechanization.

The salient operating characteristic of all three systems is the capability of classifying TV pictures in real-time (i.e., in the time interval between successive pictures).

SUMMARY

This investigation was conducted in three parts – (1) a literature survey, (2) a software feasibility study, and (3) a hardware feasibility study. The literature search, discussed in Section 2.0, was conducted to seek promising pattern recognition techniques for investigation in the software feasibility study. The software feasibility study (Section 3.0) comprised an experimental program to compare and evaluate the selected techniques on sample recognition tasks. The hardware feasibility study (Section 4.0) entailed the preliminary design of three system mechanizations of a spaceborne recognition processor. For each design an assessment of such important parameters as weight, power, processing time, and storage requirements was made.

The number of pattern recognition techniques encountered in the literature search was almost as great as the number of authors. Furthermore, the authors' penchant for antonomasia creates much confusion. However, with the aid of Figure 1 it is possible to classify most of the techniques reported. In that figure, signal conditioning is nonabstractive processing which formats input data, and transforms it to achieve invariance to irrelevant pattern differences (such as rotations, translations, etc.). Known property extraction refers to the detection of pattern features which the designer knows, or suspects, to be of value in the decision process. Statistical property extraction refers to the detection of important features by statistical analysis of sample patterns. The decision mechanism is the means by which a set of pattern features is converted to a decision.

All systems reported in the literature search bypass at least one of these recognition system components. Indeed, the work described in Section 3.3.7 of this report is the first we have encountered in which both known properties and statistical properties are used. Most research investigations concentrated on the study of one algorithm for designing a decision mechanism; most applied studies centered on the design of known properties.

Based on this literature survey, six state-of-the-art algorithms for the design of decision mechanisms were selected. These were –

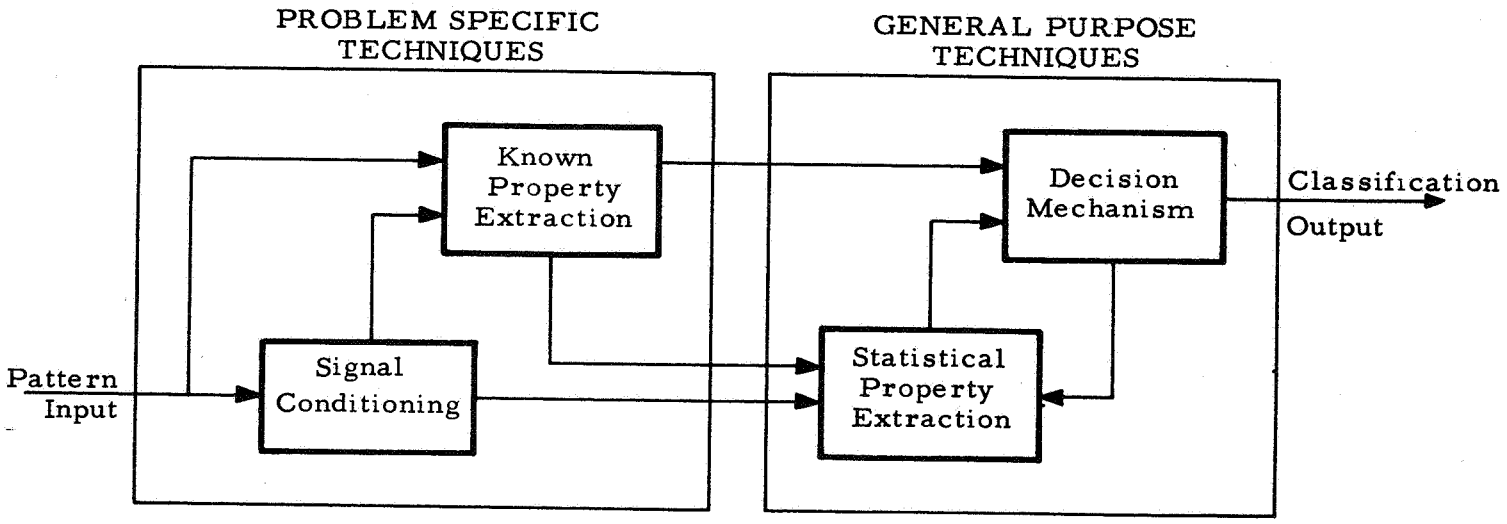


Figure 1. Pattern Recognition Mechanism

(1) Forced Learning
(2) Bayes Weights
(3) Error Correction
(4) Iterative Design
(5) Mean Square Error
(6) MADALINE

MADALINE produces a piecewise linear decision surface in the property space; the rest of the techniques are linear. These algorithms are described in Section 3.3.1. Subsequently, two additional nonlinear techniques

(7) Piecewise Linear
(8) Distribution Estimation

were added, as well as several modified error correction and MADALINE schemes (Section 3.3.5).

Two methods for statistically designing property filters were chosen (Section 3.3.1). Both use random selection of subspaces for the property filters, and in both cases, the property filter output is binary. In one method, the first moments of the sample patterns are analyzed to obtain linear switching surfaces for the property filters. In the second method, the first two sample moments are analyzed to derive quadratic switching surfaces. The second method also involves a selection process.

The data samples (Section 3.2) used in these experiments depicted the following four types of lunar terrain and three kinds of cloud patterns:

Lunar Terrain

- Craters with no conspicuous central elevations
- Craters with one or more conspicuous central elevations
- Wrinkle ridges
- Rima (i.e., rilles)

Clouds

- Noncumulus
- Cumulus, solid cells
- Cumulus, polygonal cells

Five binary recognition tasks were performed:

Lunar Terrain

(1) Task CVC – separate the craters with central elevations from those without

(2) Task RVR – separate the wrinkle ridges from the rima

(3) Task CVR – separate the composite class of craters from the composite class of ridges and rima

Clouds

(4) Task NVC – separate the noncumulus clouds from the composite class of cumulus clouds

(5) Task PVS – separate the cumulus solid cells from the cumulus polygonal cells

The lunar patterns were taken from the USAF Lunar Atlas;[155] the cloud patterns from NIMBUS TV pictures. The patterns were rephotographed using a slow-scan TV, recorded on magnetic tape, and digitized. The resolution was then reduced to one-fourth of that in the original digitized pictures, and the gray scale was compressed to three bits. The resolution and raster sizes were selected on the basis of pilot resolution studies. Subframes of 50 x 50 elements were found to be sufficient for the lunar data, while 75 by 75 element subframes were found best for the NIMBUS data. By translating and rotating patterns within subframes the number of sample patterns available was increased. In this way two pattern files were established: a file of 1000 sample patterns used to design the recognition systems; an independent file of 200 sample patterns used to test the systems.

All twelve combinations of six decision function algorithms and two property filter design methods were applied to the three lunar recognition tasks. Performance with the quadratic property filters proved to be much better than with the linear property filters. Consequently, experimentation with linear property filters was discontinued. Using quadratic property filters only, six decision techniques were tested on both cloud pattern tasks. The highest generalization performances obtained on the five recognition tasks were:

Task	Performance %
Lunar Terrain	
CVC	63.75
RVR	75.75
CVR	99.50
Clouds	
NVC	86.25
PVS	85.50

The significant feature in lunar task CVC is the central elevation of the crater. This feature is not always well defined and, at best, occupies only one percent of the sampling aperture. On lunar task RVR the ordering of the bright and dark parts of the wrinkle ridge or rima, the significant feature, is not always well defined. In this case the significant feature occupies five percent or less of the aperture. On lunar task CVR, the crater itself forms a significant feature. It is almost always well defined, and occupies perhaps 30% of the aperture. (A test on the CVR recognition system, however, indicated that features of both classes were being utilized in the discrimination.) It would therefore seem that the signal-to-noise ratio has a critical effect on the performance levels achieved.

In order to eliminate noise which might corrupt the design of a decision network, artificial noise-free patterns were generated for lunar task RVR. These idealized patterns were used to design the system. Testing was conducted on the real patterns. Performances obtained were much poorer than in the original experiments. The most likely cause for this result was the difficulty in generating a representative set of artificial patterns.

In a second attempt to improve the signal-to-noise ratio, a 25 by 25 and 15 by 15 element subaperture was generated from each 50 by 50 element pattern used in lunar tasks CVC and RVR. Examples of these reduced aperture patterns are found in Section 3.3.4.4. The best generalization performances achieved with decision networks designed on these reduced aperture data were the following:

Lunar Task	Performance, %	
	25 by 25 Aperture	15 by 15 Aperture
RVR	77.00	84.50
CVC	83.25	96.25

Reducing the aperture size improves the signal to noise linearly on Task RVR (as part of the pattern is eliminated along with the noise), and quadratically on Task CVC (as none of the significant features is lost). Subsequent experiments with reduced aperture sizes on Task CVR lead to poorer performance than with the original aperture. This result can probably be attributed to the fact that with the smaller apertures substantial portions of the craters are lost.

Thus, with suitable choices of aperture size, the three lunar recognition tasks were performed with accuracies of 84.5, 96.2, and 99.5%. Reduced aperture experiments were not conducted for the two cloud recognition tasks because the results were very good with the 75 x 75 element aperture. With respect to the relative merits of the six decision function design techniques the following conclusions were reached: (1) The Bayes weights, and mean square error algorithms showed up well in the comparisons only in those cases where the performance of all systems was unacceptably low. They were not competitive in the other cases. (2) Error correction and iterative design were superior to the other techniques on lunar task CVR; MADALINE was best on task RVR (15 by 15 aperture). (3) On all other tasks, there was little to choose among these three techniques.

One modification in the error correction algorithm, two modifications to the MADALINE technique, and an additional nonlinear decision mechanism, "Piecewise Linear," were tested on three tasks. Performances achieved showed remarkably little variation, the best being 88 percent on cloud task NVC, 86 percent on cloud task PVS, and 75 percent on lunar task RVR (50 x 50). Another decision function technique, based on nonparametric distribution estimation, leads to very complex decision systems. However, this technique achieved no better than average performance on a variety of tasks.

An attempt was also made to improve the property filter generation process. A technique employing a mutual information function was used to select one subspace for cloud task PVS and one subspace for cloud task NVC. Twenty-five translations were then used to obtain 25 subspaces for each task. Distribution estimation was used to generate a switching surface for each subspace. The system decision was made by majority vote. Performance was 91 percent on task PVS and 71 percent on task NVC. Similar systems using randomly selected subspaces yielded virtually identical performance, thereby indicating that this mutual information technique is not useful.

A set of 29 known properties was designed for cloud tasks PVS and NVC by Mr. Eugene M. Darling, Jr., NASA Technical Officer for this project. Of these properties, 15 measure general characteristics of the pattern brightness field (e.g., mean gray level, variance of gray level, conditional entropy of the field, etc.), and 14 measure specific characteristics of clouds (e.g., cloud number, size, etc.). The fifteen general properties were applied to the three lunar recognition tasks at the reduced aperture sizes. Recognition systems were designed using these fifteen known properties. The known properties were then augmented by statistical properties in order to test the effectiveness of the augmentation procedure. The following results were obtained:

Task	Aperture	Performance, %	
		15 Known Properties	Augmented System
Lunar Terrain CVC	25 x 25	71.50	85.75
	15 x 15	96.00	96.25
RVR	25 x 25	78.50	80.00
	15 x 15	81.00	82.75
CVR	25 x 25	92.00	97.00
	15 x 15	81.50	96.00
Clouds NVC	75 x 75	90.60	92.75
PVS	75 x 75	97.00	94.25

Thus, eight augmentation experiments were performed. In two of these (in which performance with the known properties alone was excellent) no improvement or a performance decline was obtained. In two cases small improvements (judged not to be statistically significant) were noted. One case of a moderate increase (significance at the 92 percent level) occurred; and three cases of large improvements (significance at levels above 99.9 percent) were achieved.

With the best choice of aperture, property filter set, and decision network, the performances obtained on the five tasks were:

Task	Performance, %
Lunar Terrain	
CVC	96.25
RVR	84.50
CVR	99.50
Clouds	
NVC	92.75
PVS	97.00

These figures seem sufficiently high to warrant further development of a spaceborne recognition system.

A study was conducted to determine the accuracy with which a recognition system could position the boundary between two regions of homogeneous cloud cover. To accomplish this, approximately 35,000 different montages (Section 3.3.8) were created, each of which contains parts of patterns from two different cloud classes. The data from 20,000 of these montages were subjectively evaluated to determine the effect of the presence of a sharp discontinuity. This effect was judged to be of little importance. The data from 14,592 of the montage patterns (those created from pairs of well classified patterns from different classes) were computer analyzed to determine the accuracy of boundary placement as a function of the spacing between consecutive sampling apertures. It was found that by interpolating the discriminant function values (i.e., the input to the decision function), the average error in

boundary placement could be maintained between 8.6 and 10.2 raster elements for all aperture spacings up to 75 raster elements (the no-overlap case). At the resolution used, a raster element corresponds to about 2.3 miles. If, instead, the decision of the recognition system is interpolated, then the average error is more than 20 raster elements at a spacing of 75. To obtain an average accuracy of 10.2 in this latter case, the spacing between apertures would have to be 14 raster elements or less.

Three possible mechanizations of a spaceborne recognition system were designed. The system itself is divided into two parts — (1) an input device and (2) a decision network. The mechanizations are designated in accordance with the manner in which the decision network is implemented. The three systems considered are: (1) parallel/analog, (2) sequential/hybrid, and (3) sequential/digital. The term "parallel" refers to the fact that all quadratic computing elements are individually implemented. In the sequential approach only one quadratic unit is implemented at a time and the computation process is repeated the required number of times, using the same computing element. The term "analog" refers to the analog nature of the voltage from the input device to the logic units. "Hybrid" implies that both analog and digital voltages are supplied as logic unit inputs. "Digital" means that the input voltages, as well as the weights and thresholds to the logic units, are all digital.

The input device common to each system is an image dissector tube which views the scene and supplies subsections of a TV frame to a decision network for classification. In the parallel/analog system the analog output from the photomultiplier of the image dissector provides the input signal via a sample and hold circuit. For the other two systems an A/D converter converts the image dissector output voltage to digital form for subsequent input to the logic units. All of the systems require a memory to store the connectivity of the system, the quadratic and linear weights, and the thresholds. Table I summarizes the characteristics of these three implementations of a spaceborne recognition system.

This implementation study shows that, with current engineering technology and commercially available components, either the sequential/hybrid or sequential/digital recognition system could be constructed with weight,

size, and power requirements compatible with TIROS or NIMBUS-type satellites. Furthermore, any of these spaceborne systems is capable of classifying TV pictures taken by the Automatic Picture Transmission (APT) System in real-time.

TABLE I

SUMMARY OF IMPLEMENTATION STUDY

System	Subsystems				Processing Time per Picture Strip (Seconds)	Duty Cycle	Volume (Cubic Feet)	Weight (Pounds)	Average Power Dissipation (Watts)	Advantages
	Input Device	Computing Element		Memory						
		Type	Quantity							
Parallel/Analog	Image Dissector	Analog Multipliers	6 x 256= 1536	Fixed Resistors and Hardwired Connections	6.29	26%	3.7	50	180	None*
Sequential/Hybrid		Hybrid Multipliers	33	Diode Matrix	2.25	9.3%	2.1	32	37	Simplicity, cost, power, size, weight
Sequential/Digital		Arithmetic Unit	1	Cores	2.45	10.2%	.9	26	30	Flexibility, power, size, weight

* (1) Best speed if photo sensor array is substituted for image dissector (less than 110 msec/strip).

(2) With stored coordinate data and image dissector (100 msec/data/point), the parallel analog system speed may also be increased (less than 1.8 seconds/strip).

CONTENTS

FIGURES

TABLES

1.0 INTRODUCTION

1.1 Purpose and Scope of the Program

On April 1, 1960, TIROS I was launched into orbit to prove the feasibility of meteorological satellites. A succession of TIROS and NIMBUS satellites have supplied meteorologists with a staggering number of photographs. A feeling for the dimension of the collection, transmission, and processing tasks may be gained by considering that the television cameras on a NIMBUS satellite deliver 96 pictures with a total of 5×10^8 bits of data per orbit. All of these data are stored aboard the spacecraft and transmitted to data acquisition stations on earth. Transmission time is only a few minutes. The Mariner IV spacecraft, on the other hand, required 8 1/2 hours to transmit a single picture to earth. From Jupiter, the transmission of a single Mariner-type picture over the same data link would take about one month.

As an approach to increasing the transfer rate of significant video information from unmanned spacecraft, the NASA Electronics Research Center is investigating the possibility of performing pattern recognition prior to transmission. It is envisioned that the data to be transmitted would then be simply a few numbers characterizing the imagery in a sampling aperture. From these, the boundaries separating large homogeneous areas could be computed and the original scene could be reconstructed on earth from idealized models of the pattern classes.

Under contract NAS 12-30, Astropower Laboratory of Douglas Aircraft Company investigated the feasibility of such a spaceborne recognition system. Two aspects of feasibility are considered – (1) whether or not a recognition system capable of classifying imagery with sufficient accuracy could be designed, and (2) whether or not a spaceborne recognition processor compatible with spacecraft system requirements could be constructed.

1.2 Design Philosophy

The adaptive design techniques are considered in the context of the design philosophy illustrated in Figure 1. The process of recognition is composed of four parts – signal conditioning, known property extraction, statistical property extraction, and computation of the decision.

The purpose of signal conditioning or preprocessing is to build into the system the required invariances prior to the actual recognition, to

enhance the pattern to ease the recognition process, and to format the data for further processing. Techniques such as Laplacian filtering for edge enhancement, scanning, magnification control, autocorrelation, resolution regulation, and obtaining correlations against a variety of orthogonal and non-orthogonal functions are examples of signal conditioning methods which have been applied in pattern recognition.

From the conditioned signal, or from the raw data, properties which are known to be, or are suspected of being, of value in performing the recognition are extracted. This phase of the design requires careful study and analysis of the patterns. Together with the signal conditioning phase, it provides the system designer an opportunity to transfer some of his knowledge and experience in performing the recognition tasks to the system. The results obtained are, however, specific to the task at hand and are not readily extended to new pattern recognition tasks.

The list of known properties may not be adequate for high performance on the task at hand, or may not be suitable for the nature of the decision function. In these cases, the list of properties must be expanded. This is accomplished by statistically analyzing samples of patterns of each class. The value of the properties derived from the analysis of these samples is dependent on the representativeness and size of the samples, and the signal-to-noise characteristics of the sample patterns.

The number of possible property profiles in general will be too large to permit cataloging them. Therefore, the classification to be assigned by the system to an unknown pattern must be computed from its property profile. It is in the area of specifying the decision function based on the sample patterns that most research in adaptive pattern recognition has been performed. For the most part, linear and piecewise linear decision functions have been considered. Research on the design of decision mechanisms and on statistical property extraction provides more general purpose results than does work done in signal conditioning and known property extraction.

1.3 Areas of Investigation

Pattern recognition systems were to be assessed experimentally on recognition tasks which might arise in spacecraft imagery. To accomplish

this empirical evaluation, five recognition tasks were defined on seven classes of patterns. Four of the pattern classes represented features of the lunar terrain, the remaining three classes being textural patterns occurring in satellite cloud imagery. The initial experiments were directed toward comparatively evaluating the techniques. For these experiments, the signal conditioning consists of a limited degree of size and position normalization, resolution regulation, and contrast and brightness control.

Six adaptive techniques for designing the decision mechanism were selected. In addition, two methods for statistically designing property filters were chosen. Both of these methods provide binary output property filters, one technique using linear switching surfaces, and the other, quadratic surfaces. The generation of complex switching surfaces using a distribution estimation technique was also attempted as was the optical design of property filters based upon the sampling of the spatial frequencies of the patterns.

An important area in pattern recognition, which had not received any attention, is that of statistically designing property filters to augment a set of known property filters. A set of known property filters for the cloud pattern tasks was designed by Mr. Eugene M. Darling, Jr., NASA Technical Officer for this program. A subset of these property filters was applied to the lunar data. Augmented systems were designed for both cloud pattern tasks, and the three lunar feature tasks at the reduced aperture sizes.

The accuracy with which the boundary between adjacent cloud patterns of different classes could be determined was also investigated. This task is relevant to the extent of overlap between "looks" at a scene required for accurate boundary delineation. Montages of pairs of (correctly classified) test patterns were generated, and the discriminant function of the recognition systems as a function of the portion of each pattern was examined.

Finally, methods for implementing the system design for on-board processing of video data were considered. Of particular concern were the performance of the classification task under the constraints of decision time, accuracy of the mechanization of the property filters and decision function, and the weight, volume and power consumption of the resulting system.

2.0 PATTERN RECOGNITION SURVEY

2.1 Introduction

A pattern recognition system accepts patterns as inputs and produces classifications of those patterns as an output. Generally the input pattern is represented by a finite number of measurements taken from an actual object or signal and the output by one of a finite set of numbers. In some special cases in which analog processing occurs, this is not exactly true, but this exception is not especially significant in practice.

In any practical problem, in which the number of possible input patterns is enormous, dictionary techniques in which the pattern is "looked up" in a table are inadequate. The decision must be computed from the measurements. In some relatively trivial problems a linear or a simple nonlinear discriminant function of the measurements is adequate for producing the decisions. In some rare instances, success can be obtained by correlating the pattern measurements against prototype patterns.*

For more interesting pattern recognition problems, these two methods cannot cope with the complex distributions of patterns in the measurement space. In more effective methods, transformations are applied to the measurement vectors, in order to represent the patterns in a new form for which the correlation or discriminant function methods are more suitable. In the generally accepted terminology, properties of the pattern are extracted from the measurements, and the pattern is represented by a property profile.

As is apparent, this is a convenience. The computation of a property profile, followed by the computation of a simple discriminant function from this profile, is equivalent to the computation of a more complex discriminant function. This formulation provides a rational means for synthesizing a complex discriminant function from simple elements.

The process of extracting properties from the raw data is often considered as a sequence of simpler processes as well. For the discussions which follow, this process will be considered as being composed of two

---

*For example, the automatic map matcher, ATRAN.

phases — a preprocessing phase and a property filter design phase. Clearly, a greater breakdown might be considered.

The total recognition system is thus considered as being composed of four parts. The input device has the function of obtaining the original measurements from the object or signal. The preprocessor serves the function of formatting the data, and possibly performing such general operations as edge enhancement, conversion to the frequency domain, normalization of gray scale and so forth. Unlike preprocessing, the property extraction phase is abstractive; that is, the patterns are represented by a property profile vector of significantly lower dimensionality than the measurement vector. The discriminant function phase provides the final abstraction to a simple decision.

The boundaries of this division into four parts are often quite nebulous and often the classification of the phases of a design technique are quite arbitrary. The experimental program was primarily concerned with the abstractive processes — the design of the property filters and the construction of a discriminant function.

A considerable body of literature exists on input devices. Most of these are special devices to serve as inputs to character recognition systems. Since the input devices are for the most part more complex than the rest of the character reader, it is perhaps natural that these devices should have been emphasized. Input devices have been given imaginative treatment, possibly because it is the most hardware-oriented phase of the recognition process. The devices suggested for optical data incorporate such devices as flying spot scanners, photosensor arrays, lasers, fiber bundle arrays, and image duplicating arrays of lenses.

Preprocessing is, or should be, an attempt to provide invariance mechanisms for the recognition system, and to provide a representation of the input data which simplifies the form of the decision function required. Thus, preprocessing might be used to provide a pattern representation that does not change if the actual object is subjected to a translation or rotation, or if the illumination level changes, etc. As a literature survey indicated, however, this phase often takes the form of a random reshuffling of the input data.

Current methods in property extraction may be divided into three types. Known property extraction, in which the design consists of finding a mechanization for extracting properties which the designer knows to be of value to the problem, is perhaps the most effective of the three, since it is applied to problems of which the designer has considerable knowledge. Statistical property extraction utilizes statistical analyses of sample patterns to derive the properties. Random property extraction uses randomly selected properties, depending upon the novelty of some of the properties thus obtained to offset the lack of efficiency of the method.

An attempt has been made to classify the methods for designing discriminant functions into six categories. These categories are not disjoint, and often a choice has to be made between two or three in order to classify a particular technique. These categories are:

(1) Adjustable Linear Discriminants — Linear discriminants are obtained by recursive processing of a set of sample patterns.

(2) Adjustable Nonlinear Discriminants — These are generally piecewise linear discriminants obtained by recursive processing of the sample patterns.

(3) Correlation Techniques — The discriminants are taken as the maxima of sets of linear discriminants. The linear discriminants are usually obtained by statistical processing of sample patterns.

(4) Probabilistic Techniques — Randomized decision functions are obtained by recursive processing of the sample patterns.

(5) Sequential Techniques — A decision tree is utilized, in which the effect of each test (property filter output) is determined by the results of all preceding tests.

(6) Statistical Techniques — This category included all satistically based methods which would not fit into one of the above categories.

2.2 Input Devices

The various implementations of input systems for video pattern analysis tend to obscure the fact that there is a basic similarity in these systems. The individual patterns may be examined by a vidicon, flying spot scanner or other imaging tube. The pattern is scanned by the reflection of an electron beam and the resulting video data processed directly or converted to

digital form. The pattern may be segmented by an array of light sensitive devices such as photo resistive or photo conductive cells or photomultiplier tubes. The video output of the input sensor can then be fed to a storage device or converter for further processing.

For the array of photosensitive elements the entire pattern may be illuminated, and the light reflected from the scene or transmitted through the film and imaged onto the sensor array. The binary (for black and white patterns) or analog (for gray scale patterns) output of each sensor element can be stored or used directly as an input to the logic of the recognition system. In addition, a single column or row of photosensitive elements can be scanned across the illuminated scene and the resulting element outputs sequentially stored.

The flying spot scanner device is used to direct an electron beam onto a scene, and the modulated beam is detected by a photomultiplier tube. The resulting output is amplified and can be imaged onto a storage tube for optical processing or stored for analog processing or digital conversion. Imaging tubes, such as vidicons, are used to sense the pattern directly. The video signal caused by the modulation of the electron beam as it sweeps across the tube face can also be processed or stored directly in analog form or converted to digital form. The accessing of individual elements on the tube face or the positioning of the flying spot can be accomplished by using a programmable sweep circuit. This allows the selection of any sensor location consistent with the input connection of a logic unit and enables the detailed examination of the image without storage requirements.

These two devices, the image tube and the photosensitive array, form the bulk of the devices proposed for optical inputs. There are, however, a large number of novel devices which have been proposed or constructed. These include a zoom lens-fiber optic bundle-photosensitive array combination; a TV camera-monitor-collimating lens-array of plastic lenses-photosensitive array combination; lasers for spatial filtering, scanners using moving reflective surfaces and one or more photosensors; and devices using templates or moving film strips (for property list correlations) in front of photosensors.

A novel approach to an input system design includes a matrix array of avalanche diodes. Each individual diode (or a number of diodes) is accessed by energizing the appropriate row(s) and column(s). This causes that diode(s) to fire. The diodes emit a narrow beam of high intensity light that can be transmitted through a film. The modulated light beam(s) is sensed by a photomultiplier and forms the individual input (or summed input) to a logic unit.

Applicable References

2, 8, 9, 18, 19, 29, 56, 59, 62, 65, 72, 74, 78, 83, 86, 89, 93, 97, 111, 121, 125, 136, and 137.

2.3 Preprocessing

The preprocessing component is concerned with

1. transforming the output of the sensor into a form that emphasizes certain properties of the input necessary for recognition,
2. presenting a property list in parallel to the recognition device.

Frequently the preprocessing (or property extraction) operation is included in the recognition device. However, at present the feeling is that the transformation occurring between raw input signals and the decision process is crucial, and currently this aspect of the problem is receiving much attention.

As an example of such a transformation consider the case where the output of the sensor is a continuous function of time. A time-segmented portion of this signal can be represented as a series with respect to some orthonormal system of functions (e.g., sines and cosines — yielding a Fourier series expansion). The transformation T, in this instance, takes the continuous input f(t) into a finite set of the coefficients a_n in the series expansion, i.e.,

$$f(t) = \sum_{n=1}^{K} a_n \varphi_n(t),$$

$$Tf = (a_1, a_2, \ldots, a_K).$$

The reduction in dimensionality thus effected is, of course, desirable. However, a more important consideration is that there should be a close association between the input patterns to be recognized and the coefficients generated. That is, the significant features of the class of patterns should bear a strong relation to the orthonormal base so that the coefficients do indeed simplify the recognition task. The usefulness of a given orthonormal base can be measured in terms of the degree to which its member functions represent the redundant features in the set of patterns to be recognized or classified. By collecting several sets of coefficients associated with successive time-segmented portions of the input, a larger list is obtained which contains information concerning the variation of the coefficients in time. For example, if Fourier series is used, the coefficients in this large set contain both time and frequency information. A plot of the absolute value of these coefficients versus time and frequency will yield an approximation to the time-frequency history of the original input. This technique has been used in discriminating between the vowels and consonants in human speech.

Many pattern recognition problems are "spatial" in character, i.e., the initial inputs to the system are two-dimensional patterns. The recognition task usually is to classify certain figures or "objects of interest" covering only a portion of the two-dimensional field. The presence of additional detail in the remainder of the field causes a background "noise" component to appear in the output of whatever device is used to "sense" the pattern. Preprocessing is concerned with techniques which provide some or all of the following to some degree:

1. Invariance to position and size of object of interest in the field of view.
2. Enhancement of certain features in the pattern to ease the task of property extraction and recognition.
3. Reduction of background noise.
4. Property extraction.

If the object of interest is centered in the field, background may be suppressed by a masking technique which reduces the area of the field being sensed. If the sensor is a flying spot scanner the same effect can be had by "underscanning," i.e., scanning only a particular area about the center of the field.

For a flying spot scanner the output of the scanner can be subjected to such operations as gradient or Laplacian filtering which tends to enhance edges and contours. These techniques require the scanning of each point of the field four times, back and forth in the horizontal direction and up and down in the vertical direction. For the Laplacian filtering operation the output of the scanner is differentiated twice and then an average taken over the four scan paths. The result is equivalent to applying the Laplacian operator $\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$ to the original image (brightness function). Since in the regions of the field where pattern intensity $f(x, y)$ is constant $\nabla^2 f(x, y)$ is zero, the pattern which could be reconstructed from the processed scanner output would consist of the fine detail in the original. That is, boundaries of highly contrasting regions are displayed against a dark background. Gradient filtering is similar but involves only a single differentiation and a squaring operation before averaging over the four scan paths.

In many applications the output of the scanning device is sampled at the Nyquist rate and the set of sample values is used directly. If a high resolution scanner is required the property list may be extremely large, thus complicating the data handling and the design of the recognition device. The determination of the minimum resolution necessary for classification is an important part of the overall pattern recognition problem. This is especially true for problems involving complex spatial inputs such as the recognition of vortices in cloud patterns. No satisfactory method for determining the minimum resolution required is known at present. When high resolution inputs are supplied as photographs, the use of a slow scan flying spot scanner can appreciably reduce the data rate and ease the task of data handling.

Several special purpose computers oriented toward spatial problems have been proposed in the literature, e. g., Unger's SPAC.[143] Briefly, this computer consists of a two-dimensional array of logic modules each of which may receive inputs from the modules immediately above, below, to the left, and to the right of it. Also, there is a master control from which each module receives commands. Spatial inputs are converted into a binary-grid

representation and then stored in the spatial computer. Programs have been written for such computers (or they have been simulated on a conventional computer) which make possible the detection of corners, edges, edge sequences, and the convexity or concavity of simple spatial figures. These may be used as properties in the classification problem.

Optical filtering as a technique for property extraction has received considerable attention. A simple technique consists of projecting an image through a strip of film which possesses regions with different transmissivities. This effects a weighting of the spatial input. A photosensor is used to collect the total amount of light passing through the film. This is equivalent to correlating the spatial input with the function representing the transmissivity of the film strip. Repeating this with a succession of different transparencies amounts to measuring the degree to which the input possesses a given property. The set of photosensor outputs corresponding to the collection of transparencies forms the property list.

It is difficult to specify a transformation or sequence of transformations to condition the sample set of patterns in a way which will provide all the desirable features (1) through (4). An interesting method for choosing properties for spatial patterns which is invariant under translations and size variations of the object of interest consists of describing the patterns by means of a finite number of their normalized bivariate central moments. F. L. Alt[4] has described such a technique and applied it to the character recognition problem. The characters were assumed to be of the binary-grid form. For more complicated spatial inputs involving many gray levels and much background, the usefulness of this technique is questionable.

The easiest way to provide a degree of rotational invariance is to augment the sample set of input patterns by adding a number of rotated versions of each pattern. A similar procedure can be used to provide a degree of translation and size invariance.

Applicable References

4, 8, 30, 43, 50, 51, 57, 68, 81, 92, 105, 120, 126, and 136.

2.4 Recognition System Design Techniques

2.4.1 Adjustable Linear Discriminants

An adjustable linear discriminant results in a partitioning of an n-dimensional space with a hyperplane. Adjusting the discriminant allows parallel displacement and orientation variation of the hyperplane. The discriminant can be implemented by the linear input threshold device (or unit) that has been so popular in pattern recognition research. The simple form of the function and its ease of mechanization makes it attractive from both an analytic and an experimental point of view. A transformation of the signal space (which is usually of high dimensionality, approximately equal to the number of resolvable elements in the sensor) is performed and the adjustable discriminant partitions this transformed signal space (recognition space) under the control of a learning procedure.

Design of a recognition system assumes that one is given a set of property vectors (i.e., pattern vectors) which are divided into classes. The recognition system is to implement this division of pattern vectors into classes so that it can be applied to new pattern vectors derived from the same source as were the given set of vectors. The basic building block of the recognition systems to be discussed is the linear input threshold device. It takes a weighted sum of the vector components of the vector used as its input, this sum being called its input sum, emits a +1 output if its input sum was greater than 0, and emits a -1 (or 0) output if its input sum was less than 0.

A recognition system consisting of one linear input threshold device, whose input vector is a pattern vector, has been used to try to separate two classes of pattern vectors. Let A and B denote the two classes, let a +1 linear unit output mean that the linear unit is recognizing its input vector as a member of A, and let a -1 output mean that the linear unit is recognizing its input vector as a member of B.

One method of adjusting the linear input threshold device's weights, called forced learning, adds each input vector component to the corresponding linear unit's weight if the input vector is a member of A, and subtracts each input vector component from the corresponding linear unit's weight if the input vector is a member of B. Forced learning has the

advantage of being independent of the sequence of pattern vectors presented to the linear unit but cannot guarantee to solve the recognition task if the task is solvable by one linear unit.

A second method of adjusting the linear input threshold unit weights, called error correction, does not change the linear unit's weights if the linear unit correctly classified its input vector. If the input vector was actually in A but the linear unit indicated that it was in B, then error correction adds each input vector component times a constant, c, to the corresponding linear unit weight. If the input vector was actually in B but the linear unit indicated that it was in A, then error correction subtracts each input vector component times c from the corresponding linear unit weight. For the correction of any one error, c remains at a certain positive value; but between errors, c may be changed. If c is a constant or if c is always chosen to satisfy certain conditions, error correction is guaranteed to solve the recognition task in a finite amount of time (if the task can be solved by one linear unit) and if one continues to present each pattern vector to the linear unit. Even if one linear unit cannot solve the task, using the above procedure will result in approaching the state where no errors are made on those pattern vectors which are not involved in a contradiction. Whether the linear unit can solve the task or not, if c is a constant or if c is chosen to satisfy certain conditions, the linear unit's weights are bounded. Varying c in the proper manner has been empirically shown to hasten convergence on relatively difficult tasks. Error correction has the disadvantage of possibly poor generalization to new pattern vectors because it tries to eliminate all errors on the given pattern vectors.

Another method of adjusting the linear unit's weights, called the Widrow-Hoff minimum mean square error algorithm, assumes that each pattern vector component is either a +1 or a -1. The error for a pattern vector is defined as the desired linear unit output minus the linear unit input sum. At each presentation of a pattern vector to the linear unit, this algorithm adds each input vector component, times the error, times a constant, to the corresponding unit's weight. It has been shown that if the mean square error is a hyperparabolic function of the unit's weights, then this algorithm is equivalent to searching for the minimum of the hyperparabola using the method of

steepest descent. This algorithm tends to cluster all input sums for members of A around +1 and all input sums for members of B around -1. This algorithm may be modified to adjust the linear unit's weights only if the unit's output is wrong.

In the preceding methods used for adjusting the linear unit's weights, the given pattern vectors were examined sequentially. The following methods examine all of these vectors at once. Each of the following methods defines for each given pattern vector a cost or loss function based on the linear unit's input sum (or output) for this particular vector. These methods then attempt to minimize the total loss for the given pattern vectors.

One of these methods, called the method of steepest descent, adjusts the linear unit's weights a step at a time. At each step, each component of the gradient of the total loss times d, a constant, is added to the corresponding linear unit's weight. At any one step, d remains at a certain negative value; but between steps, d may be changed.

A similar method, called iterative design, adjusts the linear input threshold device's weights one at a time. For an adjustment of the i-th weight, let the partial derivative of the total loss with respect to the i-th weight be denoted by p, and the i-th weight modified such that p changes to gp, where g is a number whose magnitude is less than 0.99. Between weight adjustments, g may be changed.

Consider the case where a pattern vector's loss is equal to the exponential of minus the linear unit's input sum if the pattern vector is in A, a pattern vector's loss is equal to the exponential of the linear unit's input sum if the pattern vector is in B, and there is a constant b such that every weight has been adjusted within every b consecutive weight adjustments. For this case, iterative design is guaranteed to solve the recognition task in a finite amount of time if the task can be solved by one linear unit. Even if one linear unit cannot solve the task, using the above procedure will result in approaching the state where each pattern vector which is not involved in a contradiction has a loss of zero. Whether the linear unit can solve the task or not, if the column vectors of the matrix whose row vectors are the members

of A and B are linear independent, then the total loss approaches its smallest possible value.

Discriminant analysis is a method which with certain assumptions specifies the linear unit's weights which will minimize the total loss. One can use this method if one knows or can estimate from the given pattern vectors the probability of class A, the probability of class B, the probability of any observed pattern vector if one knows that it is in A, the probability of any observed pattern vector if one knows that it is in B, the loss associated with recognizing a pattern vector as a member of A when it is actually a member of B, and the loss associated with recognizing a pattern vector as a member of B when it is actually a member of A. Then depending on the assumptions made in order to obtain these probabilities, the specified decision function may be realizable by a linear unit. For example, if one assumes that the components of a pattern vector are either +1 or 0 and are independent, i.e., uncorrelated, then the minimal loss decision is realizable by a linear unit whose weights are conveniently expressed as logarithms.

Discriminant analysis can be used to design a recognition system which will classify a pattern vector as being a member of one of k classes. It can be used under conditions similar to those mentioned in its application to the two class problem. And its classifications can be approximated by a method such as the Fix and Hodges method, which only assumes that the pattern vector probability density functions exist and are continuous within each class.

Another way of attacking the problem of k classes is to use a layer of linear units which have the pattern vector as their input vector. Then each of the k classes is indicated by a particular pattern (or partial pattern) of the linear unit's outputs. Error correction can be successfully applied to such a system. An example of such a system is a class pair separator, which uses $k(k-1)/2$ linear units.

Consider the problem of designing a recognition system which will classify a pattern vector as being a member of one of k classes. If there is only one prototype or typical pattern vector per class, a linear machine can be used. A linear machine contains two layers, the first layer consisting of k linear input threshold devices (one associated with each class) which have the pattern vector as their input vector. The second layer contains a maximum selector; the system classification of a pattern vector is the class associated with the linear unit whose input sum is the largest. Many ways have been used to reinforce such a system. One way is an error correction procedure which adjusts the weights of two linear units whenever the erroneous classification of a given pattern vector occurs. If the i-th class is designated when the pattern vector really is in the j-th class, each input vector component times c is added to the corresponding weights of the j-th unit, and each input vector component times c is subtracted from the corresponding weight of the i-th linear unit. If one replaces "one linear unit" and "the linear unit" by "this linear machine," then all of the statements made about the error correction method of adjusting the linear unit's weights are true for this error correction procedure. A minimum Euclidean distance classifier is a linear machine.

Systems have been designed using adjustable linear discriminants in both analog and digital form. The analog mechanization is cumbersome since the weight variability is to be retained. In some cases the system is simulated on a digital computer and the resulting design constructed with deterministic values in analog form. Digital implementation involves the use of the normal computer components of an input system, an arithmetic and memory unit and an output system. In this case the parallel operation of sampling the input field, and adjusting the weights and forming the discriminant function is done sequentially with the various values being stored and accumulated during the process.

Applicable References

8, 15, 16, 22, 25, 37, 39, 42, 48, 55, 60, 70, 71, 73, 76, 77, 78, 79, 82, 87, 96, 99, 100, 101, 102, 111, 116, 120, 124, 125, 128, 131, 138, 139, 141, 142, 148, 149, 150, 151, and 152.

2.4.2 Adjustable Nonlinear Discriminants

Adjustable linear discriminant functions, as described in Section 2.4.1 operate to classify patterns of n inputs. Each input is weighted and included in the analog sum which is compared to a threshold to determine which of two output states the unit will assume. The inputs may be derived from the outputs of property filters or pattern processing devices. In some cases the pattern itself may be classified directly. In either case the linear threshold logic unit attempts to separate the patterns by passing a hyperplane in the n-dimensional input space.

The class of nonlinear discriminants includes devices which attempt to separate patterns by passing nonlinear surfaces through the input space. Classification is again based upon which side of this surface a pattern lies. An example would be a unit forming a hyperquadratic surface in the input space. Quadratic or higher order surfaces may be implemented by a single threshold logic unit. Each term in the quadratic polynomial is considered as a separate input to a linear logic unit; the weights on the input lines would represent the coefficients of each term. These parameters may be adjusted by any of the methods used for linear discriminant functions. When the inputs are binary, the cross product terms may be easily obtained using "and" gates.

The piecewise linear surface, composed of several planar surfaces, is another form of adjustable nonlinear discriminant often implemented. Sharply varying or closed surfaces may be achieved with these systems. The MADALINE would be representative of such a system. The MADALINE consists of m linear threshold units each of which is connected to all of the n inputs. The binary output decisions are collected by some form of fixed logic to implement the overall binary decision. The use of various fixed logic outputs for the MADALINE depends upon the type of classification desired.

When a majority vote taker is used, the training procedure requires the input pattern to correlate with a majority of the paradigms stored in the threshold units' input weights. If a decision is correct, no adjustments are made. If a decision is incorrect, a number of units sufficient to

complete the correct majority decision are adapted toward the correct output. (The total number of threshold units must be odd to avoid a tie vote.) The adaption procedure for the individual threshold element is identical to the Widrow-Hoff algorithm described in Section 2.4.1.

An alternate system which should work well where the positive pattern class is strongly multimodal, uses an "or" gate for the output logic. This training procedure adds the positive patterns to the weights of the logic unit with which it is most likely correlated. When a positive pattern is being classified, the threshold unit whose analog sum is closest to threshold is trained toward +1. When a negative pattern is misclassified, all units giving +1 output are trained toward -1. When patterns are classified correctly, no adjustments are made. Replacing the "or" gate with an "and" gate would accomplish exactly the same function if the pattern classes were reversed. The majority, "and" and "or" gates are special cases of a general fixed logic unit with the properties of yielding a +1 output if at least k threshold units have +1 outputs; otherwise the output is -1. k is a positive integer less than or equal to the total number of inputs to this unit. This parameter may be varied to determine the most advantageous classification scheme.

Another configuration uses the parity function for an output gate. If the number of threshold units giving positive decisions is odd, the output is positive; if the number of units giving positive decisions is even, the output is negative. The advantage claimed for this method is a stable adjustment procedure. When a pattern is incorrectly classified, it is not necessary to adjust more than one threshold unit to obtain a correct decision. This results in the least disturbance to the system with each adjustment. It is not clear how the various threshold units will relate to the pattern distributions.

Several alternatives have been suggested for cases where more than two pattern classes are to be separated. One or more threshold logic units may be assigned to each class to act as paradigms for that class. The output logic selects the class assigned to the threshold unit which correlates most highly with the input pattern. A mode-seeking method of adjusting the weights of the paradigms has been suggested. The weights in

each threshold unit are adjusted to place the weight vector in the center of a cluster of similar pattern vectors.

An alternate procedure involves using several adjustable nonlinear networks whose outputs form a binary classification "word." The individual nonlinear discriminants are adjusted to present 1's or 0's in the proper bit positions when an input of a particular class is presented to the machine.

The work performed by Barus[10] may also be considered in the context of adjustable nonlinear discriminants. Barus' approach is to use the maximum likelihood criterion in classifying an input vector. This requires estimates of the probability density distribution of the input vectors for each class. To accomplish this, several "best" representations are stored as paradigms for each class. Their representativeness is evaluated as the machine experiences identified input specimens. New specimens are stored and less useful ones discarded as a result of this experience. His method of estimating the probability density distribution of each class at the point occupied by the input vector is to correlate the input vector with each of the stored paradigms of a given class and count the number of times the correlation is greater than a certain percent. This number divided by the total number of paradigms of that class is $\tilde{p}_i$, the estimate of the probability density function. He assigns the input pattern to the class having the largest $\tilde{p}_i$. This classification process is analogous to storing each paradigm in the weights of a separate linear threshold unit, setting the threshold for the desired percent of correlation, and taking a majority vote of the outputs of all the units.

Applicable References

5, 10, 11, 12, 19, 20, 60, 115, 123, 149, and 151.

2.4.3 Adjustable Properties

The greatest improvements in pattern recognition techniques will come in the area of finding good sets of property filters, or measurements for the discriminant function. Although this area has been receiving considerable attention recently, few workable systems have resulted.

Sometimes the constructors of pattern recognition design techniques choose properties randomly, or at the other extreme, intuitively based on careful study of the particular recognition problem. For the most part, however, these constructors assume that someone will provide them with a set of suitable property filters.

When the designer has a good understanding of the problem, the intuitive approach can yield good property filter sets. This approach has been heavily used in character recognition. More often than not, when the designer was unable to find a satisfactory property set, strong restrictions are placed on the characters that are allowed, and occasionally, they are distorted almost beyond recognition in order to utilize a property set. More recently, human selection is being applied to the photointerpretation problem with some notable success. Even in this case, a technique for adding and deleting properties seem desirable.

The earliest techniques, and ones which are still being studied, for adjusting the set of property filters were evolutionary ones. In these methods, a set of property filters is obtained (most often randomly), and a decision network designed for these. Property filters that are found not to contribute significantly to the decision are deleted from the set. In some methods, additional properties are added to replace those deleted. In these cases, the design process is repeated until some design criterion is met.

More recent techniques use statistical analyses of a small fraction of the data in the sample patterns to derive property filters. These analyses are generally performed under strong distributional assumptions (which usually result in linear or quadratic input threshold devices for property filters). The property filters are, in essence, linear or quadratic discriminant networks designed to base their decision on a small fraction of the data in the input pattern. The assumption that the input data has a multivariate normal distribution, whose parameters depend upon the pattern class, is common. Although this distributional assumption is not usually accurate, the property filters form only part of the system; other aspects of the design technique evaluate the effectiveness of the property filters.

Several techniques combine the above methods. In one system, discriminant analysis is performed on substantial subsets of the input array to produce property filters with a large number of input connections. Those input connections which do not contribute markedly to the property filter's operation are deleted. In another system, many property filters are designed using discriminant analysis, but only those which contribute significantly to the overall system are retained.

In a different approach, property filters are designed which tend to cluster patterns of the same class, while maintaining a separation between the pattern classes. Despite the difference in starting points, the method leads to property filter generation methods which are the same as those in which the multivariate normal assumption is made.

In one method, the property filter set and the decision structures are developed concurrently. A large number of property filters are designed using discriminant analysis on limited portions of the input fields. The most effective of these are retained and a decision structure is designed for these property filters. A new set of property filters is then designed using a discriminant analysis modified to reflect which sample patterns are not being classified properly by the existing network. Only those property filters which form an effective addition to the existing network are retained from the second set. The decision structure is then redesigned and the process repeated until satisfactory performance is achieved.

While this last system seems very reasonable, the design process is very tightly coupled to the task of separating the sample patterns. The technique thus places much higher demands on the representativeness of the sample patterns than does the more statistically based methods described above. The stronger the dependence of a method upon distributional assumptions, the less disastrous are the effects of anomalies in the sample of patterns, but the more disastrous are the effects of inaccuracy of the assumptions.

It is evident from the survey that substantial improvements in the design of property filters will result in great improvements in the design of pattern recognition systems. What is needed is a compromise

between the extent of human design, the degree to which distributional assumptions are utilized, and the degree to which the design is dependent on separation of a particular set of sample patterns.

Applicable References

4, 8, 18, 21, 38, 39, 43, 58, 78, 79, 81, 82, 90, 91, 92, 105, 124, 125, 128, 129, 130, 131, 132, 140, and 141.

2.4.4 Probabilistic Elements

Typical of the systems or techniques that fall into this category are the ARTRON of Melpar and the Probability State Variable (PSV) device of Adaptronics. They consist of devices whose internal states are not deterministic but are controlled by statistical switches whose probability of closure is adjusted during the training period.

Specifically, the ARTRON is an element that realizes functions of binary input variables by training statistical switches to select various combinations of these inputs. Variations of this element have been used in systems such as LANNET (Large Artificial Nerve Net) and SOBLN (Self-Organizing Binary Linear Network). In a single ARTRON the first-layer units consist of "and" gates that form all possible minterms (or maxterms) for the input variables. Each "and" gate has an associated statistical switch; therefore, for n binary variables, 2^n minterm generators and 2^n statistical switches are required, giving the network the capability of generating any of the 2^{2^n} switching functions of n variables. Under the influence of goal directed reward and punish signals, the probability of each statistical switch being open or closed varies between the limits of .01 and .99. The final output is the union, through the action of an "or" gate, of these switches and their associated minterm. The statistical switch portion consists of three components. These are a probability register, an adjustable noise control, and the reward/punish sign control. The probability register consists of an up/down counter whose contents are proportional to p_i. The output of the counter is converted to analog form (with register summing) and biases an adjustable noise control. The output of the noise control, which is p_i and $(1 - p_i)$, triggers a flip-flop to produce a pulse density modulated

output. The final output of the switch is the logical "and" of the flip-flop and the minterm input. The state of the switches then determines the output of the "or" gate and fixed increment reward and punishment signals are presented until the correct minterm is selected. The final state is not deterministic, however, and the noise signal causes the switch output to jitter around any operating point.

Though two-input/one-output ARTRONS may be pyramided with fixed logical elements to form n-input/m-output networks, convergence and stability are not assured. The SOBLN is a special implementation of ARTRON devices which realizes all possible Boolean connections and includes redundancy for error correction (i.e., statistical switch failure). The LANNET is an extension of the n-input/m-output concept with a hardware mechanization. The basic form consists of a 2^n minterm generator with parallel banks of 2^n statistical switches and n output "or" gates. It is implemented with digital hardware and utilizes a 1024-word core memory for storage of 1024 statistical switch probabilities.

A Probability State Variable (PSV) device is defined as an artificial nerve cell featuring statistical operation and with memory being stored as a probability of having certain transfer characteristics. Both the ARTRON, discussed above, and the NEUROTRON, discussed in the next paragraph, are considered as PSV devices. In addition to the PSV, Random State Variable (RSV) devices have also been considered. The PSV strategy begins by performing a trial experiment with one of two alternate values of a parameter. Initially the choices between alternates is made at random with equal probability. As learning progresses, the statistics of selection are progressively biased towards the value producing a more consistently favorable outcome. In the RSV strategy the system begins by making a random experimental change in a system parameter. If the system performance is improved as a consequence of this experiment, the new parameter is retained; otherwise, the change is discarded and a new random experiment, centered about the original state, is attempted.

The NEUROTRON is a PSV device, similar to the ARTRON, with the addition of variable gain and time constant circuits and a capability for handling continuous inputs that are pulse density modulated, as

well as binary coded inputs. It is capable of one of sixteen logic states plus a phase-lag or a phase-lead. The first layer consists of "and" or "or" gates that compute the logical combination of the two input variables; the gate (or gates) that are energized provide a signal to a second "and" (or "or") gate that also receives an input from the statistical switch. In addition, a control signal is provided to the switch from the "active" input gate to allow reward or punish signals to affect that switch. The output from the second layer of logical gates is input to an "or" (or "and") gate. Up to this point the NEUROTRON (except for the nature of the input signal) is identical to the ARTRON. The reward, punish and noise signals, in addition to varying the probability of closure of the statistical switches, are also provided as inputs to the variable gain and time constant circuit. The logical output of the final "or" (or "and") gate is combined with the gain and time constant output to provide the system transfer function. The nature of the actual and desired response determines whether a reward or punishment signal will be generated. The adjustment of the state of the statistical switch is accomplished by biasing the noise signal on the switch as in the ARTRON. For the gain and time constant circuit the new value is made less than the old by a negative noise signal and more by a positive noise signal, the difference between the new and the old being proportional to the noise amplitude. If a reward signal follows a new value, that value is retained; if a punishment signal follows, the old value is retained. Various logical, as well as analog, functions have been generated using small networks of these NEUROTRONS.

Applicable References

26, 27, 64, and 88.

2.4.5 Correlation Techniques

The majority of pattern recognition techniques involve the generation of one or more prototype patterns for each decision class. In general, the choice of a prototype depends on the set of sample patterns in a given class and its interrelation with the sample sets in each of the remaining classes. More specifically, the determination of prototypes is a function of the probability distribution determined by the sample patterns in a given class and its interrelation with the distributions of each of the remaining classes.

Briefly stated, the "matching" or correlation technique consists of correlating an input pattern with each prototype pattern. The decision of membership in a particular class is made when the correlation of the input pattern with the prototype (or one of the prototypes) for that class is greater than all other possible correlations. This section discusses techniques involving more than one prototype per pattern class. Single prototype systems are discussed in Section 2.4.1 and 2.4.6.

One technique for estimating the probability density from a set of sample patterns in a given class may be described as follows.

From the sample of patterns an estimate of the distribution or probability densities is obtained. These estimates are used to evaluate the likelihood ratios (decision making accomplished by comparing the ratios of the probability densities against a constant). The process of estimating the probability densities from labeled samples of known classes is regarded as "learning" while the evaluation of likelihood ratios at points in the vector space corresponding to an input stimulus is called "recognition."

In approximating the probability density function of an unknown class, a cell of prechosen size and shape is created and is centered on the first learning sample. The chosen size and shape is determined by prior analysis of all the training data by computing the mean and variance of these data. In addition to the vector, the estimate of the new density of the cell is also computed and stored. The density is estimated by the fraction of the total number of input vectors that fall in that cell divided by the volume of the cell. The second learning vector is used to generate a second cell similar to the first if it falls outside the first cell. If the second vector falls inside the first cell, the center of that cell is shifted to the mean of the two learning vectors, the density recomputed; hence the shape and size of the cell effectively is adapted from the knowledge about the local distribution of members of the class rather than the original estimate. If the second vector falls "sufficiently" far outside the first cell, it creates a new cell of size and shape obtained from the prior estimate of minimum cell size. If it falls outside the first cell by a small amount, it is temporarily stored. Subsequent vectors are processed similarly. To insure that each cell has enough room to grow,

and to reduce the chance for an overlapping coverage of the same region by more than one cell, an "outer" control parameter is introduced so that a vector not falling inside an existing cell is used to generate a new cell only if it is outside a larger concentric cell. As the cells grow in size, these stored vectors are forced into existing cell structures.

In approximating an arbitrary probability density function, a locally Gaussian assumption is made. A sample is used to specify the mean of the cell at a point and the distribution is assumed locally Gaussian about that point with variance estimated from the density of the sample vectors in that cell. This allows a criterion for rejection or acceptance of other vectors into that cell. This is the same as previous discussions where an acceptance region about a pattern is defined by an ellipsoid in n-dimensional space; the shape and size of the ellipsoid are estimated from the mean and variance of the input vectors which are obtained from a priori knowledge of the pattern classes and updated by the density measure.

A technique currently under investigation involves the use of the function

$$K(y) = \sum_{k=1}^{M} \exp \left\{ -\sum_{i=1}^{N} b_i \, (y_i - x_i^k)^2 \right\}$$

$y = (y_1, \ldots, y_N)$ is an arbitrary point in N-space.

$x^k = (x_1^k, \ldots, x_N^k)$ is the k-th sample property vector (or list) in a particular class.

b_i = positive constants

A point y^o at which K is a local maximum is called a "cluster point." An algorithm has been developed together with a proof of its convergence. This algorithm makes it possible to recover the modes of the (in general) multimodal probability density governing the property lists of a given class. These cluster points are used to construct an estimate of the actual probability density associated with a class. In general these design techniques lead to a collection of prototypes which are correlated against a pattern input vector.

Nilsson [115] suggests an iterative method for finding the modes of a multimodal distribution. A fixed number of prototypes are distributed through the space. The sample patterns are examined one at a time. The prototype which is closest to the sample pattern is modified so that it moves a small amount towards the sample pattern. The size of the increments decrease in time so that the positions of the prototypes become more stable.

The clustering techniques mentioned above are an attempt to find representative prototypes without the number of prototypes required becoming too large. If memory requirements are not as important as performance, a technique which has shown very good results is to save all sample patterns as prototypes. Since patterns near the more significant modes will occur many more times than those in lower probability portions of the distribution, a large number of prototypes (mostly unnecessary) are required to obtain high performance levels. A method has been suggested for obtaining a set of prototypes cyclically, saving as additional prototypes each one which is misclassified by the existing set of prototypes. This method is very similar to the clustering methods above.

Applicable References

9, 10, 12, 24, 49, 80, 115, 122, 127, 128, 129, and 132.

2.4.6 Conditional Probability

Statistical techniques play a central role in the recognition problem where they have usually been employed as an aid in choosing prototypes or in determining the weights associated with logic units in an adaptive system. Chow [31] considered the general character recognition problem as one of testing multiple hypotheses in statistical inference. His technique consists of testing for each character the hypothesis that the observed pattern is the given character against the hypothesis that it is not. A formula was derived for determining the magnitude of the expected risk in making a given decision, i.e., the expression

$$X_j = \sum_{i=1}^{c} w_{ij} p_i F(x/a_i),$$

is evaluated for $j = 1, 2, \ldots, c$ where $F(x/a_i)$ is the conditional probability of the input vector x occurring if the pattern is indeed the i-th character; p_i the a priori probability that the i-th character occurs; and w_{ij} the weight assigned to misreading the i-th character as the j-th one. The minimum risk signal is selected. This technique provides a conceptual framework for designing recognition systems but requires a technique for calculating the conditional probability function.

The conditional probability $F(x/a_i)$ is in general derived in terms of the conditional probabilities $G(x_j/a_i)$ of the components of the vector x. Two difficulties exist. One is finding a method by which the conditional probabilities $G(x_j/a_i)$ may be obtained, and secondly, finding a method for combining them into an estimate of $F(x/a_i)$. Although most investigators realize the undesirability of such an assumption, it is generally assumed that these components are independent.

Some interesting results have been derived for binary vectors X. With the independence assumption, a likelihood ratio test is implemented by using linear discriminants. The j-th coefficient of the i-th discriminant is given by $\ln \frac{G(x_j/a_i)}{1 - g(x_j/a_i)}$, or the logarithm of the odds ratio.

Kanal[82] uses an interesting and potentially very useful representation for the joint probability distribution $p(x_1, x_2, \ldots, x_N)$ where $x = (x_1, \ldots, x_N)$ represents the binary input pattern property vector. This representation, which is due to Bahadur,[167] takes the form

$$p(x) = p_1(x)\, f(x)$$

where $p_1(x)$ denotes the probability distribution of the x_i when the x_i's are independently distributed and they have the same marginal distributions. The function f(x) involves correlations of various orders. By neglecting the higher order correlations in f(x), an approximation to the distribution may be obtained.

The derivation of the conditional probabilities $G(x_j/a_i)$ presents difficulties as well. For binary variables, a customary procedure

is to use the maximum likelihood estimate of the probability, based on a sample. Thus, if the number of occurrences of a_i in the sample is n_i, and if x_j occurs in m_{ij} of these, the estimate of $G(x_j/a_i)$ is $\frac{m_{ij}}{n_i}$. This gives an estimate of $\ln \frac{m_{ij}}{n_i - m_{ij}}$ for the logarithm of the odds ratio above. This estimate gives coefficients whose magnitudes are too large when the number of occurrences or nonoccurrences of x_j are very small, and gives infinite values if the x_j or not x_j does not happen to occur.

Mosteller and Wallace[109] provide an elegant method for avoiding this overemphasis. The conditional probability is estimated using methods derived from subjective probability. An a priori Beta distribution is assumed for the conditional probability, and the estimate is derived from the a posteriori probability. The result simply amounts to starting the counters for m_{ij} and $n_i - m_{ij}$ at α and β (arbitrary parameters of the Beta distribution). Any amount of de-emphasis of rare events can be selected by adjusting α and β.

Applicable References

1, 2, 5, 6, 31, 32, 40, 82, 87, 90, 91, 109, 118, 145, 146, 147, and 167.

2.4.7 Sequential Techniques

As applied to pattern recognition, the term "sequential technique" is used to imply the decision tree method, rather than any manner of implementation of the recognition device. The game of twenty questions provides an example of the decision tree method. The tests which are applied to the input pattern depend upon the results of the preceding tests. The logic decision tree can be duplicated with a nonsequential network in which all tests are always performed, and the decision structure provides the selection of the equivalent sequences, just as it is possible to simulate a parallel logic network with a sequential machine such as a digital computer. These artifices do not, however, change the basic nature of the decision processes.

The sequential techniques range from the extremely complex heuristic programs, such as the General Problem Solver, to the relatively simple ones, such as EPAM. The former appears to be an attempt

to simulate the general thought processes of people; the latter is a mechanization of the twenty questions game. The latter type is most applicable to the current program.

Ideally, each test on a binary decision tree should eliminate about half of the possible cases from further consideration. When the patterns are relatively noise free, and the figure-ground separation problem is not significant, this is accomplished relatively easily. In EPAM, for instance, the tree is developed automatically. Existing tests are applied to a sample pattern until a terminal branch is encountered. If the sample pattern is the same as the pattern at that location, no changes are made. If the patterns are different, a test is devised to separate these two patterns. The terminal point is replaced by a test node, and two terminal branches then replace the one.

This automatic design process is less desirable when the sample patterns are noisy. It is true that a structure could be designed to separate the sample patterns. However, the tests are devised to separate two patterns, and are likely to be based on the noise in those particular patterns. For noisy patterns, the tests should be designed statistically, based on data derived from many samples. If the decision tree has any great depth, the acquisition of a sufficient number of patterns may prove to be troublesome. If, for example, ten tests are to be performed before final classification, and if at least 100 samples of each type of pattern are required for the design of each test, then a minimum of 100,000 sample patterns are required. Another desirable constraint is that the tests performed divide the patterns into coherent sets. It would seem more desirable to have a rational means for choosing the pattern sets to be separated, rather than depending upon their order of occurrence in the sample set of patterns.

These remarks are not intended to indicate that the sequential approach is not of importance to the pattern recognition problems of this study. What is indicated, however, is that the decision tree be constrained to only a few levels, and that the structure of this tree be established by the system designer.

In the Lunar experiments performed, four types of lunar features were to be identified. These are craters with and without central elevations, rima, and wrinkle ridges. Nearly all the systems designed incorporated three binary decision networks, one to separate the craters from rima and ridges, one to separate rima from ridges, and one to separate the two types of craters. This is clearly a decision tree of the type suggested above.

It should be noted that a number of sequential testing procedures have been recommended in the literature in connection with character recognition. These schemes, however, are intended to be used with relatively stylized, noise-free, isolated characters. As with the vast majority of the work which has been done on character recognition, the techniques make full use of the nature of alphabetic characters and the designer's great familiarity with them. The design tricks thus employed are of little utility for the general pattern recognition problem.

Applicable References

18, 37, 46, 47, 52, 112, and 134.

3.0 SOFTWARE FEASIBILITY

3.1 Introduction

The pattern recognition techniques under consideration share a common structure. As shown in Figure 2, properties are extracted from the conditioned pattern, and the decision function is computed from the property profile of the pattern.

A limited amount of pattern conditioning was performed for the experimental portion of this program. This amounted to specifying the resolution and aperture size to be used and maintaining some control over the image brightness and orientation to account for prior knowledge of sun direction and illumination. To prepare a set of patterns for the simulation experiments, an acquisition, validation and conversion process had to be performed. Section 3.2 discusses the tasks involved in obtaining an adequate pattern file and the concomitant problems.

In Section 3.3.1 the six design techniques used in the initial experiments (Section 3.3.4) are discussed in detail. Additional techniques used in later experiments will be described in the appropriate sections. In these initial experiments, two techniques were used to generate the property filters. Each property filter has a limited number of input connections and hence examines only a portion of the input data. With both techniques, the output of a property filter is binary. With one technique, the switching surfaces of the property filters are linear, and are based on the average brightness of the pattern classes at the appropriate points in the input field. With the other technique, the covariances of the brightnesses are considered, and the switching surfaces which result are quadratic. Detailed descriptions of the techniques are given in Section 3.3.2.

The six techniques for specifying the decision function were applied to the property profiles determined by the two methods mentioned above. Five of these result in linear decision functions (of the property profiles). In the sixth technique, a MADALINE system, the decision function is piecewise linear.

The two techniques for designing property filters were applied to three recognition tasks on the lunar data. For each of the six sets of

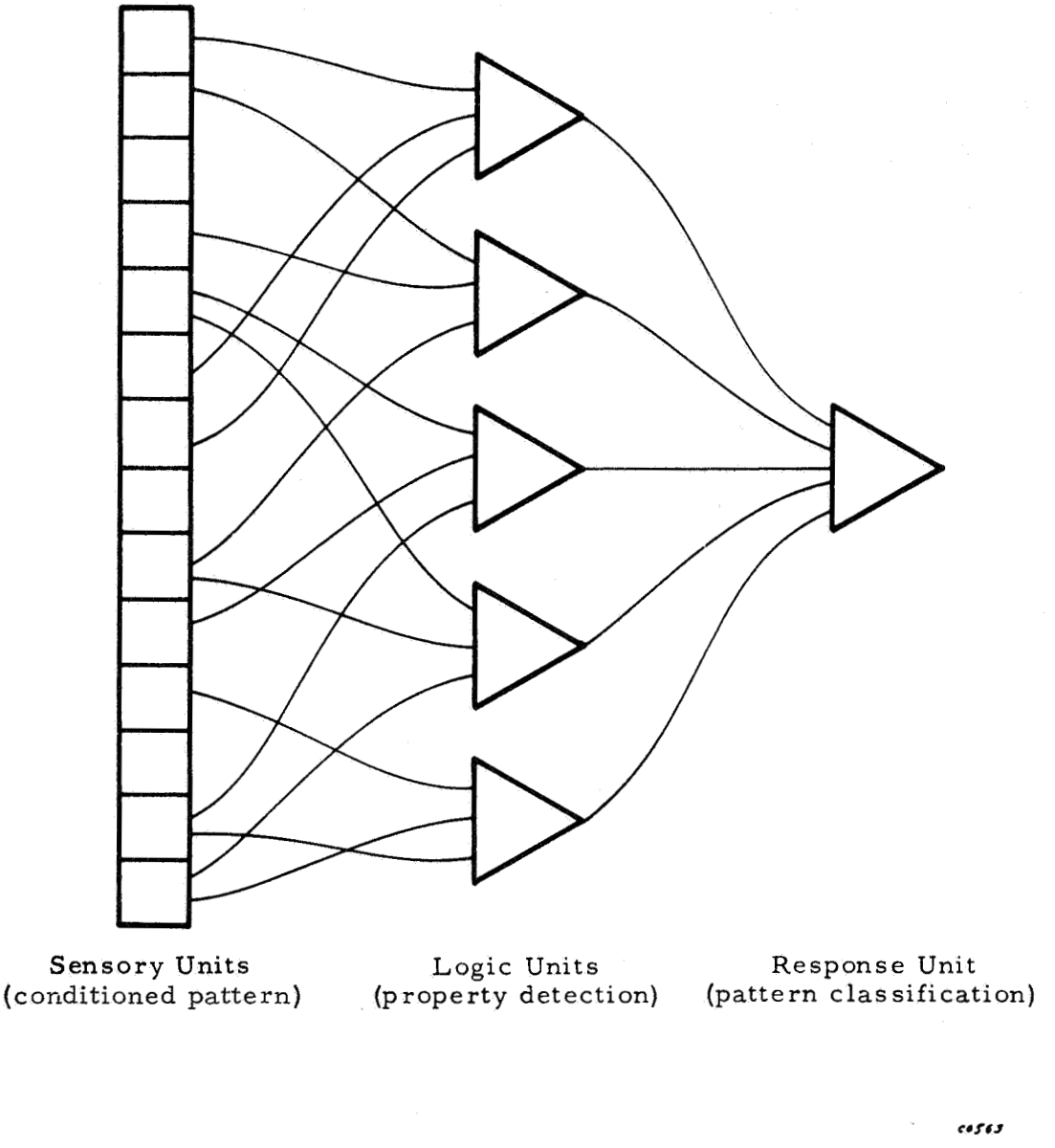


Figure 2. Decision Network Structure

property profiles generated a decision function was designed using each of the six techniques under consideration. The primary comparisons of Section 3.3.4.2 are based on the resulting 36 systems. For the two recognition tasks concerned with NIMBUS data, only the quadratic property filters were used. The comparisons of Section 3.3.4.3 are based on 12 system designs. A description of the recognition tasks is given in Section 3.3.3.

3.2 Generation of the Pattern Files

3.2.1 Origin of the Pattern Sets

Seven classes of patterns were used for the recognition experiments. Four classes were features of the lunar terrain—(1) craters with no conspicuous central elevations (2) craters with one or more conspicuous central elevations (3) rima and (4) wrinkle ridges.

Three classes were types of clous cover as seen from satellites—(1) noncumulus cloud cover (2) cumulus clouds—solid cells and (3) cumulus clouds—polygonal cells.

Sample patterns were derived from photographs. The lunar features were derived from Air Force Lunar Atlases,[155, 156] the original photographs having been taken at various astronomical observatories in the United States and France.

Mr. Eugene M. Darling, Jr., NASA Technical Officer for this program, selected from these sources 198 lunar features categorized as follows:

Feature	No. of Cases
Craters, No conspicuous central elevation	53
Craters, One or more conspicuous central elevation	53
Rima (rilles or trenches)	52
Wrinkle ridges	40

The sample cloud patterns were taken from a set of photographic reproductions of video cloud patterns transmitted from the NIMBUS I satellite. Mr. Darling examined numerous photographs and selected 323 examples of the three desired classifications from 311 picture frames. The cloud patterns are categorized in the following table.

CLOUD PATTERN CATEGORIES

Feature	Training Set (No. of Examples)	Generalization Set (No. of Examples)
Noncumulus	80	28
Cumulus, Polygonal Cells	80	16
Cumulus, Solid Cells	80	39

Mr. Darling also provided a subjective judgment of how well each frame depicts the pattern in question using this scale:

G, Good — These are the best cases showing the pattern in its most typical form.

F, Fair — The pattern shown is acceptable, but departs to a greater or lesser degree from its typical form.

P, Poor — The pattern is barely identifiable. These cases are of marginal use.

The distribution of patterns with respect to pattern quality was

PATTERN QUALITY

Pattern Quality	Pattern Class (Number of Cases)		
	Noncumulus	Cumulus Polygonal Cells	Cumulus Solid Cells
Good	53	59	65
Fair	47	37	50
Poor	8	0	4

In addition Mr. Darling describes the patterns as follows:

"The two cumulus patterns are characterized by cellular elements. Polygonal cells consist of a wall of cloud surrounding a clear area. In some cases this wall completely encloses the central clear region; in others the wall is open and the pattern becomes vermiculated (i. e., worm-like). For the purposes of this study both the open and the closed walls are lumped together into a single pattern called polygonal cells." (See Figure 3.)

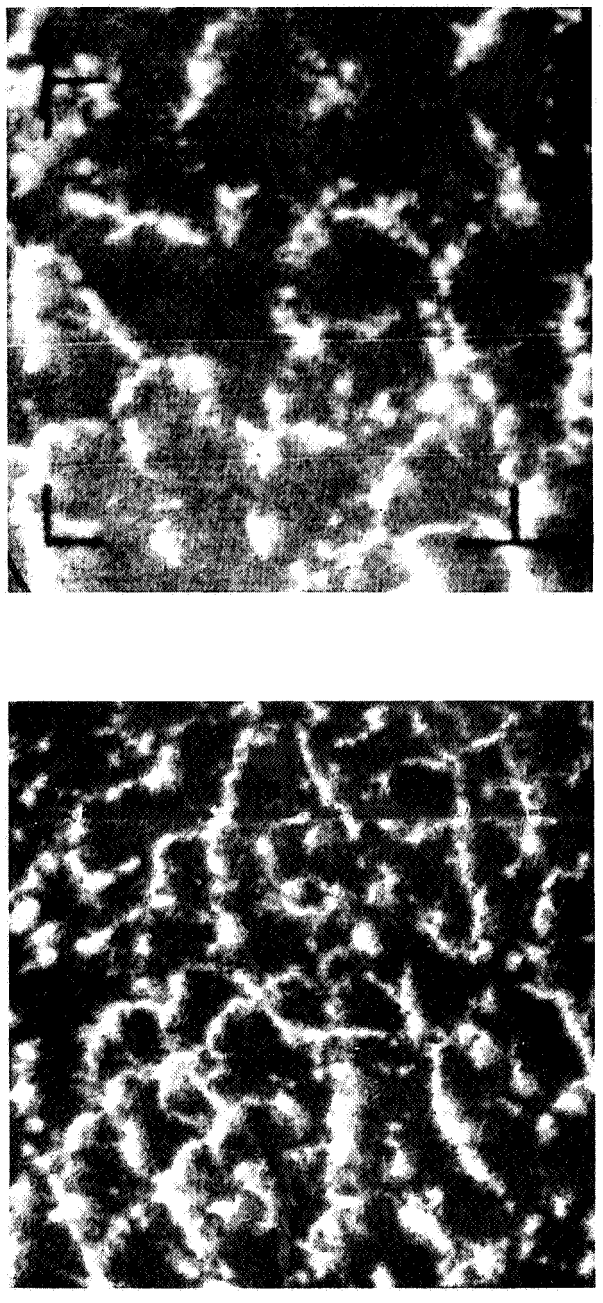


Figure 3. Cumulus, Polygonal Cells

"The solid cell case is (as the name implies) an unbroken mass of cloud, very bright, with sharp edges surrounded by a clear boundary. The solid cell may stand isolated from other clouds by a distinct expanse of clear air or may be embedded in a densely packed assemblage of cells where the boundary between cells is barely visible." (See Figure 4.)

"Noncumulus clouds appear typically in fibrous, diffuse sheets or filaments without distinct edges and without a cellular structure." (See Figure 5.)

"It is important to understand that a 'pattern' in the context of these experiments is not a single cell, but rather is an assemblage of many (or at least several) cells. The pattern must also be chosen to include a large fraction of cloud — at least 60% or so."

3.2.2 Pattern Normalizations

The goal of this study as was previously stated, was to investigate the feasibility of developing a system which could be placed aboard a spacecraft to perform an examination of the optical patterns received and relay only selective information identifying the features or objects encountered. The recognition system would be required to respond correctly over a wide range of pattern size variations, pattern translations, and rotations and to compensate for variations in picture brightness. System designs to account for these variations could possibly be achieved by including in the training set of patterns all possible combinations of size, translation, and rotation of the features recorded at varying brightness levels. However this would require a very large pattern set and would significantly increase the time and expense needed to record and process these patterns.

Limits were placed on the ranges of these variables. While these restrictions simplified the design of the recognition systems, they do not necessarily imply limited system performance. In an operating system, image brightness may be regulated by an iris, preferably controlled by knowledge of the sun's angle of incidence, or less desirably, by measured reflectance. The latter could be troublesome in cloud pictures. Translation and rotation limits were imposed only on the lunar pictures. Rotation control was primarily in terms of the direction of the sun, readily measured and controlled in the spacecraft. An expanded range in translation may be obtained by scanning, or by parallel systems using displaced input fields.

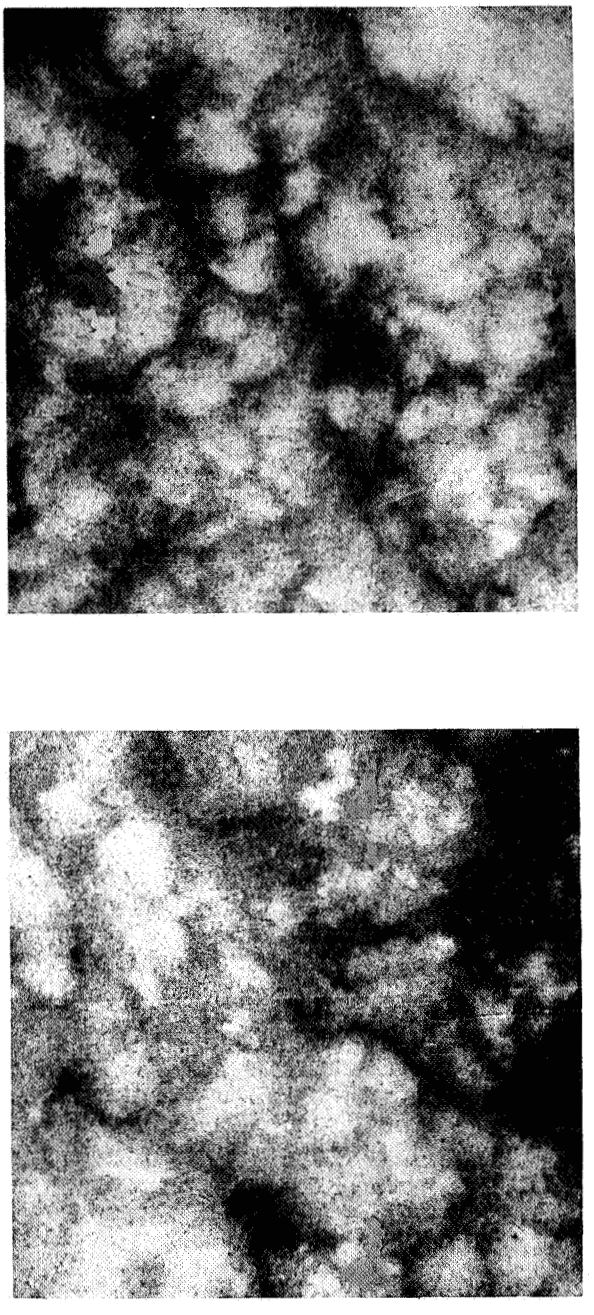
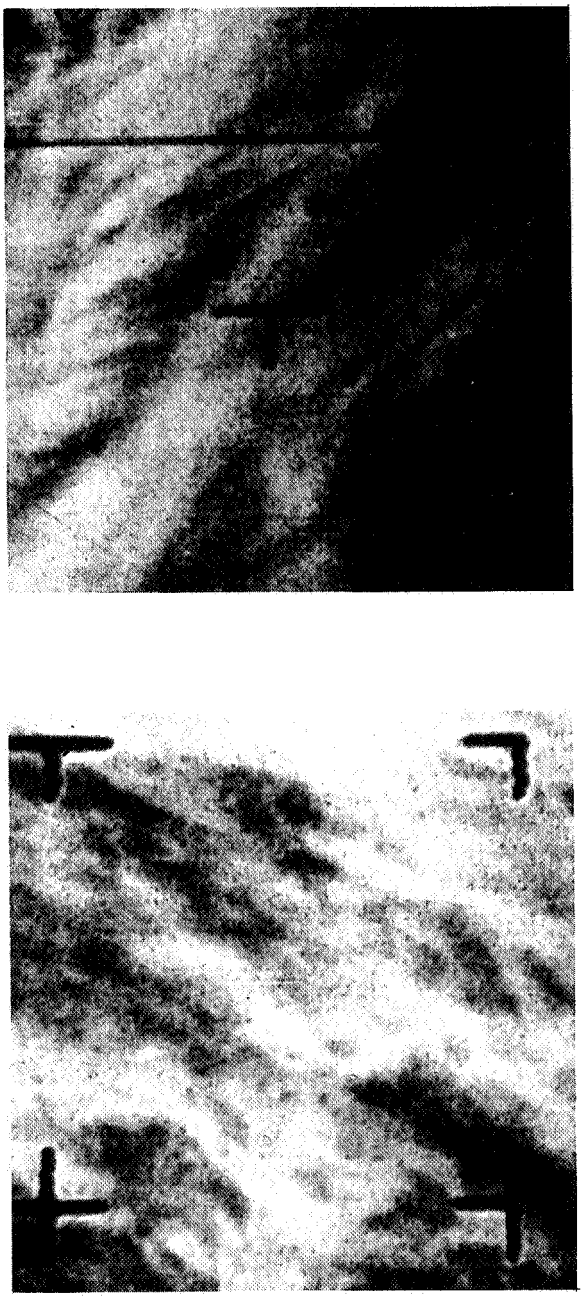


Figure 4. Cumulus, Solid Cell



C2011

Figure 5. Noncumulus

Similarly, size can be partly controlled optically, through knowledge of the spacecraft altitude, while additional flexibility can be added by techniques such as under scanning or by using different size apertures.

On the lunar patterns it was found that in most cases human observers could not distinguish ridges from rima without knowledge of the direction of illumination from the sun. For example, if the sun is illuminating the moon-scape from a relatively low angle, a ridge will be brightest on the side closest to the sun and cast a shadow on the side away from the sun. The opposite situation results for the rima or lunar canyon. While the craters could be recognized without prior knowledge of the sun's direction, the characteristic pattern of bright and dark crescents was tied directly to the direction of illumination. Orienting the camera or the scan aboard the spacecraft to bear a constant relative angle with respect to the sun in the plane of the lunar surface was considered technically feasible since such craft presently use sun sensors to position and stabilize themselves. This allows the designer of the recognition system to take advantage of a priori knowledge of the direction of the sun and greatly reduce the number of possible pattern variations.

The craters chosen for this study varied in diameter from 10 KM to 100 KM. In order to classify the smallest crater in the set with an aperture adequately large to encompass the largest crater, many different examples of craters of the smallest size would have been required at various positions in the receptive field, to make the pattern statistically significant with respect to the rest of the picture. In addition, the results of a resolution investigation indicated that such an aperture would have required 250 x 250 points to provide a sufficient number of resolvable elements to classify the smallest craters in the set. To accommodate these pattern size variations a simple size normalization of the crater patterns was performed. The diameter of the largest crater was specified to be no greater than 1-1/2 times the diameter of the smallest crater in the set. Despite the restrictions imposed by this normalization it was felt that the applicability of the resulting machine design to the general recognition problem would not be limited. A recognition system with connections specified by the experimental design could scan small sections of the received image to classify the smallest craters;

he same design, with connections expanded by a fixed ratio about the center point of the scan aperture, could be used to scan larger areas and classify he larger craters. Alternately a zoom lens as used in Surveyor I could perform the size normalization. The linear features presented a more difficult problem in size normalization. The ridges and rima did not have a convenient dimension comparable to the diameter of the craters. They varied considerably both in width and length. As a compromise, it was decided to normalize the size of the linear features so that the distance across the feature was approximately 10% to 15% of the width of the aperture used for he craters. This achieved sufficient resolution across the feature to determine the light side and shadow side of the ridge or rima but retained its inear characteristic. In most cases this meant using only part of the length of the linear pattern.

The assumption was made that the amount of light entering the vidicon aboard a spacecraft could be controlled electronically to maintain a constant average brightness for the received pictures. (The use of a controllable iris on the cameras of Surveyor I clearly illustrates this echnique). This assumption permitted adjustment of the brightness levels of the digitized photographs to compensate for variations in video gain inherent n the slow scan television system used for conversion of the pattern photographs.

To obtain an estimate of resolution required for recognition of the lunar patterns, typical examples were selected from the four pattern groups and placed before the slow scan television camera. The vertical sweep was adjusted to give 160 lines/inch on the TV monitor. The monitor display was photographed as the distance between the TV camera and pattern was increased in five steps. The farthest distance was chosen to give about 15 lines of resolution across a crater. Examination of the monitor photographs indicated that human observers could accurately distinguish between craters without and craters with conspicuous elevations with about 5 lines of resolution across the crater. The results for ridges and rima were not so well defined. It was difficult to measure the width of the linear features on the face of the monitor precisely. The width at the limits of recognition was on the order of 1/16 inch to 1/32 inch. As a result it was felt that at least 5 to 8 resolvable elements across the ridge or rima were

necessary to determine the bright side and the dark side of the feature, and that a 50 by 50 raster would be adequate for the recognition system.

A similar analysis of the NIMBUS data was performed. The prints supplied were enlarged from 35mm negatives on 8 in. x 10 in. contrast grade 3 paper. The correct gray scales were maintained during the printing process through the use of a densitometer positioned under a particula square in the middle of the gray scale wedge associated with each photograph. On most of the prints from six to eight gray levels were discernable on this gray wedge.

To obtain an estimate of resolution and aperture size required for recognition of the cloud patterns, typical examples were selected from the three pattern groups and placed before the slow scan television camera. The monitor display was photographed as the distance between the TV camera and the pattern was increased in several steps. The photographs were examined to determine the minimum size aperture with which the various patterns could be recognized. These apertures corresponded to a 0.9-inch square on the original 6 by 6 inch photographs of noncumulus patterns and a 1.4 inch square on the original polygonal cell and solid cell cumulus photographs. It was also felt that the 50 resolvable elements across the aperture did not give sufficient detail to recognize individual cells in the cumulus patterns. The digital pattern processing program was modified to achieve a resolution of 75 elements across the aperture (field of view of the recognition system). The small size of the aperture made it possible to choose several examples of the cloud features from each photograph. Thus, of the 311 prints originally provided, only the 147 prints judged by Astropower to contain the best examples of the selected features were used for conversion to video patterns.

The only other assumption introduced in the course of processing the NIMBUS patterns was concerned with the control of the amount of light entering the vidicon as was done for the lunar patterns. Again this assumption permitted adjustment of the brightness levels of the digitized photographs to compensate for variation in video gain inherent in the slow scan television system. There appeared to be no preferred rotational

orientation of the NIMBUS photographs as was the case for the lunar features and their characteristic shadow patterns. No attempt was made to preserve a particular angular relationship with the sun and no size normalization of the patterns was performed.

3.2.3 Photographic to Digital Conversion

The procedure for converting the photographic pattern sets to pattern files stored on digital magnetic tapes in a form suitable for use in the simulation experiments included the following steps:

Developing a video image of the photograph with a slow scan television system.

Recording the video output of the TV system on analog magnetic tape.

Replaying the recorded video signal through analog to digital conversion equipment creating a set of digitized video tapes.

Processing the digitized tapes to obtain correct resolution, enhance contrast, and eliminate "between picture" noise.

Editing the processed pictures to separate training and independent (generalization) samples and to provide translated versions of the recorded pictures.

The video images of the pattern photographs were obtained with a slow scan television system, Model 6030B, manufactured by General Electro-dynamics Corporation of Garland, Texas. The slow scan camera was mounted vertically on a stable structure with the lens pointing down toward the center of a 4 foot by 4 foot table. The table was easily moved up and down providing a practical range from 3 inches to 50 inches between lens and photograph. A suitable set of close up lens attachments provided an image of satisfactory clarity over this entire range to enable increasing or reducing its size with respect to the photographic pattern. A photo flood light was mounted about 2 inches from the camera on each side to eliminate shadows cast by the camera and supporting structure. The intensity of the two lights was controlled by a Variac. The vidicon tube used in the slow scan camera integrated the incident light to produce an image requiring a semidarkened room for the recording process.

A certain amount of operating experience was necessary to determine the best combinations of light intensity and exposure time for producing sharp images without saturating the vidicon. These combinations would change with distance between the lens and the photograph and, in some cases, would change with the gray scale level and texture of the photograph. A Simmons Omega repeating timer was used to control the exposure interval.

For the lunar data, the results of the resolution study indicated that 25 lines of resolution should be adequate to allow classification of the craters. It was decided to make the largest crater no larger than 80% of the width of the aperture to allow room for various translations. Since the smallest craters were to be no smaller than two thirds the size of the largest craters, a 50 x 50 point picture gave at least 25 lines of resolution across the smallest crater. Squares delimiting the smallest and largest crater sizes were drawn with grease pencil on the face of the TV monitor. The size of the chosen aperture was 1.5 inches on the face of the monitor.

The pictures were recorded with four times the final 50 x 50 resolution giving an initial aperture of 200 x 200 points. In the processing program described later in this section the resolution was reduced by a factor of four by summing the gray scale values of four adjacent points on a line for four lines and dividing by 16 to obtain a numerical average of the 16 points. This process yielded the desired 50 x 50 resolution across the aperture and alleviated the effects of noise added to the video signal in the recording process. It also tended to compensate for slight misalignments of individual scan lines.

The picture on the monitor scope was 2.47 inches high. In order to obtain 200 lines across the 1.5 inch aperture, 330 lines were required for the entire picture. The aspect ratio or ratio of width to height was 1.10 indicating that 363 sample points should be taken on each line to give an equivalent 200 lines across the 1.5 inch aperture in the horizontal direction.

The Precision Instruments' 2100 recorder was used to record the video output of the TV camera. The 30-inch/sec speed with its frequency response from dc to 10 KC yielded the best compromise between video scan time (the longer the scan, the worse the picture) and the number of

pictures (approximately 100) stored on each reel of analog magnetic tape. With careful calibration of the recorder electronics it was possible to achieve the specified signal to noise ratio of 46 db. However, it is evident that the RMS power of the gray scale information carried on the video pulse was much smaller than the dc power of the video pulse itself; therefore, the amplitude of the noise generated in the recording process was in the same range as the gray scale signal. Averaging 16 sample points to obtain one point at reduced resolution virtually eliminated the effects of this noise.

The recording process was straightforward. The photographic pattern was placed on the table under the camera at a distance which would place the feature within the limits of the aperture marked on the monitor. Craters were centered between the minimum size and maximum size limits. The photographs were oriented so the sun angle was from a specific direction. The lights were turned on for the proper exposure time, the recorder was turned on and allowed to stabilize at 30 ips, and one TV scan was initiated. When the scan was complete the recorder was turned off. A polaroid photograph was made of the TV monitor to retain a record of the actual area of the photograph scanned and the probable quality of the picture. Each photographic pattern was recorded in three rotations differing by 15 degrees.

When approximately three analog reels were filled, they were played back and processed with an analog to digital converter at the Douglas data processing facility in Huntington Beach. The sampling frequency required when digitizing the patterns was estimated as follows:

$$\text{Desired visible horizontal trace time} = \frac{\text{Number of horizontal samples}}{2 \times (\text{Video bandwidth})}$$

$$= \frac{363 \text{ Samples}}{(2)\ (10^4 \text{ cps})} = 18.1 \text{ ms}$$

The actual video bandwidth was 1 MC but the 10 KC filter on the input of the FM recorder gave an effective video bandwidth of 10 KC. The flyback time was measured at 1.1 ms so the total horizontal sweep time was 19.2 ms.

$$\text{Total frame time} = (\text{Number of lines})\ (\text{Total horizontal sweep time})$$

$$= (330 \text{ lines})\ (19.2 \text{ ms}) = 6.33 \text{ sec}$$

Allowance for 7% blanking and 75% registration probability normally considered in this calculation was not included. Tests at the beginning of the study indicated blanking was negligible. The registration problem was virtually eliminated by the average of four points in both directions since the minimum spatial frequencies appearing in the final picture would cover the width of eight lines per cycle instead of two.

The sampling frequency is then:

$$\text{Sampling frequency} = \frac{\text{Number of samples per line}}{\text{Visible horizontal trace time}}$$

$$= \frac{362 \text{ Samples}}{18.1 \text{ ms}} = 20 \text{ KC}$$

The tape speed was 15 ips and the sampling frequency was 10,000 samples/sec. Since the tapes were being played at half speed this gave an effective sampling rate of the required 20,000 samples/sec. The recorded video signal was quantized to 10 bits from the lowest to the highest signal level, obviously more than adequate to represent the eight gray levels. The A/D converter was turned on at the beginning of a set of pictures and turned off after 18 pictures had been processed and the output had been recorded on a digital magnetic tape. No attempt was made to eliminate the "between pictures" interval during the digitizing process.

The resulting magnetic tapes with the digitized video data were processed on Astropower's SDS 930 computer. The processing program read the tapes and separated the pictures from the "between pictures" interval. At the same time, it performed resolution reduction by obtaining the numerical average of blocks of 16 points as described above.

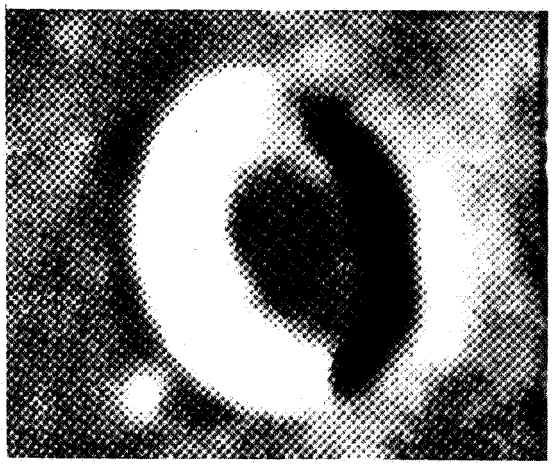
The program also performed a gray scale expansion about the features of interest. The gray scale range was determined by considering the highest video level in the pictures as white and the lowest level as black. The intervening range was divided equally between the six remaining gray levels. The floating gray scale was needed to compensate for the constant adjustments of the video gain required to maintain satisfactory performance of the TV system during the recording phase. It had the additional effect of enhancing the picture contrast in the regions of interest, serving as a mild form of signal conditioning.

The result was an 85 x 72 matrix of numbers from 0 to 7 representing the brightness level at each point on the photograph. In order to interpret the processed pictures more easily a character printout was used in addition to printing the numerical gray scale values. This consisted of assigning dense characters such as B's to represent dark points and letting blanks or periods represent light points. Intermediate points are represented by appropriate combinations of characters. The character printout was used for the reduced resolution examples shown in Figures 6 through 10.

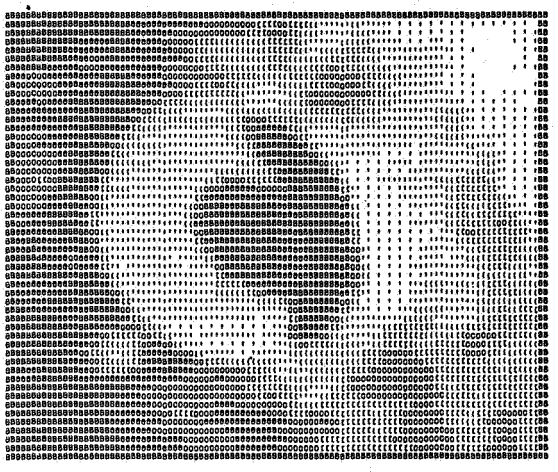
The final step in preparing the patterns was the editing process. Each of the basic patterns was initially recorded in three rotations differing by about 15 degrees, providing from 120 to 150 digital pictures for each set. The printouts were studied and the quality of each pattern was rated. The editing process consisted of choosing a 50 x 50 set of points from the larger 85 x 72 picture so that each feature was fairly well centered within its 50 x 50 frame. In addition, several translations were chosen which effectively shifted the feature within its frame up to 17 per cent of the width of field in four directions. In the case of the craters, nine translations were taken from each digitized picture. The linear features proved to have a wider variation in pattern quality, so up to 16 translations were taken from the better patterns. The editing resulted in a set of 1000 training and 200 independent test patterns for each of the four basic pattern sets.

Figures 6 through 9 show examples of each of the four basic patterns. The top two illustrations show the original photograph of a good example of the feature and its 50 x 50 character printout. The lower illustrations show other typical examples of the same feature. The illustrations underscore the fact that the distinguishing feature between the two types of crater patterns (the conspicuous central elevation) occupies only about 1 percent of the area of the picture. The distinguishing features between rima and ridges (the light and dark sides of the pattern) occupied from 5 percent to 10 percent of the picture area.

Figure 10 illustrates that the 50 x 50 point field preserves a surprising amount of detail found in the original picture. However,

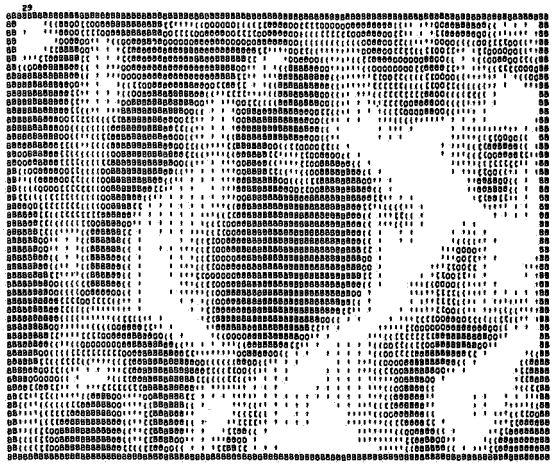


Original Photograph



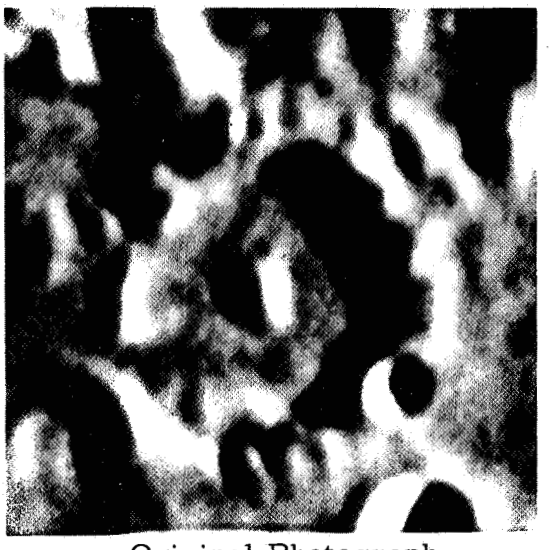
Final Resolution

Good Crater Prototypes

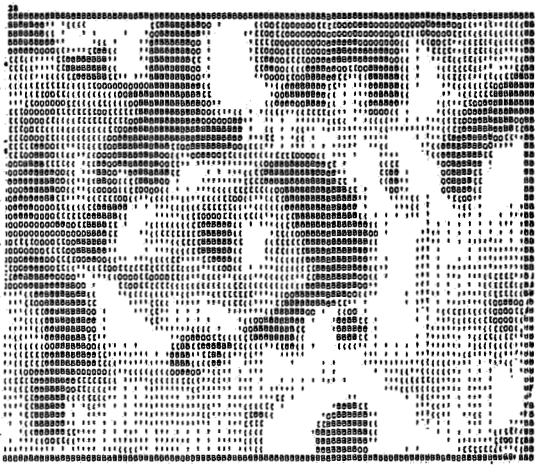


Typical Crater Patterns

Figure 6. Lunar Craters With No Conspicuous Central Elevation

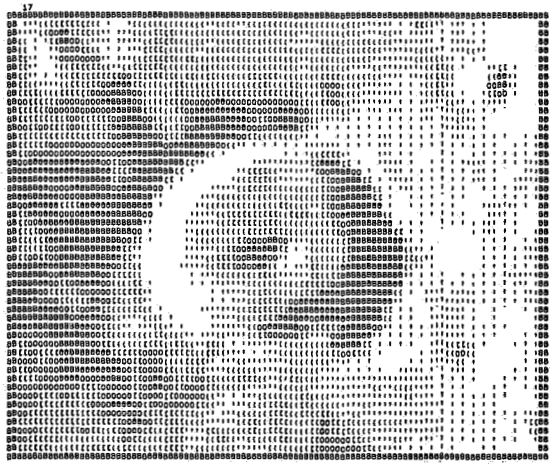


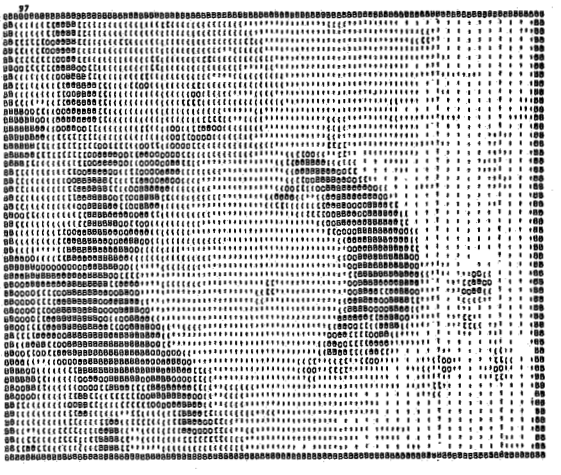
Original Photograph



Final Resolution

Good Prototype



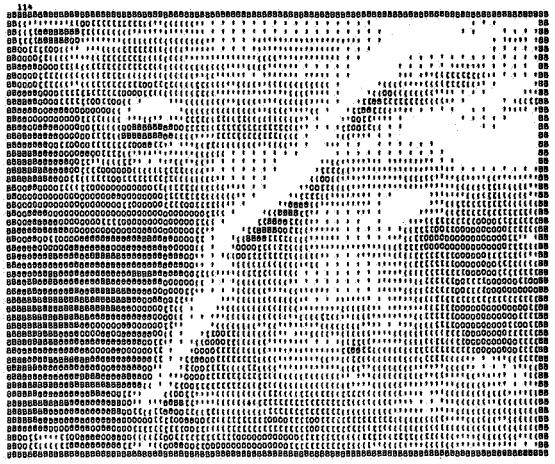


Typical Patterns

c3391

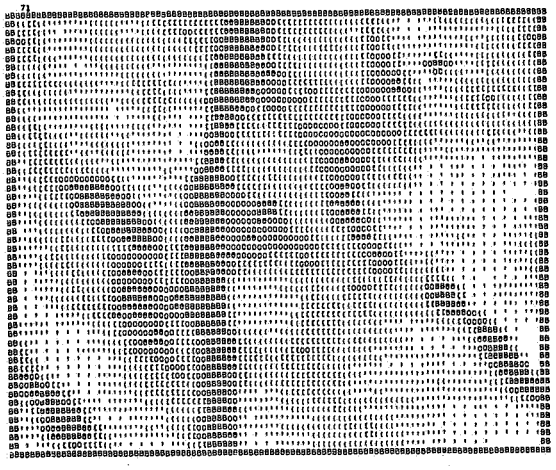
Figure 7. Lunar Craters With One or More Conspicuous Central Elevations

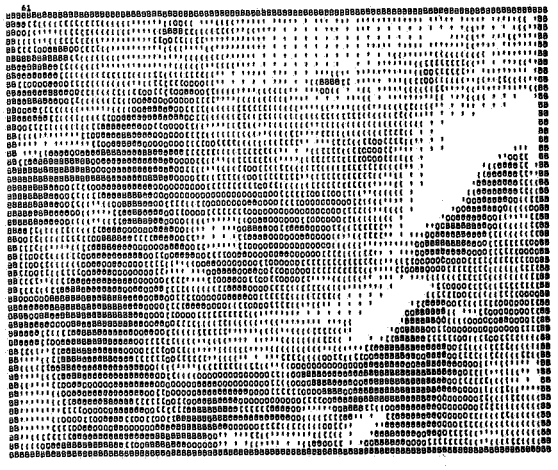
Original Photograph



Final Resolution

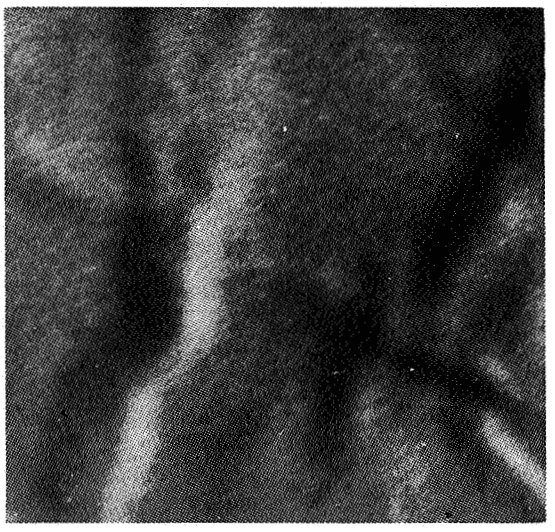
Good Rima Prototype

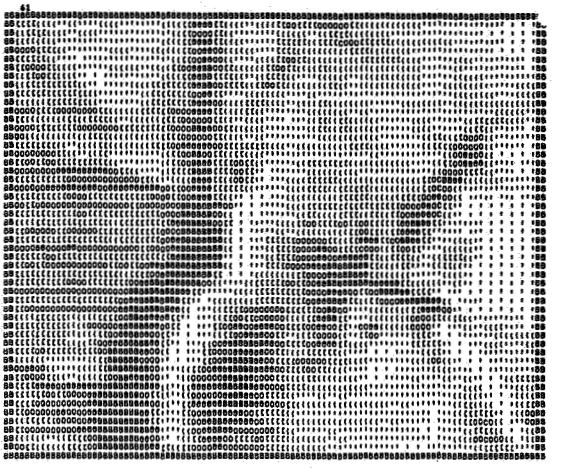




Typical Rima Patterns

Figure 8. Lunar Rima

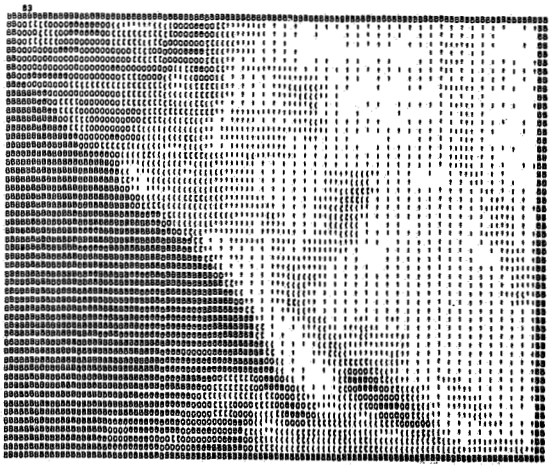


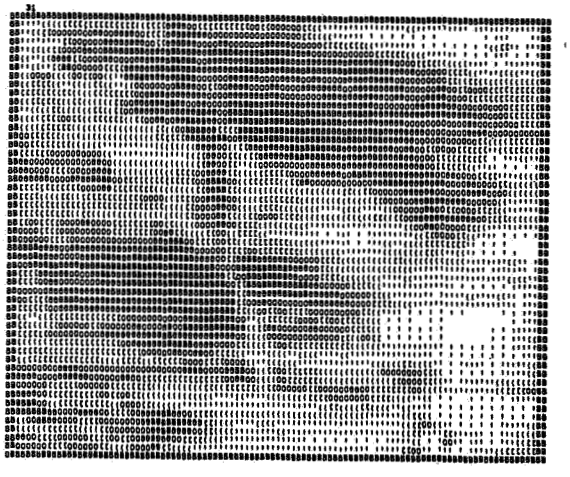


Original Photograph

Final Resolution

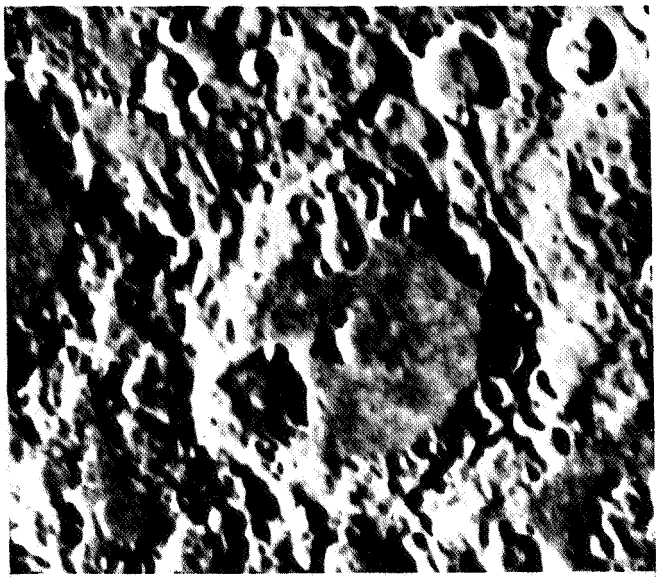
Good Ridge Prototype



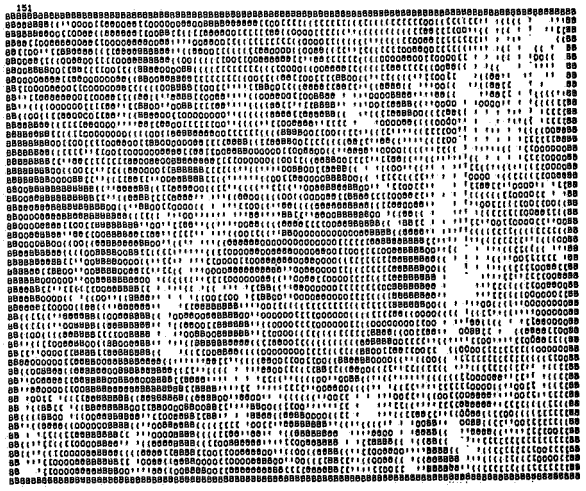


Typical Ridge Patterns

Figure 9. Lunar Ridges



Original Photograph



Final Resolution

Figure 10. Crater with Central Elevation (Albategnius)

it is still difficult to determine from inspection of the reduced resolution picture that the crater does have a central elevation.

The NIMBUS photographs were processed in a similar fashion. The sampling apertures on the original photographs were 0.9 inches square for the noncumulus cloud patterns and 1.4 inches square for the cumulus cloud patterns. With the smaller aperture used for the noncumulus photographs it was possible to avoid areas of the NIMBUS pictures containing fiducial marks; however, the larger aperture required for the cumulus patterns included at least one fiducial mark at any position on the photograph. In addition many of the photographs contained long black lines running across the areas of interest. Since these extraneous marks appeared as strong features and could confuse the classification problem, it was considered desirable to eliminate them. A water color retouching kit was used to cover the marks and an attempt was made to match the gray scale and pattern shape of the area surrounding each mark. The upper photograph in Figure 3 contains three unretouched fiducial marks and one retouched mark. The lower photograph contains two retouched fiducials and a retouched black line. Figure 5 contains examples of unretouched fiducial marks and black lines.

The video output of the TV system was recorded to allow batch analog-to-digital conversion of the video patterns. After analog-to-digital conversion at a sampling rate consistent with the resolution desired, the digital tapes produced were processed by a computer program which isolated the pictures and adjusted gray scale levels to compensate for changes in video gain in the TV system. This program also reduced the resolution of the pictures and stored the patterns in a standardized format on digital magnetic tapes. The resolution was reduced by summing the gray scale values of four adjacent points for three rows and dividing by 12 to obtain an average gray scale to represent the area originally covered by the 12 points.

An editing program was also employed to provide several translations and a 180° rotation of the stored patterns to expand the pattern files. The dimensions of the video patterns actually recorded were about 35 percent greater than the required aperture size. In editing the training set, six sample patterns were obtained from each recorded frame. The first aperture was centered in the frame, the second translation shifted the aperture

upward by 11 resolution points in the video frame, and the third translation shifted the aperture downward by 11 resolution points. The frame was then rotated 180° and the three translations were repeated. For the generalization set only the central position and 11 point upward shift were used in each rotation. The editing process produced a set of 1000 training and 200 independent test patterns for each of the three basic pattern sets.

3.2.4 Pattern Files

After the processing described above was completed, there were two pattern files for each of seven pattern classes:

(1) Craters with no central elevations
(2) Craters with central elevations
(3) Wrinkle ridges
(4) Rima
(5) Noncululus clouds
(6) Cumulus-solid cells
(7) Cumulus-polygonal cells

For each pattern class, one file, consisting of 1000 sample patterns, was used for designing recognition systems, and the other file, which contained 200 sample patterns, was used to test the recognition systems.

Design and test files were also established for three composite classes:

(8) Craters (including classes (1) and (2) above)
(9) Linear Features (including classes (3) and (4) above)
(10) Cumulus clouds (including classes (6) and (7) above)

The design and test files for each of these composite classes also consisted of 1000 and 200 sample patterns, respectively. Fewer translations were taken from each basic pattern in producing the files for the composite classes than for the single classes.

The 20 pattern files described provided the basic data for this program. Some 58 other pattern files were used in these studies, however.

Twenty-four of these files were used in the reduced aperture experiments of Section 3.3.4.4. For each class of lunar pattern (classes 1-4, 8, and 9 above), files of patterns using subapertures of 25 by 25 and 15 by 15 were generated. The subapertures were selected to contain the significant pattern features. Translations were included in these files, but their extent was controlled to assure that the significant features would be present.

Twenty-eight files represented the basic data in classes 1-7, including the reduced apertures. Computer generated translations and rotations were eliminated. These files were used for the augmentation experiments of Section 3.3.7.

Two of the files were the test files for classes 8 and 9 (craters and linear features), with the raster elements of each pattern subjected to a random rearrangement. These files were also used in the work described in Section 3.3.4.4.

The last two files were design patterns for classes 3 and 4 (wrinkle ridges and rima). The sample patterns in these files were generated in the computer, and represent idealized patterns. Systems designed using these artificial patterns were tested on real samples. This work is described in Section 3.3.4.4 as well.

3.3 Experimental Programs

3.3.1 Decision Functions

The six techniques for designing decision functions are called, in this report:

1. Forced Learning
2. Bayes Weights
3. Error Correction
4. Iterative Design
5. Mean Square Error
6. MADALINE

The first three of these are generally associated with the Perceptron program of F. Rosenblatt.[125] The last two are associated with the MADALINE program of B. Widrow.[157] Iterative design is a minimum loss approach, an approach first applied by W. Highleyman.[71] The particular algorithm used with its exponential form of the loss function, was developed at Astropower Laboratory.

The first five techniques derive linear functions of the binary property profile vectors for the computation of the decision. Let $b_i(j)$ be the output of the i-th property filter for the j-th pattern. Let

$$D(j) = \Sigma w_i b_i(j) - \theta$$

where θ is a threshold for the decision element. Then a binary decision is achieved by assigning the j-th pattern to the "positive" class if $D(j) > 0$ and to the "negative" class if $D(j) < 0$. If $D(j) = 0$, it is assumed, in this study, that the decision is in error regardless of the actual classification of the j-th pattern. For computational convenience, the two values of $b_i(j)$ are taken to be "1" and "0" in the first four techniques, and "1" and "-1" in the last two techniques. This assumption does not affect the capabilities of the techniques. The first five techniques differ in the method for assigning values to the parameters w_i and θ, based on the property profiles of the sample patterns.

3.3.1.1 Forced Learning

In the forced learning technique, the oldest of the six methods, the weights w_i are defined to be:

$$w_i = p_i - q_i$$

where p_i is the fraction of patterns in the "positive" class possessing the i-th property (i.e., the fraction for which $b_i(j) = 1$) and q_i is the fraction of patterns in the "negative" class possessing the property (again the fraction for which $b_i(j) = 1$). If the number of samples for each pattern class is equal, as in this study, the weights can be derived by examining the patterns one at a time. Let the number of samples per class be "m." Set each weight initially to zero. Then the patterns are examined one by one. The weights are modified according to the following rule.

$$\Delta w_i(j) = \frac{1}{m} \, sgn(j) \, b_i(j)$$

where sgn(j) is "1" or "-1" according to whether the j-th pattern is "positive" or "negative" and $\Delta w_i(j)$ is the increment to the w_i due to the j-th pattern. Only one pass through the sample patterns is required. Summary Figure 11 includes the forced learning technique.

The forced learning technique generally assumes the value of the threshold, θ, to be zero. In this study, it was taken to be the average value of

$$\Sigma w_i b_i(j)$$

for all sample patterns. Using this average value compensates for property filters which are assigned to be "on" primarily for patterns of one class and "off" primarily for patterns of the other class. The original forced learning technique was applied to randomly generated property filters which did not have this consistent bias.

3.3.1.2 Bayes Weights

The Bayes weights technique was originally derived as an "optimum" solution to the assignment of a decision function under the assumption that the property filters were statistically independent.

This assumption is not, in general, valid. With this technique, the weights are assigned by the following rule.

$$w_i = \ln \frac{p_i}{1 - p_i} - \ln \frac{q_i}{1 - q_i}$$

where the symbols are as described above.* These weights may be computed from the forced learning data if the accumulations of $\Delta w_i(j)$ are kept separately for "positive" and "negative" patterns. As with the forced learning, the threshold θ is taken as the average value of $\Sigma w_i b_i(j)$. The Bayes weights technique is summarized in Figure 11.

3.3.1.3 Error Correction

The error correction procedure is largely responsible for the current popularity of adaptive pattern recognition. The error correction theorem guaranteed perfect performance on the sample patterns, provided that a set of weights permitting this existed.

The weights are derived adaptively, in a manner similar to the adaptive derivation of the forced learning weights:

$$\Delta w_i(j) = \text{sgn}(j)\, b_i(j)$$

The difference is that this weight change is applied only when the j-th sample pattern is misclassified. If the sample pattern is correctly classified, the weights are not changed. This modification requires that each sample pattern be processed many times. The error correction theorem specifies that only a finite number of corrections are needed to achieve a solution, but does not specify the number. The patterns were processed cyclically in this study.

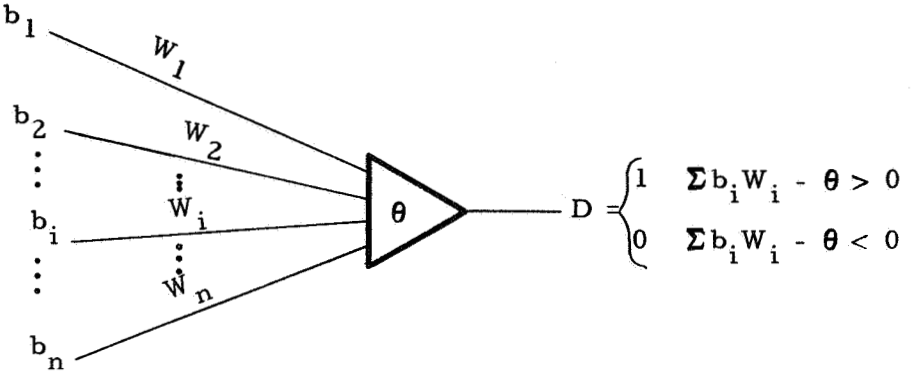
The threshold θ is also determined adaptively.

$$\Delta \theta(j) = -\text{sgn}(j)$$

Again, this change is made only when an incorrect decision is observed.

---

*To prevent the occurrence of undesirable infinite weights, a small constant is added to each of the four probability estimates p_i, $1 - p_i$, q_i, and $1 - q_i$. This may be justified by a Bayes argument in which an a priori Beta distribution is assumed (Reference 109).



Forced Learning

$$W_i = p_i - q_i$$

Sequentially $\Delta W_i(j) = \frac{1}{m} \operatorname{sgn}(j)\, b_i(j)$

Bayes Weights

$$W_i = \ln \frac{p_i}{1 - p_i} - \ln \frac{q_i}{1 - q_i}$$

Error Correction

Adaptively $\Delta W_i(j) = (\mathrm{err})\ \operatorname{sgn}(j)\, b_i(j)$

Mean Square Error

Adaptively $\Delta W_i(j) = \frac{1}{n+1} \left(1 - \operatorname{sgn}(j) \left(\sum_i b_i(j)\, W_i - \theta\right)\right) (b_i(j))$

C3429

Figure 11. Decision Function Generation

A modification of the original algorithm is incorporated to speed convergence. Whenever the length of the coefficient vector (the weights and the threshold) exceeds its previous maximum, all of the coefficients are doubled. This change was verified in a number of experimental runs, and is based on previous experiments and analytic considerations.

3.3.1.4 Mean Square Error

The mean square error technique differs from the error correction method in that, for design purposes, it is assumed that the desired value of

$$\Sigma w_i b_i(j) - \theta$$

is "1" for "positive" patterns and "-1" for "negative" patterns. Each time a pattern is processed, the weights and threshold are changed to give this desired state. This is facilitated by using the values of "1" and "-1" for $b_i(j)$.

$$\Delta w_i(j) = \frac{b_i(j)}{n+1} \left(\text{sgn}(j) - (\Sigma w_i b_i(j) - \theta)\right)$$

where n is the number of property filters. The threshold is derived similarly,

$$\Delta \theta(j) = -\frac{1}{n+1} \left(\text{sgn}(j) - (\Sigma w_i b_i(j) - \theta)\right)$$

Unlike error correction $\Delta w_i(j)$ can be positive or negative for a pattern of either class, and may change sign for one pattern processed at different times.

The patterns are processed cyclically. The technique is summarized in Figure 11.

3.3.1.5 Iterative Design

The iterative design technique is based on a minimum loss approach. Each sample pattern is assigned a loss number which depends on the value of the decision function.

$$L_j = \exp\left\{-\text{sgn}(j)\ (\Sigma w_i b_i(j) - \theta)\right\}$$

The loss for the j-th pattern, L_j, will be less than "1" if the pattern is correctly classified and greater than "1" if it is incorrectly classified. The

magnitude of the loss number reflects how definite is the decision. A system loss is defined as the sum of the individual pattern losses.

$$SL = \Sigma L_j$$

The weights and threshold are adjusted to minimize the system loss. Since this cannot be done directly, a relaxation process is used. The weights are adjusted cyclically. Each time a weight is changed, the threshold is also adjusted. These changes are defined by

$$\Delta\theta = \frac{1}{2} \ln \frac{\sum_j L_j(1 - b_i(j))\ (1 - \mathrm{sgn}(j))}{\sum_j L_j(1 - b_i(j))\ (1 + \mathrm{sgn}(j))}$$

and

$$\Delta w_i = \Delta\theta - \frac{1}{2} \ln \frac{\sum_j L_j b_i(j)\ (1 - \mathrm{sgn}(j))}{\sum_j L_j b_i(j)\ (1 + \mathrm{sgn}(j))}$$

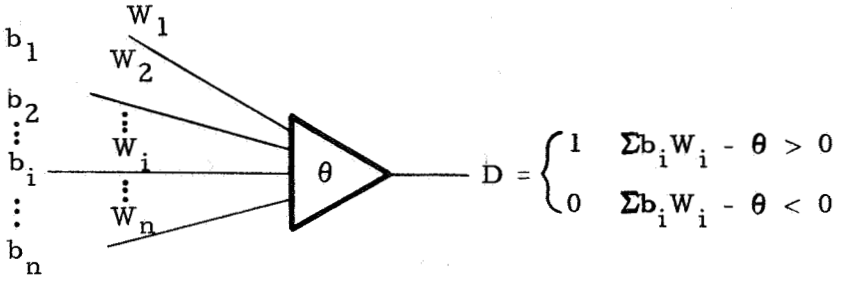
It has been shown that this process will give system losses converging to a minimum value for the system loss. In common with the error correction technique, perfect performance on the sample patterns will be achieved whenever this is possible.

To prevent the possibility of the finite samples yielding infinite weights (and hence absolute indicators) when not warranted, a small constant is added to each of the summations shown above.

The iterative design technique processes one property filter at a time, but needs data from all of the sample patterns. The computer requires "unit profiles" for the sample patterns, rather than property profiles. The technique is summarized in Figure 12.

3.3.1.6 MADALINE

The MADALINE technique is the only one considered capable of producing a nonlinear decision function. It does so at the expense of system complexity (see Figure 13).



Iterative Design

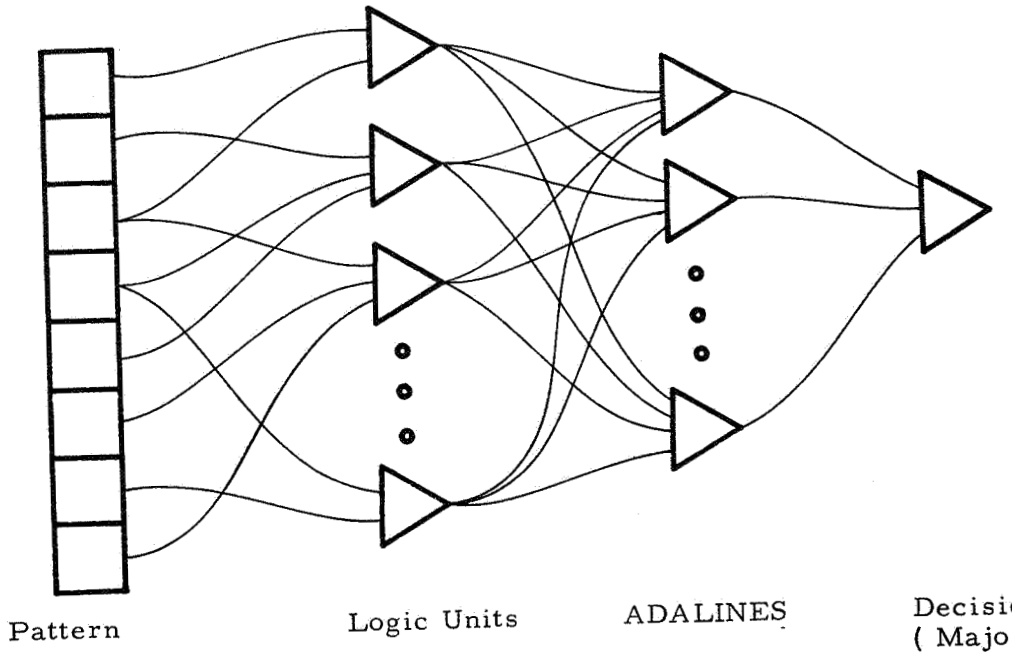
$$L_j = \exp\left\{\mathrm{sgn}(j)\,(\theta - \sum_i b_i(j)\, W_i\right\}$$

$$SL = \sum_j L_j$$

Adaptively $\Delta\theta = \frac{1}{2} \ln \frac{\sum_{neg} L_j}{\sum_{pos} L_j}$ Sums for $b_i(j) = 0$

$$\Delta W_i = \Delta\theta - \frac{1}{2} \ln \frac{\sum_{neg} L_j}{\sum_{pos} L_j}$$ Sums for $b_i(j) = 1$

Figure 12. Decision Function Generation



C1827

Figure 13. MADALINE Structure

The design technique has the added job of allocating the task among the subdecision elements, or ADALINES. The algorithm selected is an extension of the error correction technique.

The basic principle behind the algorithm is that of minimum change. As a sample pattern is processed, the first step is to determine if the decision is correct. If it is, no changes are made. If it is not, the number of ADALINES whose decisions must be changed to give a correct majority vote is noted. This number of ADALINES is selected from the (incorrect) majority on the basis of minimum magnitude of the linear subdecision functions. Each of the selected ADALINES is given a sufficient number of corrections, according to the error correction technique, to reverse its decision. It is to be noted that the minimum change principle is simplified by using "1" and "-1" values for $b_i(j)$.

The patterns are processed cyclically. Convergence to a solution, when it exists, is not guaranteed, and indeed, counterexamples to convergence have been found. Figure 14 summarizes the technique.

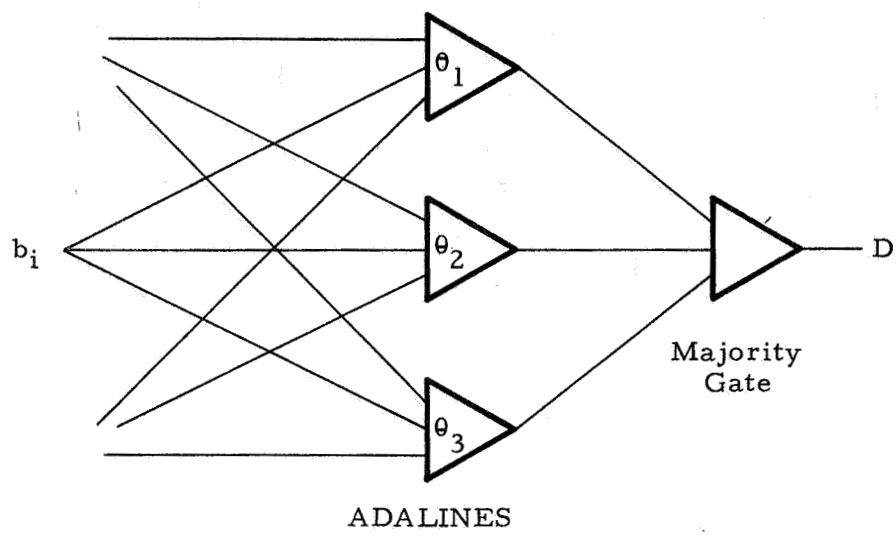
3.3.2 Statistical Property Filters

Two techniques for generating binary property filters, or logic units, are used in the initial experiments, one yielding units with linear switching surfaces and one yielding quadratic input units. The units have a limited set of randomly selected input connections.

Each logic unit thus views the patterns as projected on a subspace of the signal space. The switching surface for the unit is derived from the analysis of the amounts of the sample patterns as projected into the subspace. The analyses are optimum if the underlying pattern distributions are multivariate Gaussian. The linear surface arises as a special case when the covariance matrices for the pattern classes are assumed equal, and a multiple of the identity matrix.

3.3.2.1 Linear Units

For each of the recognition tasks, a set of 1000 property filters was derived. Each unit has 10 randomly selected input connections. Within the 10-dimensional subspace, the logic unit is defined by the vector equation:



C/1026

1. Test Decision
2. Determine Number of ADALINES to be Changed if Decision Incorrect
3. Choose those ADALINES Requiring Least Change

Figure 14. MADALINE Technique

$$X^T(M_1 - M_2) - \frac{1}{2}(M_1^T M_1 - M_2^T M_2) = 0$$

where M_1 and M_2 are the sample mean vectors of the "positive" and "negative" classes within the subspace. The switching surface consists of vectors X which satisfy this equation.

As shown in Figure 15, the switching surface consists of a hyperplane which is the perpendicular bisector of the line segment joining the means M_1 and M_2. The equation of the switching surface might also have been written:

$$\Sigma a_i x_i - \theta = 0$$

The coefficients a_i are given by

$$a_i = (m_1)_i - (m_2)_i$$

that is, the difference in the mean brightness for the two pattern classes at the appropriate sensory field point. It is to be noted that if the patterns were binary valued these coefficients would be identical to those derived from the property profiles by the forced learning technique described earlier.

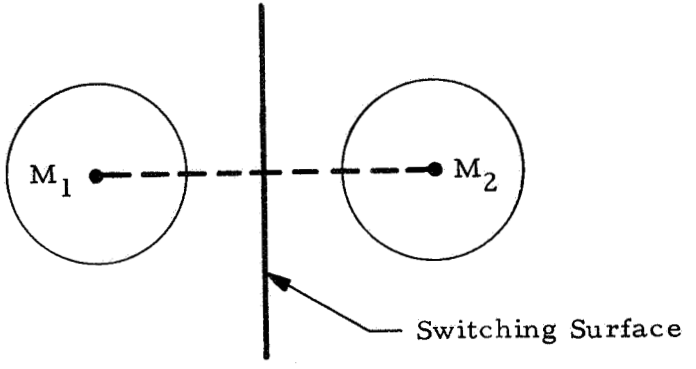
The advantage and the weakness of the technique are apparent. The coefficient for a connection does not depend on which other connections are selected for the unit, thus the units are very easily generated. However, the units are not designed based on contrasts and gradients in the patterns, and require a high degree of pattern centering, size normalization, and brightness and contrast control if the units are to be effective.

In the discussions of Section 3.3.4, the linear units are referred to by the code "SDA" for simple discriminant analysis.

3.3.2.2 Quadratic Units

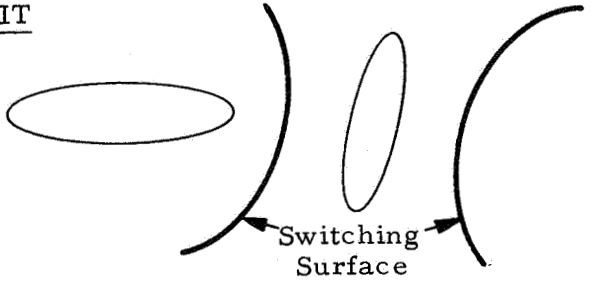
Two different computer programs were used to generate quadratic property filters in this study. One of these, written for an IBM 7094, was used in the initial experiments. This program is called "DAID" for discriminant analysis-iterative design. This program was

SIMPLE DISCRIMINANT ANALYSIS



$$X^T (M_1 - M_2) - \frac{1}{2} (M_1^T M_1 - M_2^T M_2) = 0$$

QUADRATIC UNIT



$$X^T(\Sigma_1^{-1} - \Sigma_2^{-1}) X - 2X^T(\Sigma_1^{-1}M_1 - \Sigma_2^{-1}M_2) + M_1^T\Sigma_1^{-1}M_1 - M_2^T\Sigma_2^{-1}M_2 + \ln\frac{|\Sigma_1|}{|\Sigma_2|} = 0$$

c/825

Figure 15. Property Filter Generation

modified for an SDS 930, and this modification is referred to as "QSID" (quadratic surface-iterative design). The QSID program was used in all experiments, subsequent to the initial ones, in which quadratic property filters were to be designed. The following describes the DAID program. The modifications contained in QSID are then listed.

The DAID program designs the set of property filters sequentially, ten property filters being added to the list in each pass. A decision function is defined concurrently, using the Iterative Design algorithm (Section 3.3.1.5). The loss function of the Iterative Design method is used to control the overall system design.

A selection process is used to obtain the ten filters added to the system in each pass. 100 seven-dimensional subspaces (of the signal space) are chosen by selecting their coordinates randomly. Within a randomly selected subspace, the switching surface of a unit is defined by

$$X^T(\Sigma_1^{-1} - \Sigma_2^{-1})X - 2X^T(\Sigma_1^{-1}M_1 - \Sigma_2^{-1}M_2) + M_1^T\Sigma_1^{-1}M_1 - M_2^T\Sigma_2^{-1}M_2 + \ln\frac{|\Sigma_1|}{|\Sigma_2|} = 0$$

The vectors M_1 and M_2 are the sample mean vectors and the matrices Σ_1 and Σ_2 are the sample covariance matrices of the "positive" and "negative" classes within the subspace. The switching surface is illustrated in Figure 15. The switching surfaces would be optimum if the M's and Σ's were actual means and covariance matrices, and if the underlying statistical distributions were multivariate Gaussian.

Since the equation of the switching surface does incorporate the covariance matrices, the resulting property filters do reflect average contrasts and gradients for pairs of sensory field points. The price for this is the necessity for matrix inversion in computer simulation, and the complexity of the quadratic input property filters in its hardware mechanization.

The 100 candidate property filters are then evaluated, and ten are selected. To ensure that the new filters are selected to complement those already in the system, the evaluation is based on the

system loss. Those candidates which, if added to the system, would provide the greatest decrease in the system loss are selected. During this evaluation, the output weights of property filters already selected are held fixed. A candidate property filter which performs well only on sample patterns already correctly classified, would have little effect on the system loss, and would not be selected.

Output weights are assigned to the ten selected candidates, the output weights of the 20 most recently selected property filters are updated, and the 10 filters are added to the system. Thus the output weight of each unit is adjusted twice after its initial assignment.

To save computer time, the DAID program folds this design process. Five blocks of subspaces and property filters are processed concurrently. In a single pass through the sample patterns, (1) mean vectors and covariance matrices are collected for 100 subspaces in Block A, (2) activities are determined and loss sums are collected for 100 candidate units in Block B and for 10 selected property filters in Blocks C, D and E, and (3) pattern losses are modified due to the changes in output weights for units in Blocks C, D, and E. After the sample patterns are processed, the property filters in Block E are placed in a permanent file, the output weights of the units in Blocks C and D are updated, and these property filters are transferred to Blocks D and E respectively. The candidate units in Block B are evaluated, the selected units are transferred to Block C and output weights assigned. Switching surfaces are computed for the subspaces in Block A, these units are transferred to Block B, and a new set of random subspaces generated for Block A. The next pass is then started.

Since the output weights of 30 property filters are adjusted at the same time rather severe instabilities result, due partly to the folding. If a correction is required and several units can make the change, each makes a full correction. This results in an overshoot much larger than the original change required. To control this, a constant is added to the loss sums. This results in undercorrection, with a much greater effect on large changes than on small ones. The constant was chosen to be sufficiently large to control the instabilities, but small enough to allow adequate weight variation

in three adjustments. A desirable side effect of this constant is the elimination of infinite weights. Based on a finite sample of patterns, a property filter should not be considered an absolute indicator.

A number of changes were required or desired in the adaptation of this program for the SDS 930. The QSID program uses six-dimensional subspaces. Fifteen candidate property filters are designed in each pass, and five are selected. The process is not folded and two cycles through the sample patterns are required in each pass. In the first cycle, mean vectors and covariance matrices are collected, in the second the activities of the property filters for the sample patterns are computed and stored. With the stored activities, it is possible to adjust the output weights one at a time without cycling the sample patterns. No instabilities can result, but a small constant was added to prevent infinite weights.

Due to the improved output weight adjustment procedure, property filter sets designed by QSID are substantially smaller than those designed by DAID. Competing systems were designed for both NIMBUS data tasks, and resulting performances were virtually identical. The QSID systems, however, required only 1/8 to 1/4 as many property filters as the DAID systems.

3.3.2.3 Recognition Tasks

Five binary recognition tasks are considered. Three of these are concerned with differentiating between features of the lunar terrain.

1. Task CVC: Separate craters with central elevations from craters without central elevations
2. Task RVR: Separate wrinkle ridges from rima
3. Task CVR: Separate the composite class of craters with and without elevations from the composite class of ridges and rima

Of the three tasks, separating the craters with central elevations from those without is the most difficult. Although the craters themselves are usually well defined, the central elevations are not. The elevations, which are the significant portion of the image in this task, cover less than one percent of the sensory field. Separating the ridges from

the rima represents a task of intermediate difficulty. The patterns in general are not well defined. Inspection of the computer printed patterns reveals a number of samples which are not readily classified by the observer. The significant portions of the patterns are larger than in the crater problem, covering perhaps five to fifteen percent of the sensory field. The task of separating the craters from the linear features is the easiest of the three, as the significant feature, the crater, is usually well defined and may constitute more than thirty percent of the image.

The remaining two tasks involved cloud textures in NIMBUS photographs.

4. Task PVS: Separate cumulus polygonal cells from cumulus solid cells
5. Task NVC: Separate noncumulus cloud cover from the composite class of cumulus polygonal and solid cells

This seems to be the most natural processing, although an examination of the patterns indicates that first splitting off the polygonal cells, and then separating the solid cells from the noncumulus cover would provide easier recognition tasks. Unlike the lunar patterns, the NIMBUS features cover the entire sampling aperture. Thus the lunar features can be centered in the field of view, but the NIMBUS patterns cannot.

For each class of each recognition task, a file of 1000 sample patterns was established. These patterns are used to design the property filters and the decision functions. The ability of the recognition systems to identify the class of these sample patterns correctly is referred to as "classification" capability.

A recognition system may register a high classification capability without being able to perform a useful function. A system requiring a complete match with one of the sample patterns would score 100% in classification capability, but would be useless on new patterns. The patterns contain a considerable amount of data, much of which is extraneous to the recognition task. With finite sets of sample patterns, some of the extraneous data is bound to exhibit a spurious correlation with the pattern classification. Systems designed on this noise would be of no value. In the

QSID and DAID designs it appears that the first half to two-thirds of the property filters required (for 100% classification) are relevant to the overall recognition task, and the remainder are more specific to selected patterns in the training set.

Ideally, the recognition systems would be evaluated using the underlying distributions in the signal space of the pattern classes. With the decision surface of the system known, the fraction of patterns of each class that would be classified correctly can be computed. Unfortunately, these underlying distributions, in general, are not known. In this study, these fractions are estimated from the system performance on an independent sample of patterns. A second file of 200 sample patterns for each class of each task was established for this purpose. The performance of a system on these patterns is called the "generalization" capability.

Considering the dimensionality of the signal space, a file of 200 test patterns for each class is not large. This is especially true since the 200 patterns represent translations and rotation of a still smaller set of basic patterns. Therefore, small differences in generalization capabilities should not be considered decisive. However, as the same test and design patterns are used for each system, it is felt that the comparisons are valid.

3.3.4 Initial Experiments

3.3.4.1 Introduction

Twelve combinations of design techniques were tested on three recognition tasks dealing with lunar topography. The experiments are summarized in Figure 16.

For each of the three lunar feature tasks, two sets of property filters were designed as discussed in Section 3.3.2. One set consisted of 1000 linear units, each having 10 input connections. This set is designated by the code "SDA." Three hours of computer time, on an SDS 930, were required to generate the units and to determine the property profiles of the sample patterns. The other set consisted of 310 quadratic units, each having 7 input connections. The quadratic units are designated by the code "DAID." Generation of the units and the corresponding property profiles required seven hours of computer time on an IBM 7094.

TASKS

1. Craters with Central Elevations vs. Craters Without
2. Rima vs. Wrinkle Ridges
3. Craters vs. Linear Features

PROPERTY FILTER GENERATION

1. Simple Discriminant Analysis (SDA)
2. Discriminant Analysis with Iterative Design (DAID)

DECISION FUNCTION GENERATION

1. Forced Learning
2. Bayes Weights
3. Error Correction
4. Iterative Design
5. Mean Square Error
6. MADALINE

c/824

Figure 16. Experimental Program (Lunar Features)

The decision functions designed by the programs which generate the property filters were discarded. For each of the three tasks, and for each of the two sets of property filters, six techniques were used to reassign decision functions. The six techniques were discussed in Section 3.3.1. All of these functions were determined on the SDS 930. The first two methods, forced learning and Bayes weights, were combined in a single program of 20 minutes. The remaining four methods are recursive. Each of these methods had its own computer program. Each program was allowed to run a minimum of eight hours (maximum 12 hours) for each decision function, the exact time depending on computer availability. Three exceptions occurred when the programs terminated upon reaching perfect performance. More cycles could be accomplished on the DAID units than on the SDA units, since there were fewer adjustments to be made in each cycle.

Six design techniques were tested on two recognition tasks dealing with cloud cover imagery. The experiments are summarized in Figure 17. Unlike the lunar feature patterns, the NIMBUS patterns cannot be centered in the sampling aperture. Unless the classes can be separated by the average gray level of the patterns, the linear property filters generated by the simple discriminant analysis (SDA) will give poor performance on uncentered patterns. In the experiments on the lunar features the linear units provided poorer performance than the quadratic units (DAID). Therefore, only the quadratic units were used in the NIMBUS experiments. For each of the two tasks, a set of 400 quadratic property detectors, each unit having 7 input connections, was derived.

Decision mechanisms were designed for the sets of property filters using the six adaptive techniques (Forced Learning, Bayes Weights, Error Correction, Iterative Design, MADALINE, and Mean Square Error). The last four are recursive routines and were modified to compute the classification and generalization performance levels after each cycle. The results of the NIMBUS experiments are presented in a somewhat different format than for the lunar experiments, reflecting the additional information provided by these program modifications.

The design runs for the decision mechanisms were longer than for the lunar experiments. The Iterative Design procedure

TASKS

1. Polygonal vs. Solid Cells
2. Noncumulus vs. Cumulus

PROPERTY FILTER GENERATION

1. Discriminant Analysis with Iterative Design

DECISION FUNCTION GENERATION

1. Forced Learning
2. Bayes Weights
3. Error Correction
4. Iterative Design
5. Mean Square Error
6. MADALINE

03394

Figure 17. Experimental Program (Cloud Features)

was run on an SDS 930 for 13 hours for the first task and 8 hours for the second. The MADALINE design was run for 9 and 16 hours, respectively. The MADALINE system used 3 ADALINES instead of the 7 used on the lunar tasks. This, combined with the longer run brought the MADALINE performance levels up to the best in the group. The Error Correction procedure was run for 10 and 19 hours and the Mean Square Error for 10 hours on each task.

3.3.4.2 Lunar Data

1. Craters vs Craters

The task of separating the craters with conspicuous central elevations from those without elevations, as previously discussed, is the most difficult of the three lunar data tasks.

Figure 18 presents the results achieved. A distinct bias in favor of the DAID units may be observed. With these units, MADALINE is able to achieve perfect classification performance, error correction and iterative design achieve an intermediate level (92-93%), and the remaining three techniques give the poorest performance (82-84%). Bayes weights gives the best generalization performance, closely followed by forced learning. Mean square error gives the poorest generalization performance, while the other techniques are approximately equal.

The generalization performance figures achieved on this difficult task are not satisfactory, so that the comparisons drawn from them should not be given much emphasis.

2. Ridges vs Rima

The separation of the rima from the wrinkle ridges is a somewhat easier task than Task CVC, but is more difficult than any of the other remaining three tasks.

Figure 19 summarizes the results achieved. These results again show a distinct advantage for the quadratic DAID units over the linear SDA units. With the DAID units, both error correction and iterative design are able to achieve complete separation of the sample patterns.

Technique	% Design Pats.		% Gen. Pats.	
	SDA	DAID	SDA	DAID
Forced L.	65.45	82.60	51.00	62.75
Bayes W.	65.60	84.10	51.00	63.75
Error Corr.	87.05	92.80	55.75	59.75
Iter. Des.	89.80	92.10	55.00	59.00
Mean Sq.Er.	57.35	81.80	52.00	53.00
Madaline	72.20	100.00	54.00	58.00

Figure 18. Craters With Central Elevations vs. Craters Without

Technique	% Design Pats.		% Gen. Pats.	
	SDA	DAID	SDA	DAID
Forced L.	70.40	81.20	72.25	71.0
Bayes W.	70.40	82.60	72.50	75.25
Error Corr.	90.25	100.00	65.50	70.25
Iter. Des.	92.10	100.00	64.00	73.75
Mean Sq.Er.	78.55	91.35	56.25	74.50
Madaline	80.30	95.80	67.50	75.75

Figure 19. Rima vs. Wrinkle Ridges

MADALINE, despite having seven times as much structure, did not achieve complete separation. It is felt that this is attributable to the inefficient algorithm which could not arrive at a suitable state within the allotted time. A longer design run, and/or the use of fewer ADALINES might have yielded perfect separation.

The highest generalization performances are obtained by MADALINE and Bayes weights using the DAID units, but the levels achieved by all systems do not appear to be significantly different. Only the Bayes weights and forced learning techniques approach this level using the SDA units.

An examination of the patterns shows a number of patterns for which the classification by an observer cannot be made accurately, and some patterns in which the ridge or rima is difficult to locate. In view of the quality of the patterns, generalization performance of 75 percent is more encouraging than the results for Task CVC, but it is not considered high enough for an operational system.

3. Craters vs Linear Features

The separation of the craters from the linear features (ridges and rima) is the easiest of the five tasks. Despite this, it is one of the two tasks for which none of the systems achieved perfect separation of the sample patterns.

The results obtained are presented in Figure 20. In most cases, the DAID units again proved to be more suitable than the SDA units. With the DAID units, iterative design and error correction give the best classification performance, yielding over 99 percent correct decisions. Even with the SDA units, these two techniques achieved higher classification performances than the other techniques do with either set of units. Generalization performance with the iterative design decision function equals the classification performance at 99.5 percent, considerably higher than the other techniques.

Again, the MADALINE performances were disappointing. To test to see if this was due to inefficiency of the

Technique	% Design Pats.		% Gen. Pats.	
	SDA	DAID	SDA	DAID
Forced L.	76.55	78.50	77.50	74.75
Bayes W.	76.55	82.60	77.50	77.50
Error Corr.	91.75	99.25	74.00	99.25
Iter. Des.	97.25	99.50	74.75	99.50
Mean Sq.Er.	91.20	83.25	73.00	82.25
Madaline	87.40	87.35	78.25	84.25

Figure 20. Craters vs. Linear Features

algorithm, an additional decision mechanism was designed for the DAID units. Three ADALINES were used, and 150 cycles were executed. The classification performance of this system was 94.35, and the generalization performance was 89.5. Both figures are noticeably higher than those in Figure 20.

It was suspected that the design point which achieved the highest generalization performance did not necessarily coincide with the point providing the highest classification performance for the iterative design technique. The program was modified to permit testing the generalization performance at the end of each cycle. The program was then run using the SDA units. The resulting classification and generalization error rates are presented on a cycle by cycle basis in Figure 21. It can be seen that if the design was terminated at the end of 7 cycles (instead of 26), a four percent decrease (from 26 to 22) in the generalization error rate is possible, at the expense of a 4.6 percent increase in the classification error rate (from 2.9 to 7.5). In subsequent experiments, these results were found to be typical. All four of the recursive programs were modified to permit this type of presentation for the experiments on NIMBUS data.

3.3.4.3 NIMBUS Data

1. Cumulus vs Noncumulus

The first task considered for the NIMBUS data is the separation of cumulus from noncumulus cloud cover. As in the lunar experiments, 1000 samples of each class are used as training patterns, and an independent sample of 200 patterns of each class is used to test generalization. These 1200 patterns are derived from a smaller set of basic patterns by translation and rotation. The patterns are presented as a 75 by 75 array, in contrast to the 50 by 50 array used in the lunar experiments.

Four hundred quadratic property filters were derived for this task using the DAID program. This number was selected primarily on the basis of computer time available. The iterative design algorithm is able to separate the training patterns completely, demonstrating that the property profiles of the patterns are linearly separable.

The performances achieved by the six adaptive techniques are shown in Figure 22. For the four recursive techniques,

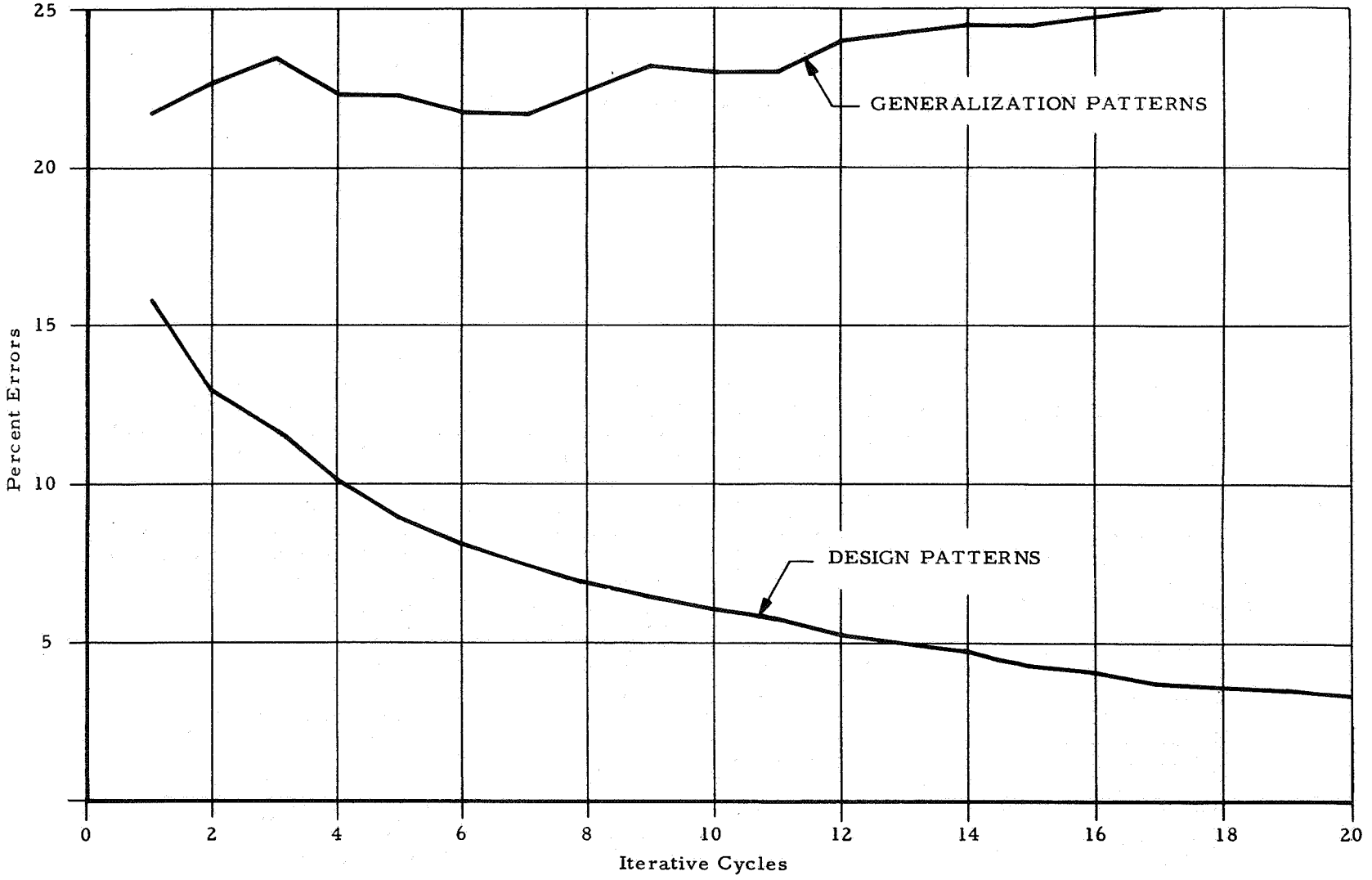


Figure 21. Craters vs. Linear Features, Iterative Design (1000 SDA Units)

Technique	Classification %	Generalization %
Forced Learning	78.00	81.25
Bayes Weights	77.90	82.50

	Total Cycles	For Best Classification			For Best Generalization		
		Classif. %	Gen. %	Cycles	Classif. %	Gen. %	Cycles
Error Correction	365	95.15	81.75	81	92.20	85.75	6
Iterative Design	78	100.00	84.25	78	93.45	86.25	2
Mean Square Error	48	83.70	78.50	1, 12	83.70	78.50	1, 12
Madaline	63	100.00	85.50	63	99.80	86.00	59

c1758

Figure 22. Cumulus vs. Noncumulus — 400 DAID Units

two blocks of data are shown. The first block shows the cycle in the recursive process at which the best performance on the training patterns is achieved, and the classification and generalization performance at that point. The second block indicates the cycle and performance levels for best generalization performance. The total number of cycles examined is also indicated.

Both the iterative design and MADALINE achieved perfect separation of the training patterns. Iterative design shows the patterns to be linearly separable. Despite a very long run (19 hours), error correction did not achieve complete separation.

The best generalization performances are achieved by iterative design, MADALINE, and error correction. The other recursive technique, mean square error fails to achieve even the levels of the nonrecursive forced learning and Bayes weights. Iterative Design and error correction achieve their best generalization performance very early in the design. This has been observed consistently, and may be characteristic of these two techniques.

Of the three high performance techniques, MADALINE shows the least sacrifice in generalization performance when classification performance is used as the criterion for terminating the design. This, of course, is a desirable feature. This observation may be related to the early peaking of generalization for iterative design and error correction. Iterative design shows a moderate drop-off, while the performance for error correction falls below the level achieved by Bayes weights.

Figures 23 through 26 show the performance of the four recursive techniques on a cycle by cycle basis (up to 100 cycles). The solid lines represent classification performance, the broken lines show generalization performance. Best performance points are indicated by small triangles under the curves. Although the error correction and MADALINE curves show more irregularity than the iterative design and mean square error curves, the degree of irregularity is not typical. The error correction and MADALINE curves of the next section show greater irregularity, and are more typical. The greater the irregularity, the more difficult it is to choose the best stopping point for the design process. Generally, the peaks

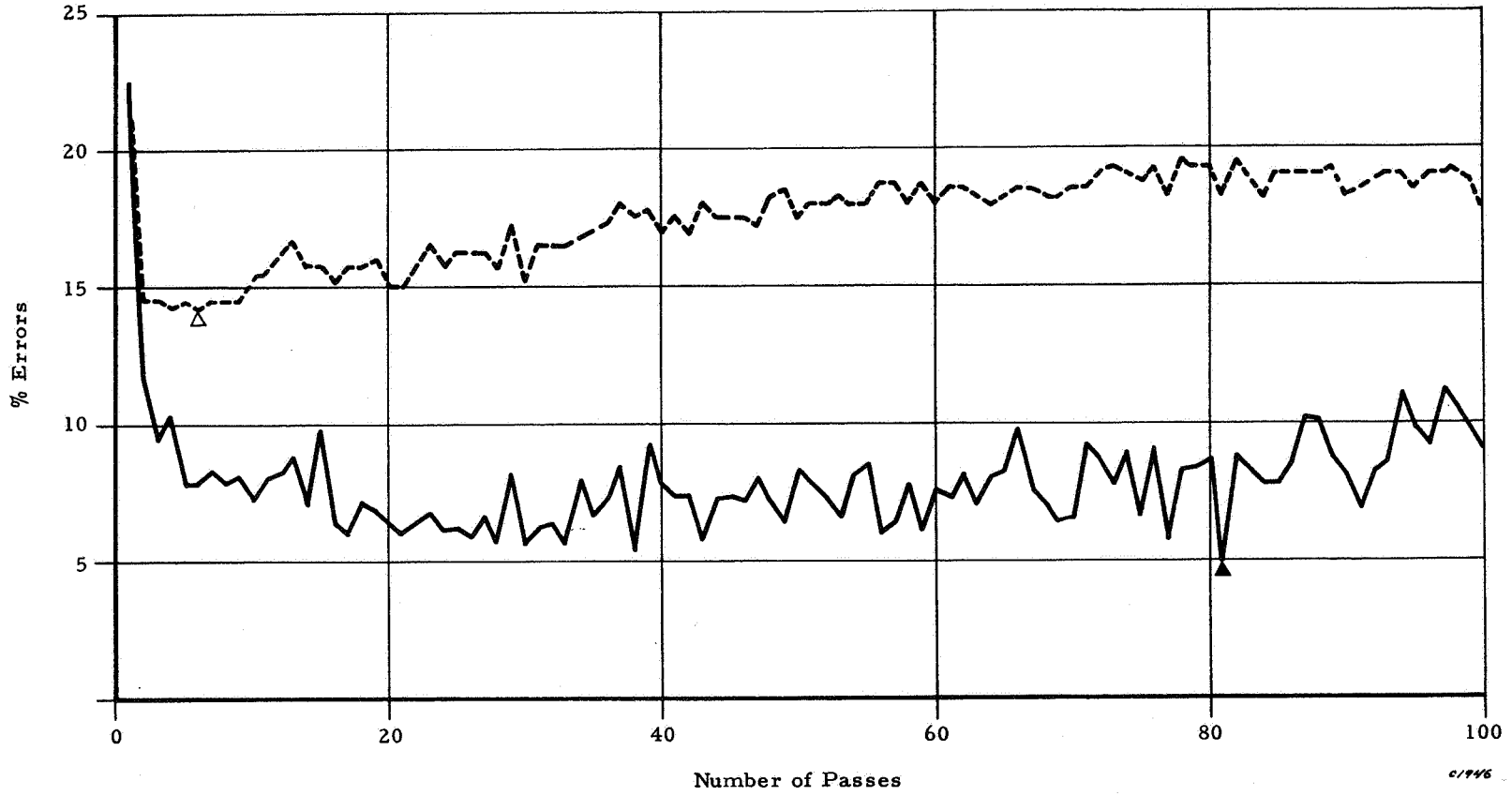


Figure 23. Error Correction — Cumulus vs. Noncumulus

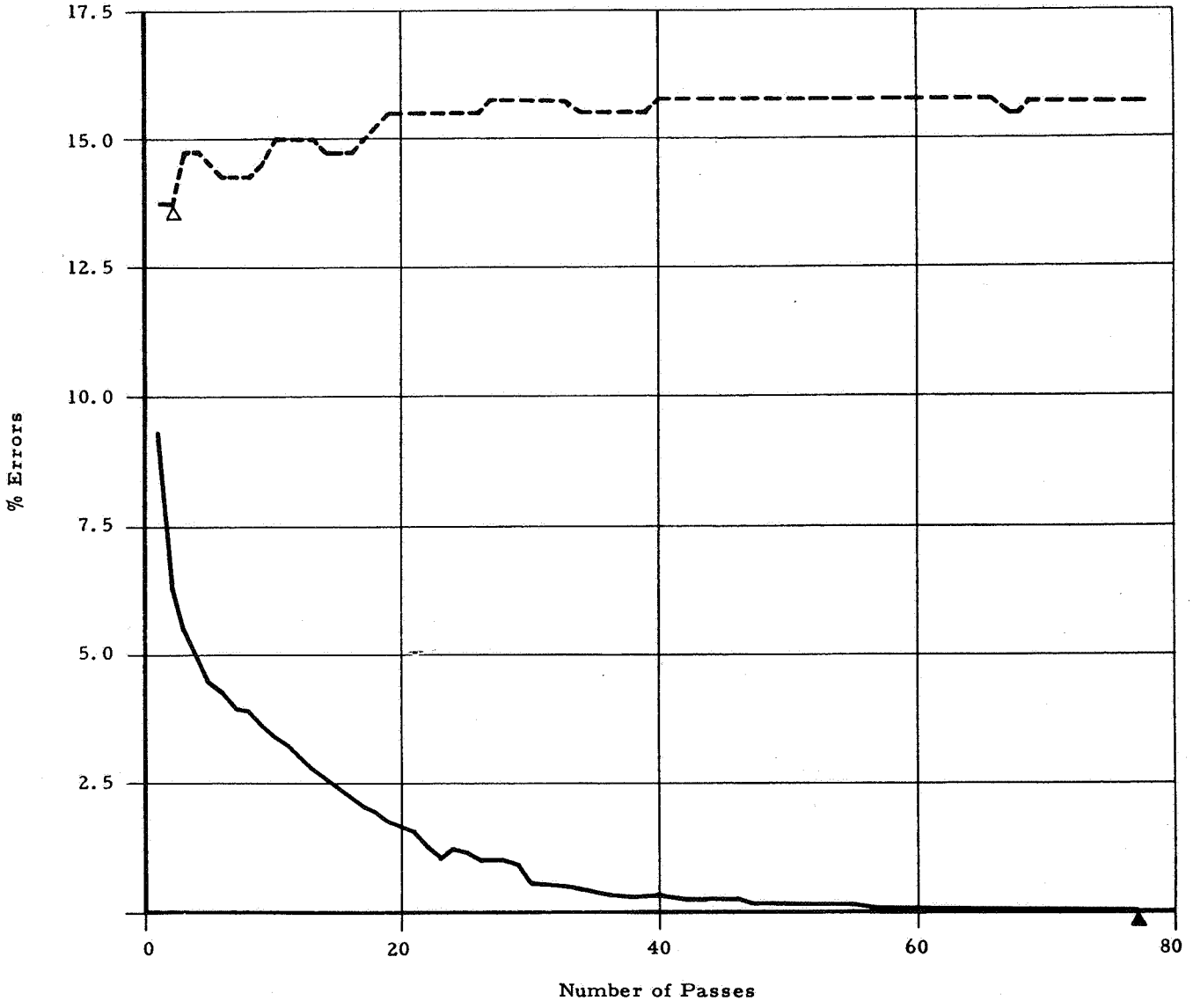


Figure 24. Iterative Design — Cumulus vs. Noncumulus

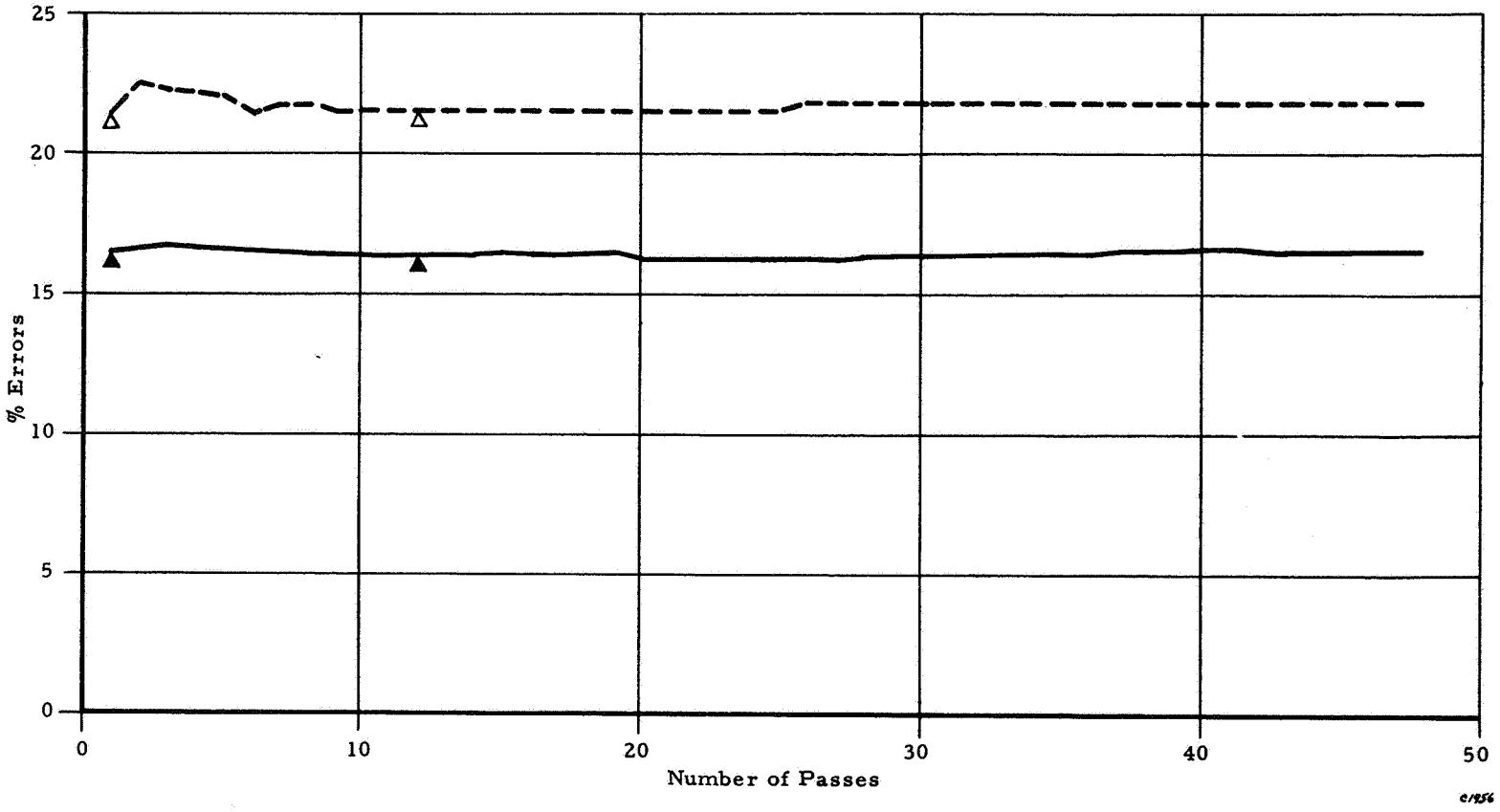


Figure 25. Mean Square Error – Cumulus vs. Noncumulus

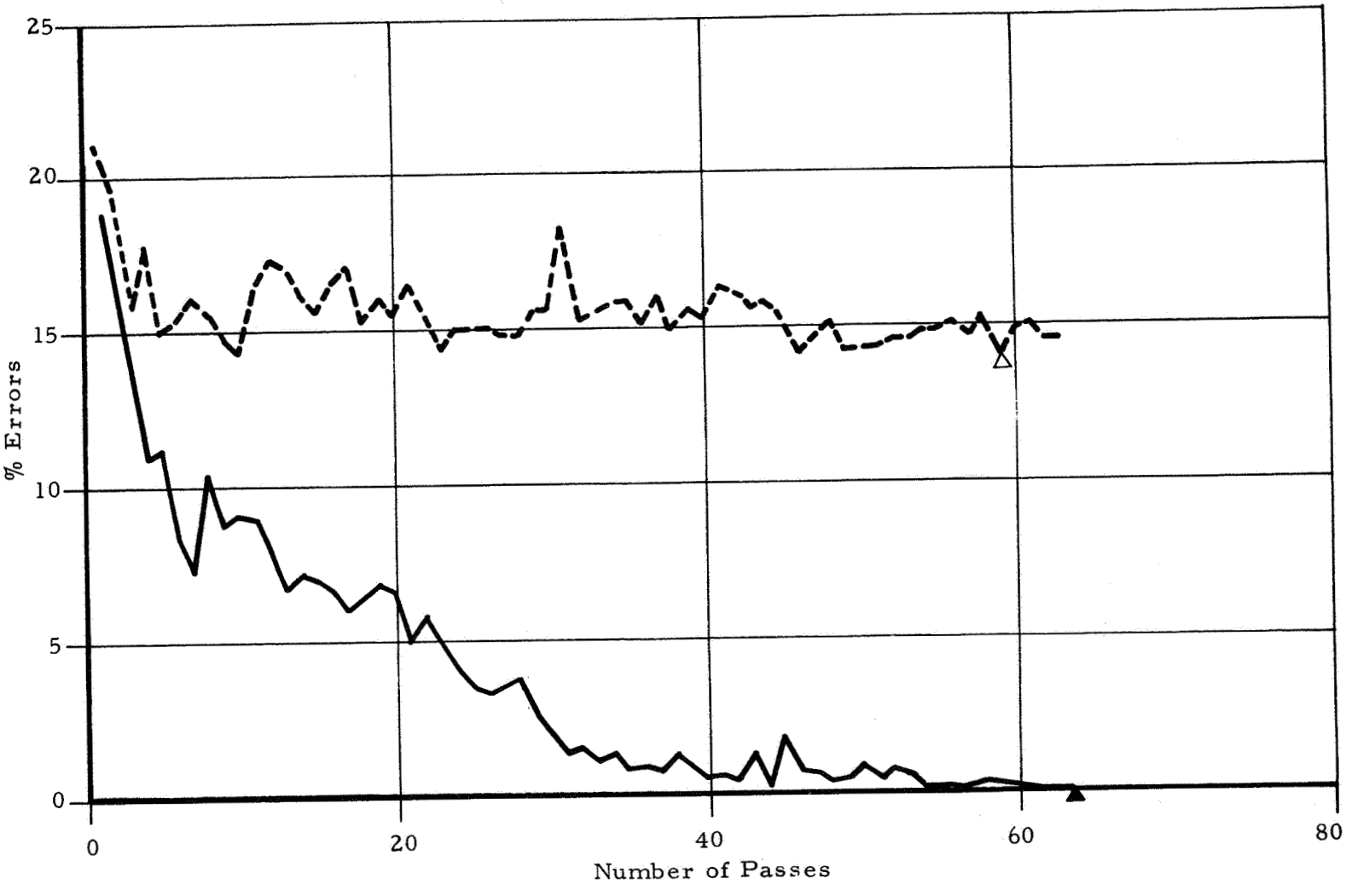


Figure 26. MADALINE – Cumulus vs. Noncumulus

and valleys of the generalization curve tend to coincide with peaks and valleys of the classification curve.

2. Solid Cell vs Polygonal Cells

The performance figures obtained on the task of separating the solid cell patterns from the polygonal cell patterns are very similar to those cumulus-noncumulus tasks. The results appear in Figure 27.

Complete separation of the training patterns was not obtained but the MADALINE and iterative design techniques both achieved less than 2 percent error rate. These two techniques also provided the best generalization performance. Error correction and iterative design again exhibit their best generalization early in the design. As before, the MADALINE shows the least sacrifice in generalization when classification performance level is used to terminate the design process. Iterative design performance falls almost to the level of Bayes weights, and error correction falls noticeably below that level.

Figures 28 through 31 show the cycle by cycle performance for the recursive techniques. The greater irregularity of the MADALINE and error correction curves appears to be more typical of the results in this study than those of the preceding section.

More recent designs for Tasks PVS and NVC have been accomplished using the QSID program. Generalization performance of 87.5 and 86.0 were achieved, using 50 and 100 property filters for these tasks, respectively. Further descriptions of these systems are given in Section 3.3.7.

3. Multiple "Looks"

For three of the four pattern categories in the NIMBUS tasks, the generalization sample of 200 patterns per class were derived from 50 basic patterns for each class. The basic patterns were sampled at two translations for each of two rotations. Thus, there are four "looks" at each of the 50 basic patterns for these classes. For the remaining category, Cumulus, 100 basic patterns were sampled at two rotations, providing two "looks" at each pattern.

Technique	Classification %	Generalization %
Forced Learning	86.10	81.00
Bayes Weights	86.10	82.00

	Total Cycles	For Best Classification			For Best Generalization		
		Classif. %	Gen. %	Cycles	Classif. %	Gen. %	Cycles
Error Correction	192	92.40	80.50	34	89.30	83.00	1
Iterative Design	47	98.50	82.75	47	90.50	85.50	1
Mean Square Error	49	86.95	80.75	25-49	86.95	80.75	25-49
Madaline	112	99.85	85.00	76	97.40	85.50	72

C/959

Figure 27. Solid Cells vs. Polygonal Cells — 400 DAID Units

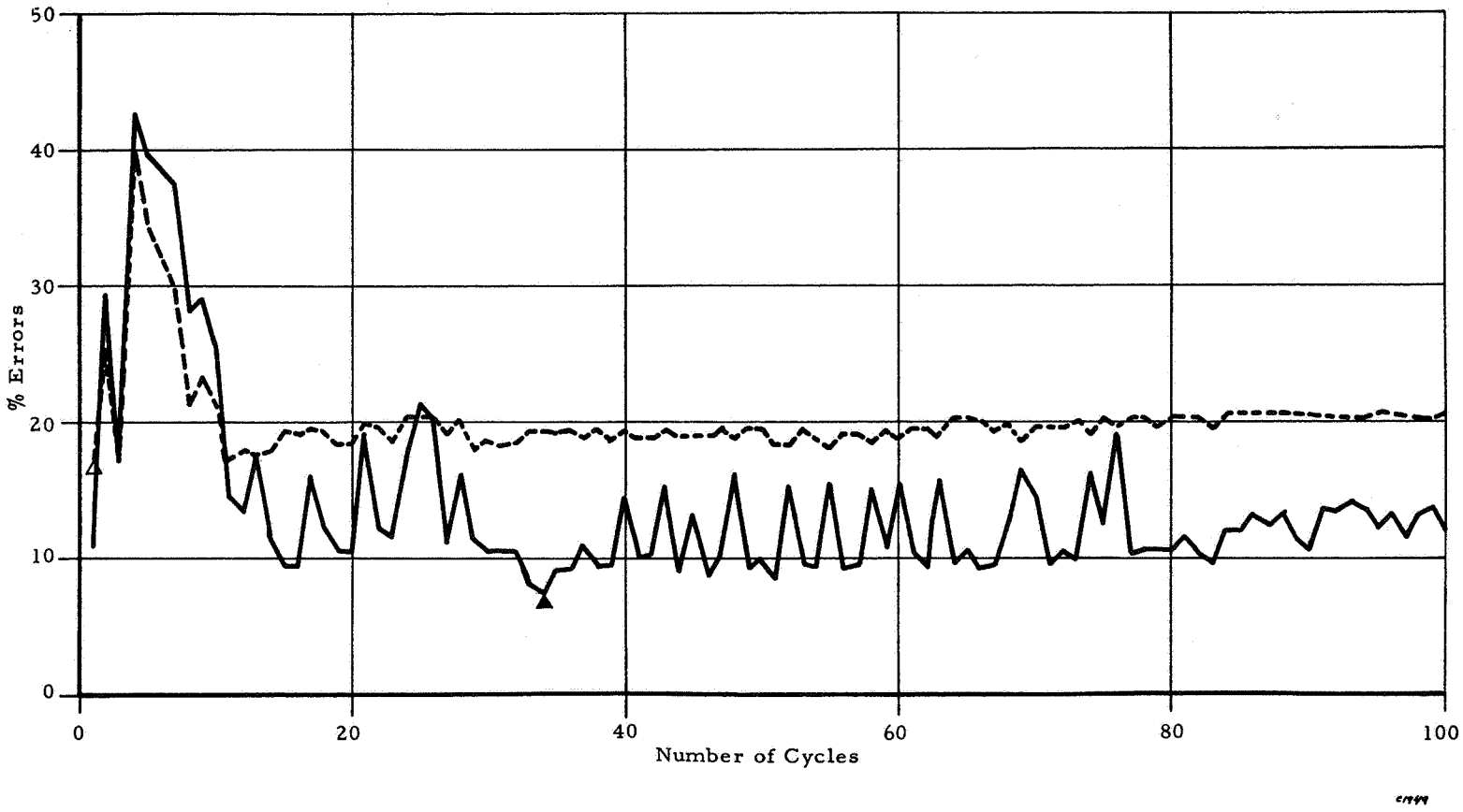


Figure 28. Error Correction— Solid Cells vs. Polygonal Cells

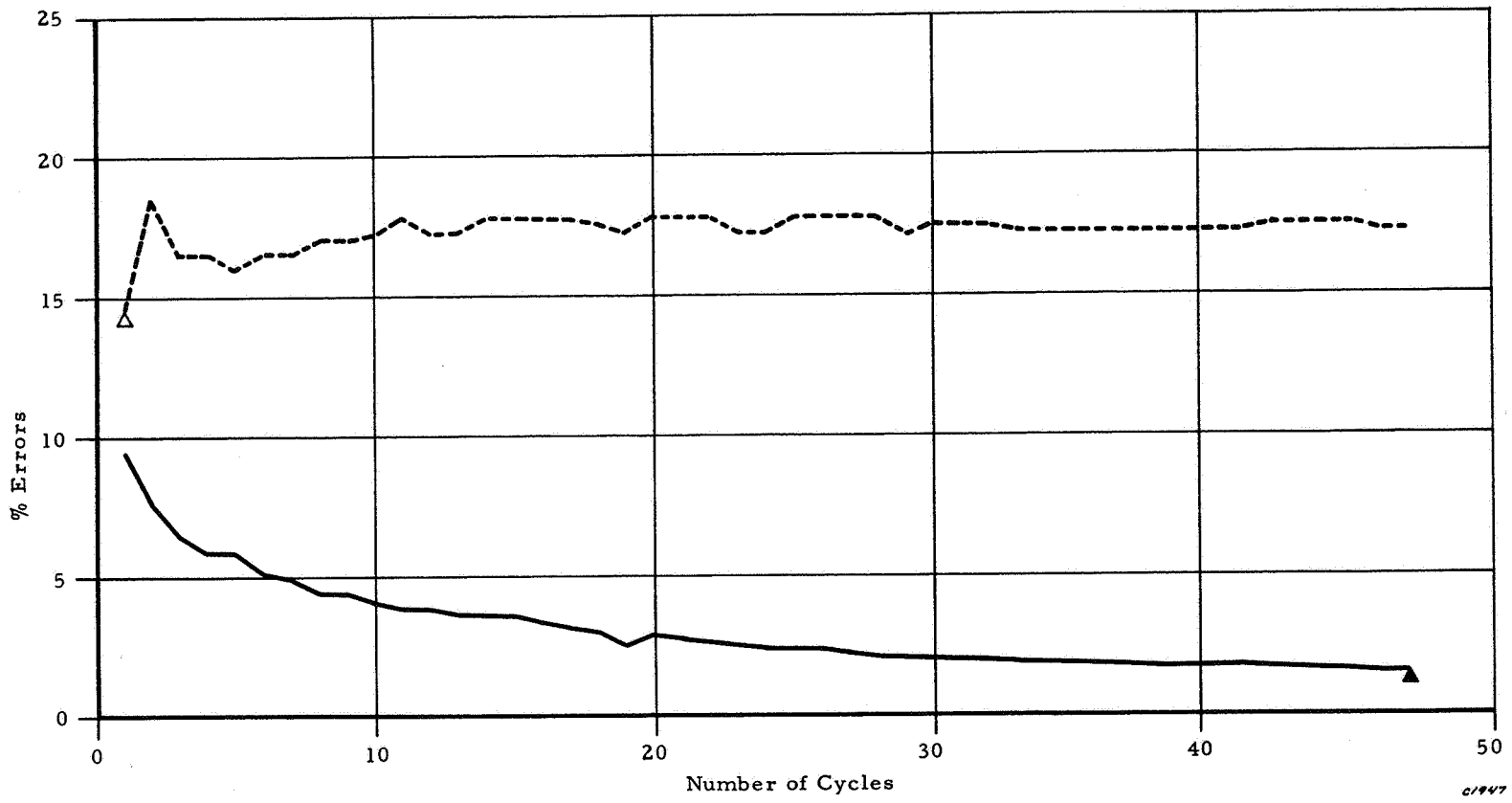


Figure 29. Iterative Design – Solid Cells vs. Polygonal Cells

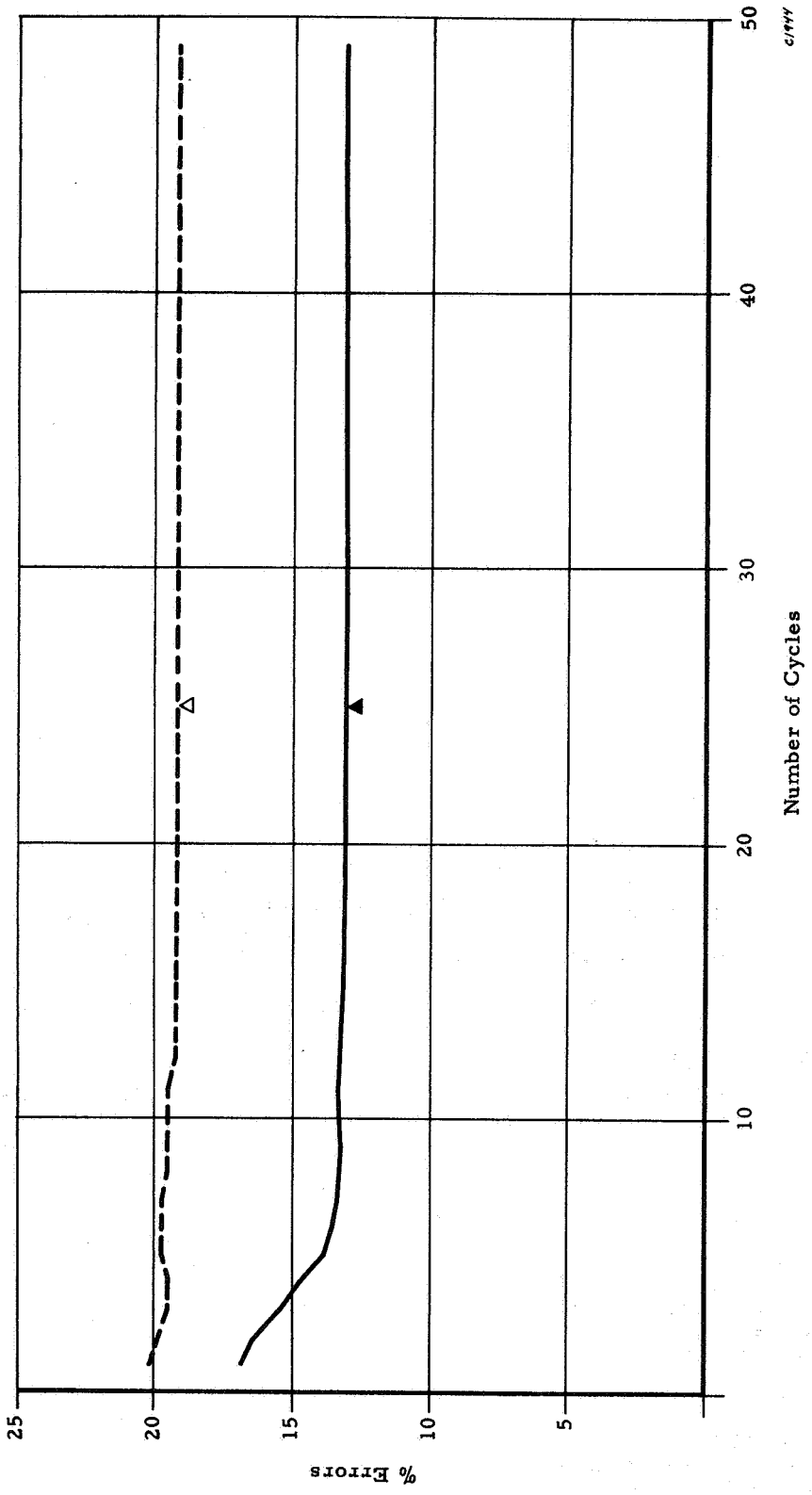


Figure 30. Mean Square Error — Solid Cells vs. Polygonal Cells

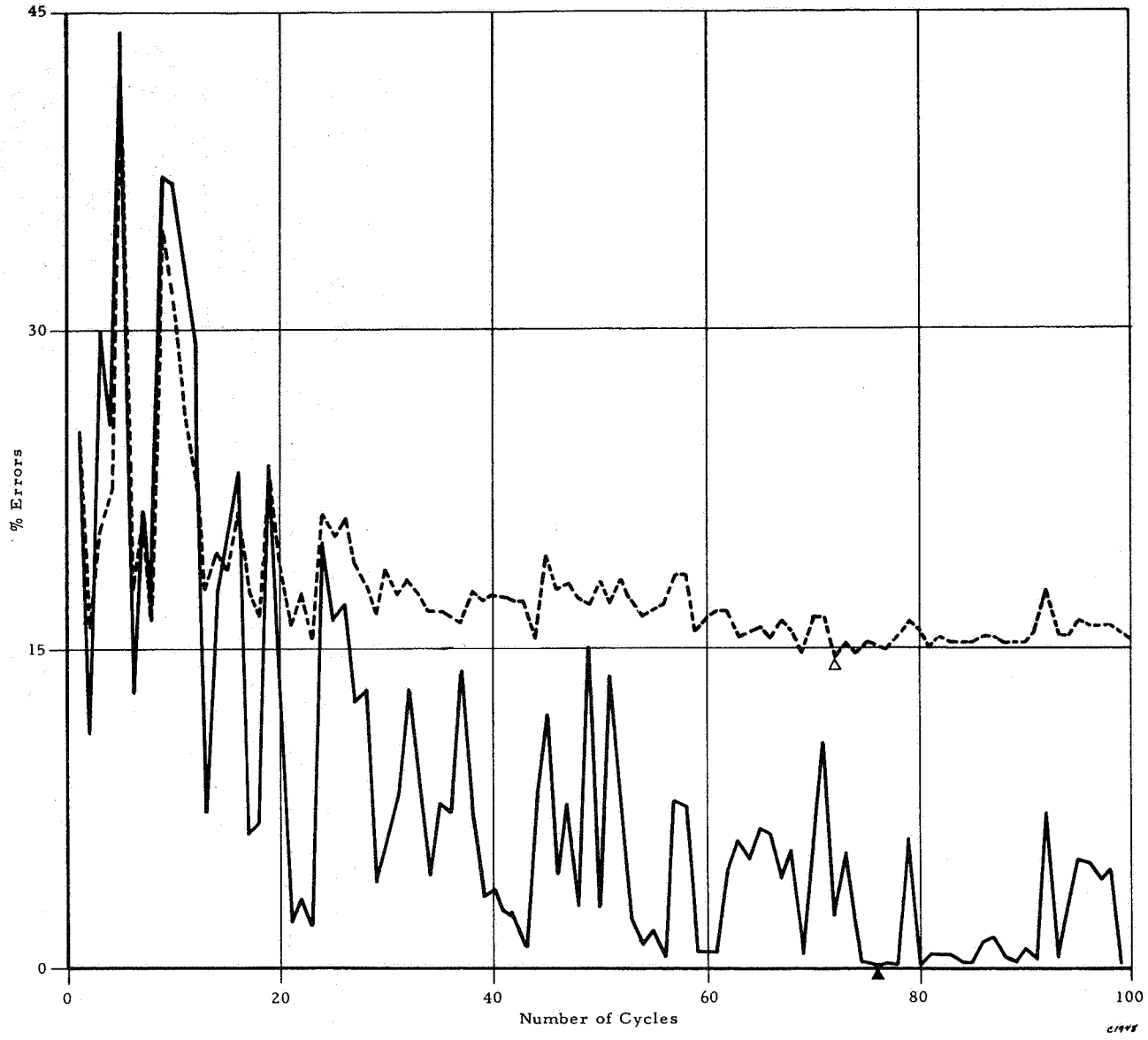


Figure 31. MADALINE – Solid Cells vs. Polygonal Cells

The possibility of improving generalization performance by combining the decisions of the several looks* was investigated for all techniques. The method consisted of simply summing the values of linear decision functions for all of the looks. The results are given in Table II.

If the decisions for the different "looks" were completely dependent, that is, the correctness of the decisions depends only on the pattern and not on its location or orientation, one would expect no change in performance. If the looks were completely independent, one might compute the expected performance as follows. Over the sample patterns in a category, the linear decision function may be considered to be approximately normally distributed. To achieve 80% performance the magnitude of the mean to standard deviation ratio should be 0.84. Considering the sum of two such independent variables, the mean is doubled, but the standard deviation is multiplied by $\sqrt{2}$. Thus the ratio is multiplied by $\sqrt{2}$. For four independent "looks," the mean is quadrupled and the standard deviation doubles. Thus the ratio doubles. The expected performance for two looks in this case is 88.3% and for four looks 95.3%. Table III gives some typical values.

The performance levels actually achieved are greater than the single look levels, but are much closer to these values than those to be expected if the looks were independent. This indicates that the original decisions are somewhat, but not greatly, dependent on position and orientation. The use of a larger training sample is suggested.

The multiple look method may be implemented by sequential processing, or by the inclusion of four times as many property filters, or by a combination of these methods.

3.3.4.4 Additional Lunar Data Experiments

This section describes additional experimentation which was performed on the lunar data. The primary tool consisted of

---

*This approach was suggested by J. H. Munson of the Stanford Research Institute, who uses nine "looks" at each pattern, computes a decision for each look, and derives a majority decision. His method is thus similar to a MADALINE in which the ADALINES compute similar functions.

TABLE II

MULTILOOK GENERALIZATION PERFORMANCES
PERCENTAGE CORRECT DECISIONS

Adaptive Technique	Task NvC		Task PvS	
	Original	Multilook	Original	Multilook
Forced Learning	81.2	84.7	81.0	86.0
Bayes Weights	82.5	83.3	82.0	86.0
Error Correction	85.8	92.0	83.0	87.0
Iterative Design	86.2	88.7	85.5	89.0
Mean Square Error	78.5	88.7	80.8	85.0
MADALINE	86.0	88.3	85.5	87.0

TABLE III

EXPECTED PERFORMANCE FOR INDEPENDENT LOOKS (%)

Single Look	Two Look	Four Look
75	83.5	91.6
80	88.3	95.3
85	92.9	98.1
90	96.5	99.5
95	99.0	99.95

manipulating the patterns themselves. For Task CVR (craters vs ridges-rima), on which 99.5 generalization performance had been obtained, the patterns were scrambled in an attempt to determine the basis for this high performance. For Task RVR (ridges vs rima), a set of artificial, noise-free patterns was generated for the system design, with the system test being performed on real patterns. For both Tasks RVR and CVC (craters with vs craters without central elevations), design and testing were conducted with reduced aperture sizes, using 25 by 25 and 15 by 15 subsections selected from the original 50 by 50 patterns. The property filter sets for these experiments were generated by the newer computer program (QSID) written for the SDS 930.

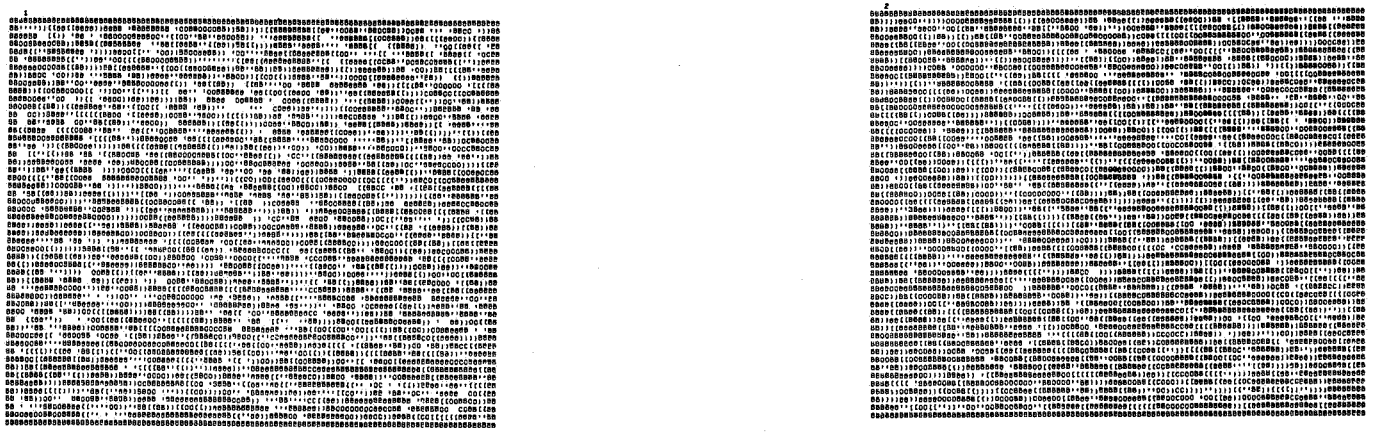
1. Scrambled Patterns

On Task CVR, the very high percentage of 99.5 correct decisions was obtained for both the crater and the ridge-rima groups. It was speculated that the system achieved this by detecting the presence or absence of the distinctive crater formations, with little or no emphasis on the detection of the less prominent ridges and rima.

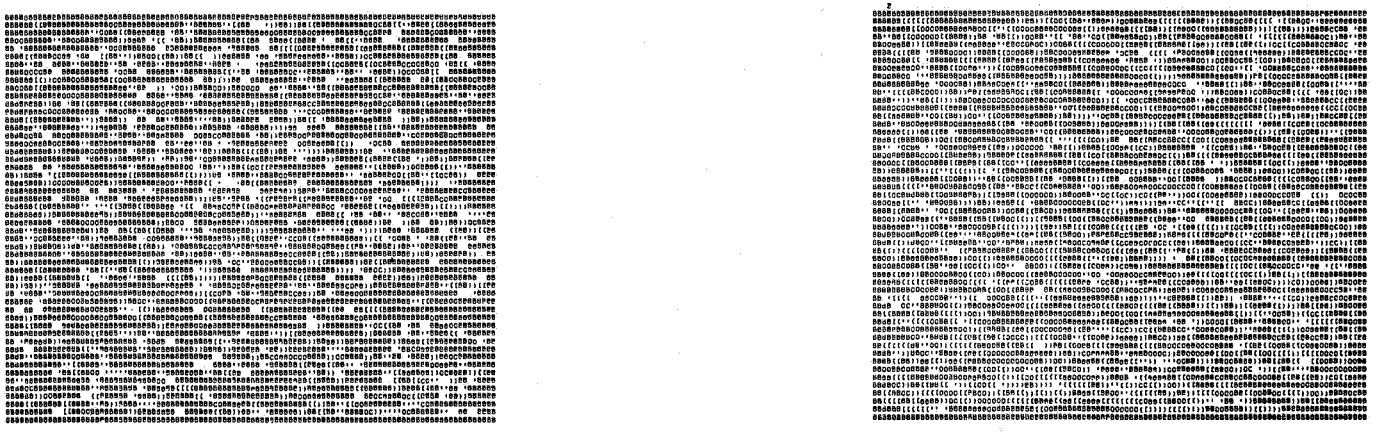
To test this hypothesis, each of the 400 test patterns for this task was scrambled. Each of the 2500 picture points in a pattern was randomly repositioned in a 50 by 50 raster. Since all of the original picture points are used in the scrambled picture, the gray scale distribution of each pattern remains unchanged. A different random map was used for each picture. Four examples of the random patterns are shown in Figure 32.

The scrambled generalization patterns were then used to test the system. If the conjecture that only the craters were being recognized was correct, one would expect nearly all of the 400 patterns to be classified as ridges-rima, as none of them has any resemblance to a crater. Additionally, performances greater than chance in both categories would indicate that at least part of the decision depends upon gray scale distribution. This test gains its validity from the high performance levels achieved on the unscrambled patterns.

The results of this experiment are shown in Table IV. The results clearly show that the system recognizes patterns of



Craters



Linear Features

c2720

Figure 32. Scrambled Patterns

TABLE IV

CRATERS VS. LINEAR FEATURES
ITERATIVE DESIGN GENERALIZATION

	Craters	Linear Features	Total
Normal Patterns	99.5	99.5	99.50
Scrambled Patterns	51.5	49.0	50.25
Scrambled Patterns Best Generalization	51.5	52.5	52.00

both classes, and that the differences in gray scale distribution do not contribute to the recognition process.

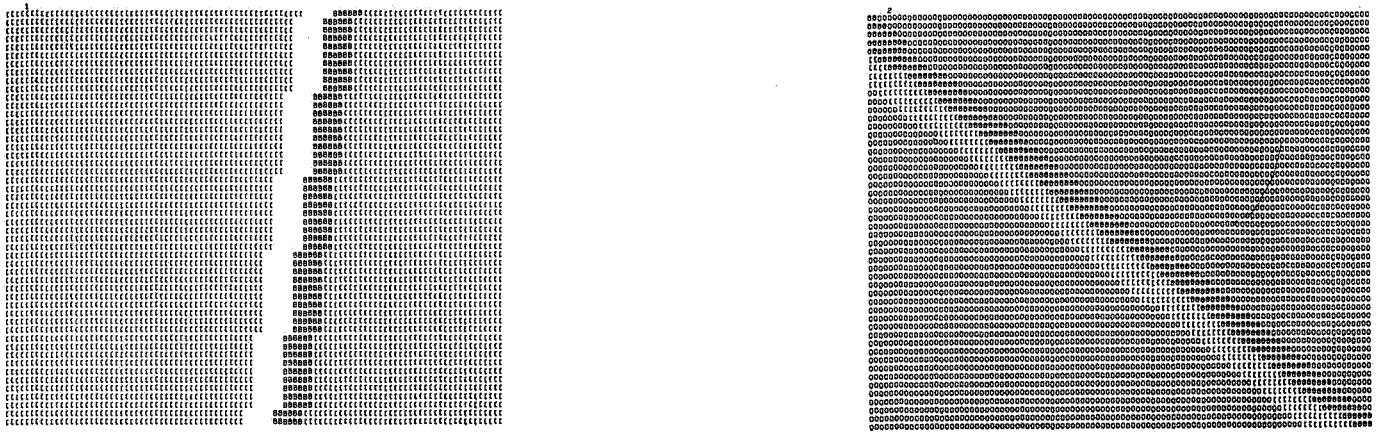
2. Artificial Patterns

On the two remaining lunar tasks, the performances obtained earlier were less satisfactory. One obvious explanation lies in the signal-to-noise ratio. For Task CVC, on which the poorest performance was achieved, the signal, or significant feature is the (presence or absence of the) central elevation. Even in the most prominent examples, this feature is small. For Task RVR, the significant feature is only slightly larger and it is often further weakened when the bright part or shadow part of the feature blends in with the background gray level.

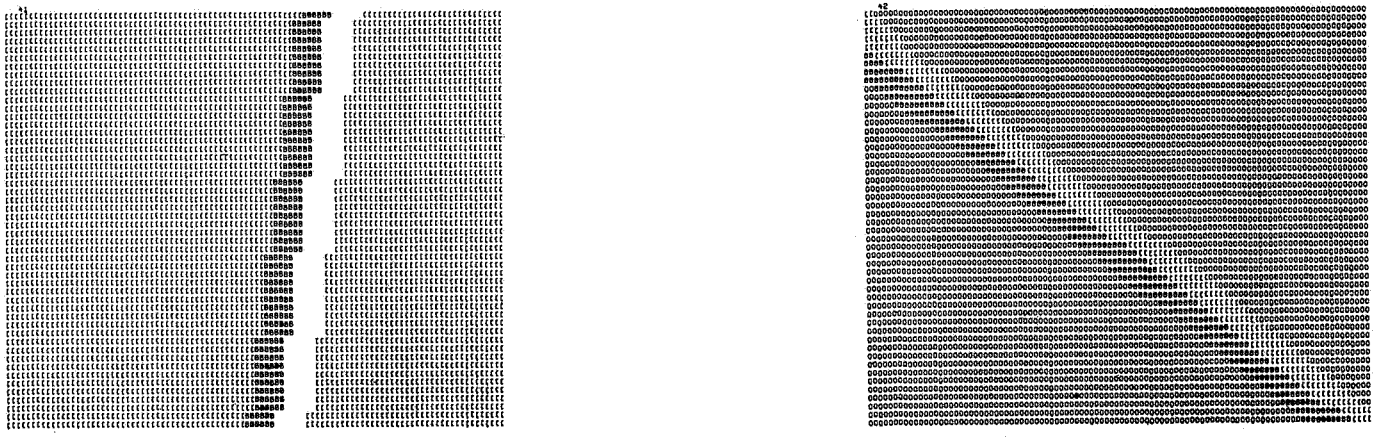
In one attempt to improve performance on Task RVR, a set of artificial, noise-free training patterns was generated. The artificial patterns consisted of a uniform gray background, on which was imposed a pair of adjacent stripes, one lighter, and one darker than the background gray level. Complementary pairs of ridge and rima patterns were generated, the ridge pattern having the lighter stripe on the right, and the rima having the lighter stripe on the left. Two such pairs are shown in Figure 33.

A relatively casual examination of the real patterns was used to establish ranges for the parameters of the artifical patterns. For the angle of the stripes, a trimodal distribution with a range of ± 64 degrees (from the vertical) was used. The center of the pattern used a uniform distribution with a range of ± 5 raster elements from the center of the aperture. The distribution of the three gray levels was most heavily concentrated toward the darker end of the scale. From these parameter distributions, 1000 pairs of artificial ridges and rima were generated.

Networks were designed to give perfect separability of the artificial patterns, and the performance of these networks was tested on the (real) generalization patterns. The performances achieved are given in Table V. It is clear that this experiment was a failure, since the best network only provides 60.5 percent performance, compared with the



Rima



Ridges

Figure 33. Artificial Patterns

TABLE V

ARTIFICIAL RIMA VS. RIDGES (50 x 50), 280 QSID UNITS

Technique	Classification %	Generalization %
Forced Learning	92.40	58.75
Bayes Weights	93.75	58.50

	Total Cycles	For Best Classification			For Best Generalization		
		Class. %	Gen. %	Cycles	Class. %	Gen. %	Cycles
Error Correction	322	100.00	53.75	322	100.00	53.75	322
Iterative Design	6	100.00	58.50	5	98.10	59.25	1
MADALINE	21	100.00	58.75	21	96.10	60.50	3

75.75 percent achieved when real patterns were used for training. The probable cause of this failure is inadequate parameter distribution.

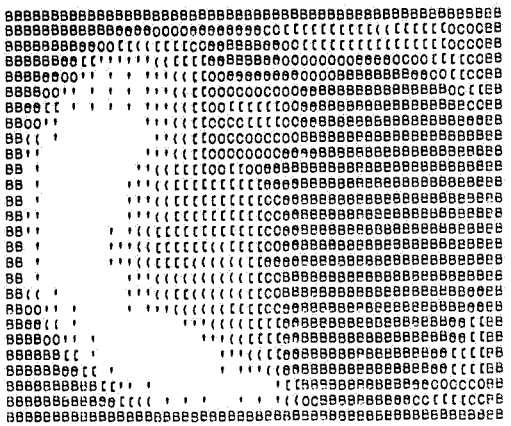
3. Reduced Aperture Patterns

In a second approach toward improved signal-to-noise ratios, reduced scanning apertures were used. Subapertures of 15 by 15 and 25 by 25 were applied to each sample lunar pattern, training and test. The subapertures were positioned to include as much as possible of the significant features of the patterns.

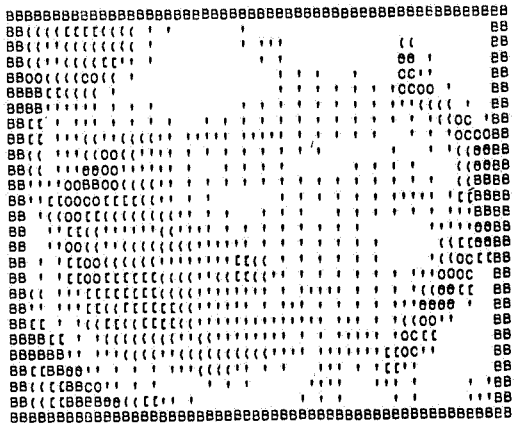
Figures 34 through 37 give examples of the 25 by 25 and 15 by 15 patterns. The effect of reducing the aperture is greater on the craters than on the ridges and rima. This is because part of the ridges and rima are sacrificed as the aperture size is decreased, while none of a central elevation is lost. The craters with elevations thus gain quadratically with aperture size reduction, while the ridges and rima gain only linearly.

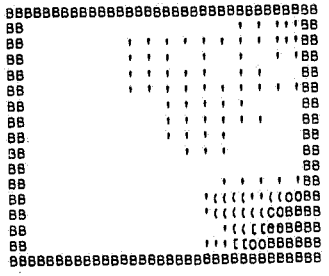
Results for Task RVR using the 25 by 25 aperture are given in Table VI. As with all networks designed on the reduced aperture patterns, 100 percent classification of the training patterns was achieved with the three recursive techniques used (Error Correction, Iterative Design, and MADALINE). The networks used 260 property filters. This is the number of filters that the QSID program needed to completely separate the training patterns. Later results in this section indicate that a somewhat smaller set of these units would probably be adequate for complete training pattern separability, and a still smaller set would likely yield marginally better generalization results. With the recursive techniques, generalization performances are all at least as good as the 75.75 percent achieved earlier on the 50 by 50 patterns, but the maximum 77 percent is not significantly better.

Table VII presents the results obtained on the RVR task using the 15 by 15 aperture. These networks used 235 property filters. Generalization performances showed a more substantial improvement, reaching a level of 84.5 percent with the MADALINE technique. Using the Iterative Design technique, decision functions were derived for the first 180, 190, 200, 210, 220, and 230 property filters in this set. Two hundred and ten

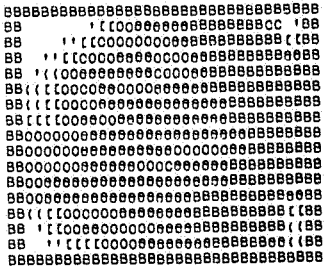


25 x 25



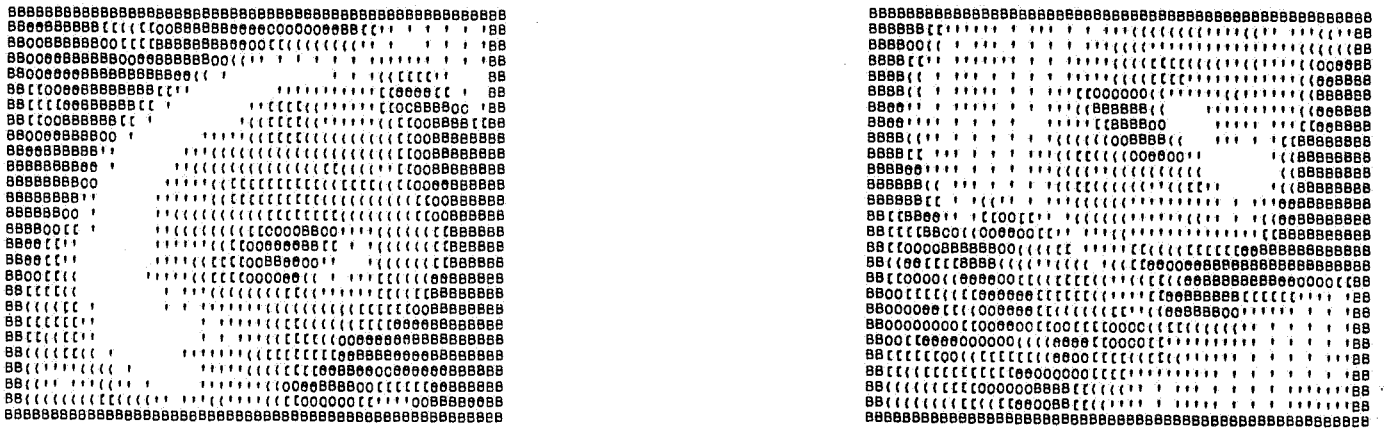


15 x 15

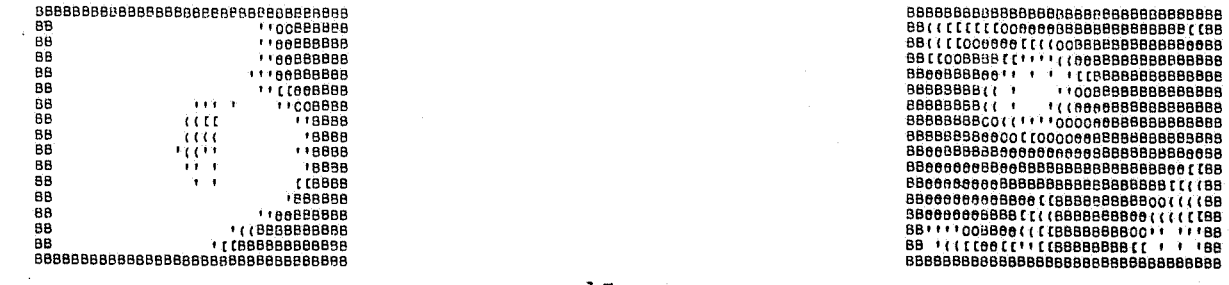


c2722

Figure 34. Craters Without Elevations



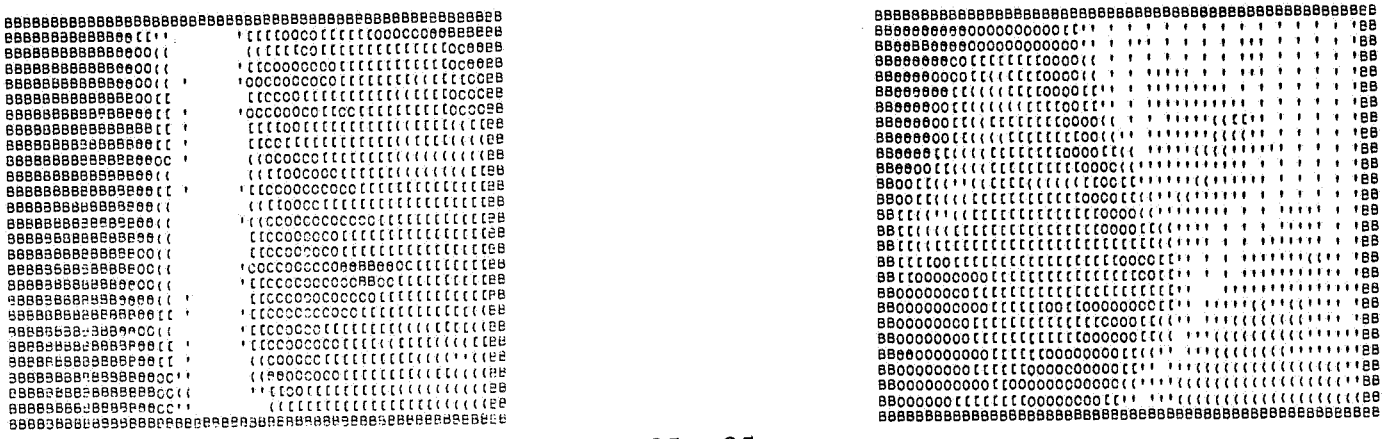
25 x 25



15 x 15

cz733

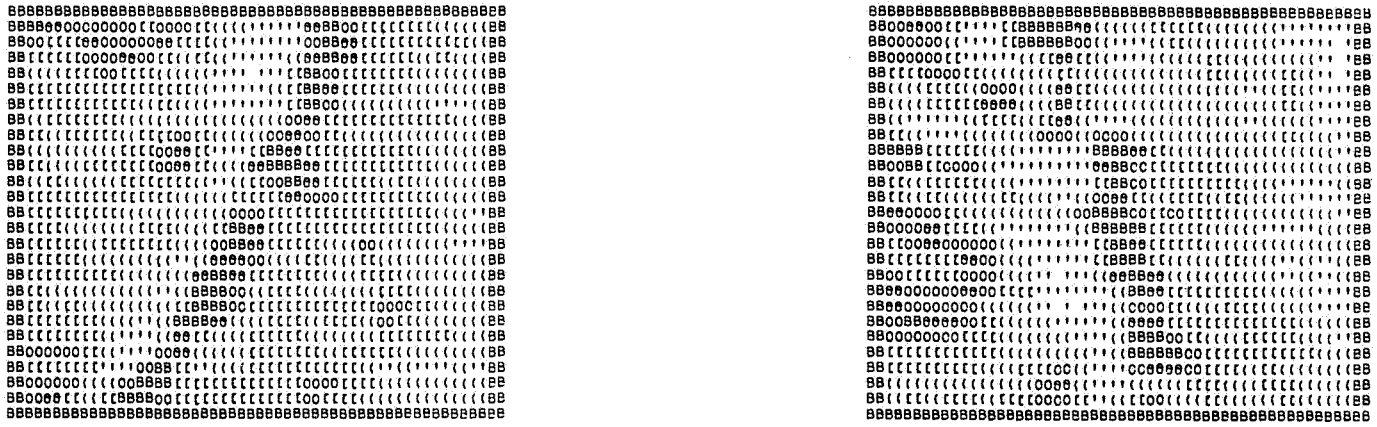
Figure 35. Craters With Elevations



25 x 25

15 x 15

Figure 36. Wrinkle Ridges



25 x 25

15 x 15

Figure 37. Rima

TABLE VI

RIMA VS. RIDGES (25 x 25), 260 QSID UNITS

Technique	Classification %	Generalization %
Forced Learning	75.50	65.25
Bayes Weights	75.45	65.25

	Total Cycles
Error Correction	194
Iterative Design	6
MADALINE	41

For Best Classification		
Class. %	Gen. %	Cycles
100.00	75.50	194
100.00	73.50	6
100.00	73.75	41

For Best Generalization		
Class. %	Gen. %	Cycles
95.60	76.00	72
99.30	75.75	2
88.65	77.00	3

TABLE VII

RIMA VS. RIDGES (15 x 15), 235 QSID UNITS

Technique	Classification %	Generalization %
Forced Learning	84.70	83.25
Bayes Weights	84.85	82.75

	Total Cycles	For Best Classification			For Best Generalization		
		Class. %	Gen. %	Cycles	Class %	Gen. %	Cycles
Error Correction	138	100.00	77.75	138	84.05	80.50	11
Iterative Design	7	100.00	78.75	7	100.00	78.75	7
MADALINE	38	100.00	78.00	38	88.20	84.50	4

property filters were enough to provide complete separation of the training patterns. The best generalization performance was obtained with 180 property filters, which gave 79.75 percent correct decisions — an improvement of one percent.

The results on Task CVC show more dramatic improvements than on Task RVR. This is no doubt partially due to the quadratic effect on crater patterns of aperture reduction as opposed to the linear effect on ridges and rima noted earlier. Part of the difference may also stem from the better quality of the crater patterns. Poorer initial performance (63.75 percent on the 50 by 50 patterns) also helps to underscore the improvement. Table VIII presents the results achieved with the 25 by 25 aperture. Two systems achieve 82.5 percent on the generalization patterns. These results are based on the full set of 140 property filters used by the QSID program to achieve complete separation. Using the Iterative Design technique, the effectiveness of smaller sets of property filters was investigated. Figure 38 presents the classification and generalization error percentages as a function of the number of property filters used. In each case, the property filter subsets represented an initial segment — the set of 100 units, for example, being the first 100 property filters designed by the QSID program. The curves of Figure 38 are apparently typical. A limited reduction in the property filter set produces a slight improvement in the generalization performance (in this case, 120 units give 83.25 percent). The most reasonable explanation for this phenomenon is that near the end of a QSID run, only a very small number of patterns influence the selection of the property filters. Lacking the statistical defense of numbers, the last property filters selected most likely are based on the individual noise characteristics of the remaining patterns. Even with the reduced aperture, the central elevations occupy less than five percent of the input field.

When the aperture is reduced to 15 by 15, another large increase in performance is noted. Only 65 property filters are needed to produce the 96.25 percent performance given in Table IX. Figure 39, which shows the effect of using smaller sets of property filters, does not indicate any improvements over the full set of units.

TABLE VIII

CRATERS VS. CRATERS WITH ELEVATIONS (25 x 25), 140 QSID UNITS

Technique	Classification %	Generalization %
Forced Learning	87.90	63.75
Bayes Weights	87.75	63.75

	Total Cycles	For Best Classification			For Best Generalization		
		Class. %	Gen. %	Cycles	Class. %	Gen. %	Cycles
Error Correction	65	100.00	81.50	65	89.00	82.50	21
Iterative Design	9	100.00	82.25	4	99.90	82.50	3
MADALINE	15	100.00	80.00	15	99.90	80.75	13

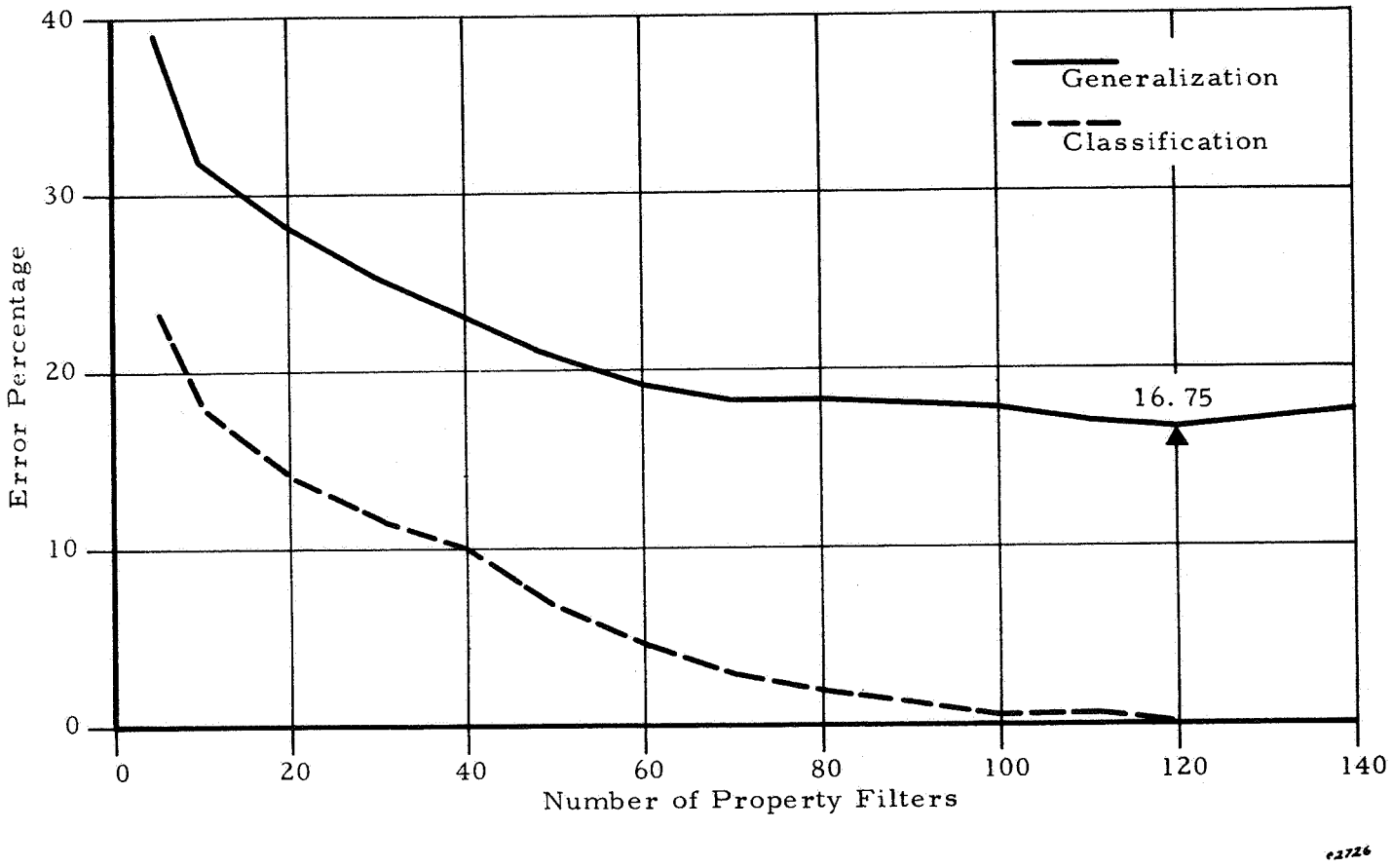


Figure 38. Craters vs. Craters With Elevations (25 x 25)

TABLE IX

CRATERS VS. CRATERS WITH ELEVATIONS (15 x 15), 65 QSID UNITS

Technique	Classification %	Generalization %
Forced Learning	96.70	87.75
Bayes Weights	96.80	87.75

	Total Cycles	For Best Classification			For Best Generalization		
		Class. %	Gen. %	Cycles	Class. %	Gen. %	Cycles
Error Correction	40	100.00	91.00	40	100.00	91.00	40
Iterative Design	8	100.00	96.25	4	100.00	96.25	4
MADALINE	26	100.00	93.75	26	99.90	95.00	19

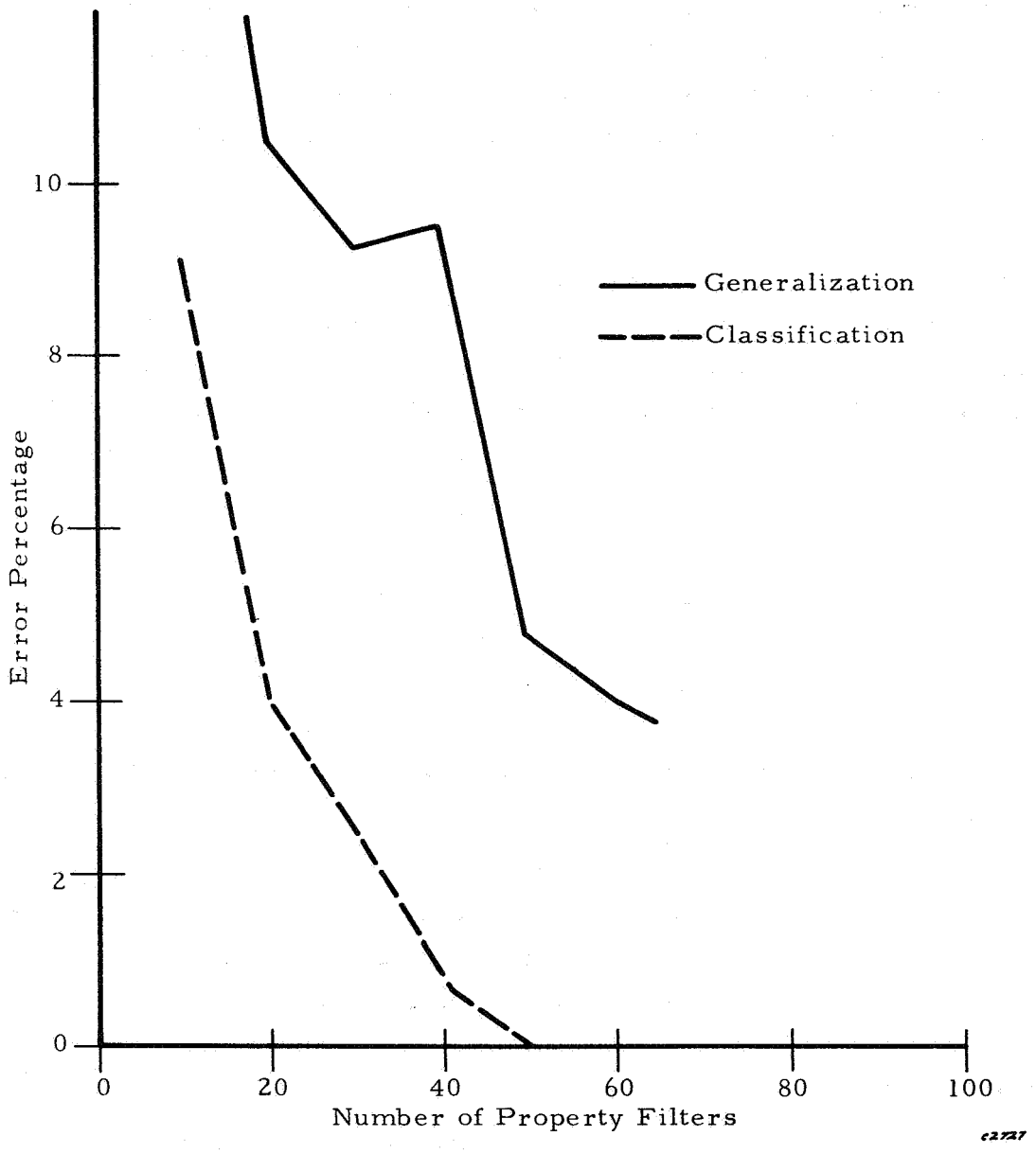


Figure 39. Craters vs. Craters With Elevations (15 x 15)

3.3.4.5 Summary

Table X summarizes the results achieved on the five (full aperture) recognition tasks by the six decision function algorithms. With some exceptions, performance differences are not large. Mean square error falls down on four of the five tasks. If one downgrades the results on Tasks CVC and RVR as being too poor for useful systems, then the nonrecursive forced learning and Bayes weights algorithms appear weak. MADALINE falters on Task CVR. Iterative design appears to be the most consistent performer.

MADALINE has the advantage that it usually provided a negligible loss in generalization performance when best performance on the training patterns is used as the criterion for terminating the design process. It has the disadvantages of requiring three times as many adjustable weights as the other techniques and longer design times due to the inefficient algorithm and the late occurrence of the optimal performance points.

Iterative design has the advantage that best generalization usually occurs very early in the design process, offering the possibility of very short design times. Further design improves performance on the training patterns in a very smooth fashion, while similarly decreasing the generalization performance. The smoothness of the curves makes the selection of the optimum point for stopping less critical. Generalization performance loss when classification performance is used as the stopping criterion is moderate.

Error correction shares with iterative design the advantage of early peaking of the generalization performance. Generalization loss was usually the largest with this technique with classification as the stopping criterion. As with MADALINE, performance curves are irregular, making stopping point selection more critical; and the algorithm is inefficient, improving classification performance very slowly.

The quadratic property filters (DAID units) provide significantly higher performance levels than do the linear property filters (SDA) units, although the total number of adjustable parameters in the specification of the units was about the same for both property filter sets.

TABLE X

GENERALIZATION PERFORMANCES FOR 30 SYSTEMS
PERCENTAGE CORRECT DECISIONS

Adaptive Technique	Task				
	Lunar Features			Cloud Features	
	CvC	RvR	CvR	NvC	PvS
Forced Learning	62.8	71.0	74.8	81.2	81.0
Bayes Weights	63.8	75.2	77.5	82.5	82.0
Error Correction	59.8	70.2	99.2	85.8	83.0
Iterative Design	59.0	73.8	99.5	86.2	85.5
Mean Square Error	43.0	74.5	82.2	78.5	80.8
MADALINE	58.0	75.8	89.5	86.0	85.5

Three factors may contribute to this result: the quadratic units are based on contrasts in the patterns as well as average brightnesses (the basis of the linear units); selection was used with the DAID units, that is, only the best 10 percent of the units generated were used; and the DAID set was constructed sequentially, with later units selected to complement the earlier ones.

The multiple look analysis indicates that the discrimination capability of the property filters is somewhat dependent upon translation and rotation of the patterns. This may be remedied by including more translations and rotations in the set of training patterns so that the property filters derived are less sensitive to these pattern changes, and/or replicating the property filters derived in other translations and rotations to implement the multiple look technique.

A study of the very high performance system designed for Task CVR indicates that this system does actually recognize patterns of both classes, and not the presence or absence of patterns of one of the classes.

Results obtained using artifical noise-free patterns for training a system were not encouraging. Indications are that extreme care is necessary in defining the variations of the artificial patterns so that they are representative of the real patterns.

The performance obtained on each task appears to be very strongly related to the size of the significant pattern features relative to the size of the sensory field. When the significant feature represents less than one percent of the field, the best generalization performance achieved was 64 percent. When the feature was five to ten percent of the field, the best generalization was 76 percent. With features covering 30 percent of the field, the generalization performance better than 99 percent was achieved.

For patterns such as the lunar data, reducing the aperture size for small features is an effective technique as can be surmised from the comment made above. Using this method, performance on Task RVR was increased from 75.75 to 84.5 percent, and on Task CVC from 63.75 to 96.25 percent.

3.3.5 Additional Decision Functions

3.3.5.1 Introduction

Earlier results obtained by using a variety of algorithms to design decision networks for a given set of property detectors did not show a clear superiority for any technique. Except for mean square error, the recursive techniques all appeared to be equally effective.

MADALINE was the only technique tested earlier which yielded a nonlinear decision on the property profiles. Consequently, it has an inherently greater capability than the other techniques, gained at the expense of greater system complexity. Its failure to exploit this added capacity might be attributed to the fact that the particular property filters used were designed to achieve linear separability of the training patterns. Two new nonlinear decision techniques were included in this study, and provide a partial confirmation of this hypothesis. They also failed to outperform the linear techniques when applied to the same sets of property filters.

The two new techniques are attractive in their own right. The piecewise-linear technique has been favorably reported in the literature.[115] The Distribution Estimation method provided some good results as reported in Section 3.3.6.3 of this report.

The Error Correction and MADALINE techniques required relatively long design times. Furthermore, these two designs exhibited strong oscillations in performance as a function of the number of training cycles. This behavior made it difficult to select a proper stopping point for the design process. In an attempt to overcome these difficulties one modified Error Correction algorithm, and two modified MADALINE algorithms were tested.

3.3.5.2 Distribution Estimation

A distribution estimation technique was applied to the QSID and DAID property profiles derived earlier for sample patterns. This method is nonparametric, and is intended to recover from sample vectors an estimate of the (continuous) underlying distribution. Because this technique was intended for continuous distributions, it is perhaps not at its best when

applied to the binary property filter outputs. The program was a modification of the one written for use in designing property filters (Section 3.3.6.3), and was limited to the processing of 150-dimensional distributions.

Given M n-dimensional sample vectors (training), denote the j-th coordinate of the i-th sample by y_{ij}. Let the j-th coordinate of a test pattern be denoted by x_j. The likelihood estimate for the test vector X is given by:

$$f(X) = \frac{1}{M} \left\{ \prod_{k=1}^{n} \sqrt{b_k/\pi} \right\} \left\{ \sum_{i=1}^{M} \exp\left(-\sum_{j=1}^{n} b_j (x_j - y_{ij})^2 \right) \right\}$$

A more complete description of this estimate, and the means for estimating the parameters b_j is given in Section 3.3.6.3.

A decision is derived for a test pattern by estimating the likelihood functions (i.e., density estimates for the test pattern vector) for each class, using the sample patterns and b_j values appropriate to that class, and then selecting the largest likelihood value. The resulting decision surface is highly nonlinear.

The technique was first applied to the QSID property filters for Task CVC, for 5, 25, 35, and 65 property filters. Figure 40 shows the results, as compared with the decision functions designed by iterative design. Distribution estimation does better with truncated property filter sets, but not when the full set of filters is used.

The technique was then applied to the first 35 DAID property filters for Task NVC. It achieved 76.5 percent generalization, as compared with 73.75 percent for error correction with the same property filters. Using the later QSID property filters for Tasks PVS and NVC (50 and 100 filters, respectively) the generalization figures were 87.25 and 78.25 percent, as compared with 87.50 and 86.00 percent obtained with iterative design. On Task RVR (15 x 15 aperture), 25 property filters were used. Distribution estimation gave 77 percent while error correction gave 79.25 percent.

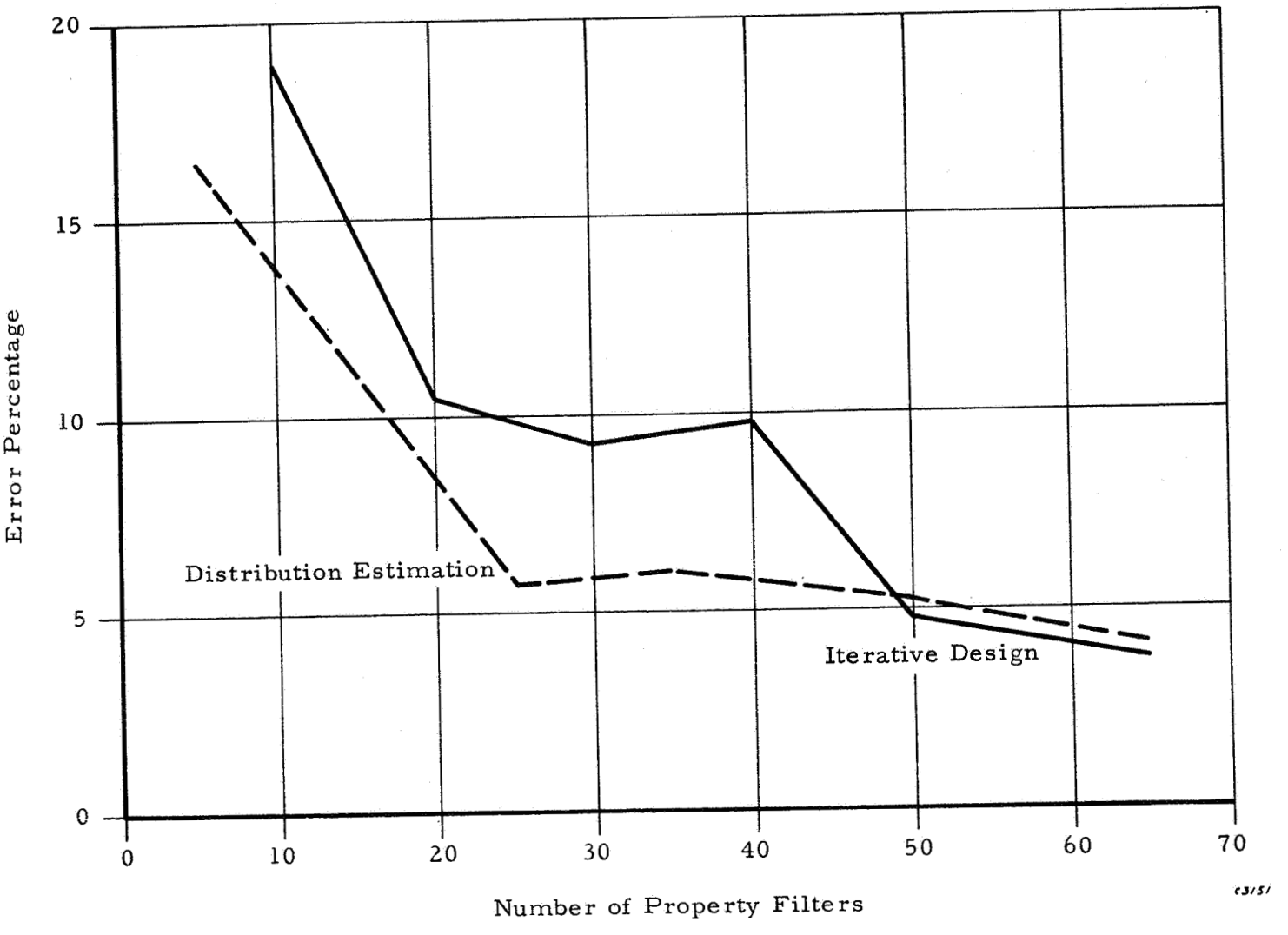


Figure 40. Performance on Task CVC (15 x 15 Aperture)

Thus the distribution estimation technique achieves performances which are, for the most part, about average on these tasks at the expense of a much more complex algorithm.

3.3.5.3 Piecewise-Linear

In the piecewise-linear algorithm used in this study, an even number of response units was selected. Half of the response units were assigned to each class. A decision is achieved by determining the response unit with the highest input sum. The decision surface is piecewise-linear, and each response unit input weight vector hopefully represents a cluster point of the training patterns of the appropriate class.

The algorithm is implemented as follows. If a correct decision on a training pattern is obtained, no changes are made in the response units' input weights. If an incorrect decision is made, the response unit for the correct class with the largest input sum is determined. For this response unit, each input weight from an active property filter is incremented by $\frac{\delta_i}{m}$, where m is the total number of active property filters. On odd-numbered cycles (through the training patterns) only, the response unit which caused the incorrect decision is also modified, the connections from active property filters being incremented by $-\frac{\delta_i}{m}$. δ_i here is 1 for a positive class training pattern, and -1 for a negative class training pattern.

For training purposes, the weights were allowed to vary as freely as with the earlier Error Correction and MADALINE systems, but these weights are not considered to be the actual system weights. The actual system weights are taken to be the average values of these freely varying weights over the last full cycle through the training patterns.

The technique was applied to the DAID quadratic property filters for Tasks NVC, PVS, and RVR (at the 50 by 50 aperture). Four, six, and eight response units, or linear decision elements, were tested. The generalization results are given below. These figures are not significantly different from earlier results. The number of linear decision elements has little effect on the results of these experiments.

Linear Decision Elements	Task NVC	Task PVS	Task RVR
4	87.75	84.00	73.75
6	87.25	85.00	74.00
8	87.50	85.75	73.50

3.3.5.4 Modified Algorithms

One modified error correction, and two modified MADALINE algorithms were considered. The method mentioned above using two sets of output weights was implemented, and the resulting techniques were called averaging error correction and averaging MADALINE. In the two weight method, one set of weights is allowed to vary freely, while the other set, the actual system weights, are the average values of the freely varying weights for one cycle through the sample patterns. These techniques were applied to Tasks NVC, PVS, and RVR (50 x 50), using the DAID units.

The other modified MADALINE is called a sequential MADALINE. This technique does not use the averaging process. The original MADALINE algorithm appeared to use a large initial segment of the design run in allocating portions of the recognition task to the various ADALINES. In sequential MADALINE, the algorithm is run with 1 ADALINE for N cycles through the training patterns. After each cycle, the performance level is compared with previous performance level, and if it is better, the machine state is recorded. After N cycles, the machine state corresponding to the best performance level is used as initial weights for one ADALINE of a three ADALINE system. After N cycles, the best three unit system is used as a starting state for a five unit system, and so on. In the experimental work, N was taken to be 10. The purpose of this change is, of course, to address the first ADALINE to the largest part of the problem, and subsequent ADALINES to increasingly finer parts of the discrimination. This technique was applied to tasks NVC and PVS, with seven ADALINES as a limit.

The generalization results achieved by the modified algorithms are given in the table below. As with the piecewise-linear technique, the results do not appear to be significantly different from earlier performance figures.

Technique	Linear Decision Elements	Task NVC	Task PVS	Task RVR (50 x 50)
Averaging Error Correction	1	87.75	83.75	74.00
Averaging MADALINE	3	88.00	85.00	75.00
	5	88.00	85.00	73.75
	7	87.50	85.50	75.25
Sequential MADALINE	1	86.00	83.50	—
	3	87.75	85.75	—
	5	87.00	85.25	—
	7	87.25	86.50	—

The modifications did provide some good effects, however. Performance with the averaging algorithms did not oscillate strongly as a function of the number of training cycles, when compared with the unmodified algorithms, and the best performance point for all of the modified systems came much earlier in the design runs.

3.3.5.5 Summary

The results achieved using distribution estimation to design the decision function were more or less average, when compared with error correction and iterative design. In view of the complexity of the resulting systems, the application of this technique to binary valued property filters does not seem desirable in its current state of development.

The results obtained with the piecewise-linear technique and the three modified algorithms are summarized in Table XI. For comparison purposes, the results achieved earlier with the unmodified MADALINE are included. This MADALINE has been the best technique on task RVR; it and iterative design were best on task PVS; it was second to iterative design by a quarter of a percent on task NVC. On task NVC, all of the systems exhibit a small but consistent improvement over earlier results. Elsewhere, only the sequential MADALINE registered any gains, and those were small.

TABLE XI

GENERALIZATION PERFORMANCE WITH NEW ALGORITHMS

Technique	Linear Decision Elements	Task NVC	Task PVS	Task RVR (50 x 50)
Averaging Error Correction	1	87.75	83.75	74.00
Averaging MADALINE	3	88.00	85.00	75.00
	5	88.00	85.00	73.75
	7	87.50	85.50	75.25
Sequential MADALINE	1	86.00	83.50	–
	3	87.75	85.75	–
	5	87.00	85.25	–
	7	87.25	86.50	–
Piecewise-Linear	4	87.75	84.00	73.75
	6	87.25	85.00	74.00
	8	87.50	85.75	73.50
Comparison:	3	86.00	85.50	–
Unmodified MADALINE	7	–	–	75.75

In summary, a number of new and modified algorithms for the design of the decision functions were tested on the same property filter sets as were the earlier algorithms. The primary objective of these experiments was to improve performance levels, and in this they failed to yield the hoped for gains.

The modified algorithms did display some operational improvements. Design times were shortened. For example, the Sequential MADALINE design runs used 7 hours of SDS 930 time, compared with 12 to 16 hours used by the earlier MADALINE, with no sacrifice in performance. The Averaging MADALINE and Error Correction did not display the large performance fluctuations during the design run which were characteristic of the unmodified algorithms. The smoother performance curve gives the designer greater confidence in selecting a stopping point for the design process.

A number of nonlinear decision functions were tested. It is felt that the failure of these techniques to consistently outperform the linear techniques is due to the fact that the property filters were selected to permit linear separability of the design patterns.

3.3.6 Statistical Property Filters

3.3.6.1 Introduction

The results presented in Section 3.3.5 indicate that the decision mechanism algorithms are probably as effective as the set of statistical property filters permits. An improved means for designing the set of statistical properties thus seems desirable.

In the preceding work, the statistical property filters were designed using the DAID or the QSID algorithms. In these, the set of property filters is selected sequentially. Each time the existing set is to be augmented, a pool of candidate filters is generated. This is accomplished by randomly selecting a number of subspaces. For each subspace, a switching surface is defined by the vectors X satisfying the quadratic equation

$$X^T(\Sigma_1^{-1} - \Sigma_2^{-1})X - 2X^T(\Sigma_1^{-1}M_1 - \Sigma_2^{-1}M_2) + M_1^T\Sigma_1^{-1}M_1 - M_2^T\Sigma_2 M_2 + \ln\frac{|\Sigma_1|}{|\Sigma_2|} = 0$$

M_1 and M_2 are the mean vectors, and Σ_1 and Σ_2 the covariance matrices of the two pattern class. If each pattern class had a Gaussian distribution in the subspace, and if M_1, M_2, Σ_1, and Σ_2 were the true parameters of these distributions, this equation would provide an optimum switching surface. When M_1, M_2, Σ_1, and Σ_2 are unknown, they are customarily estimated from sample patterns (as in the DAID and QSID algorithms). This quadratic analysis is widely used, even when it is known that the underlying distributions are not Gaussian (for example, see Reference 129). The random selection of subspaces and the assumption of Gaussian statistics can lead to property filters of low efficiency. To mitigate this, many more property filters are generated than are included in the property filter set. Selections are made on pragmatic grounds, based on how well the candidates augment the existing system in separating the sample patterns.

A different technique for generating statistical property filters was tested. This method avoids the two least desirable aspects of the earlier techniques — the random selection of subspaces, and the assumption of Gaussian statistics. Selection of coordinates for a subspace was accomplished sequentially, using a mutual information value as a selection criterion. Within a subspace, a switching surface was defined using a nonparametric distribution estimation technique.

Tests were performed on the cloud pattern tasks. The textural nature of these patterns permitted a number of subspaces to be formed by translations of a subspace generated by the mutual information process. Sets of 9 and 25 subspaces, translations of a single select subspace, were used. The decision mechanism used in these systems was quite crude — consisting of an unweighted majority vote of the property filters.

3.3.6.2 Subspace Selection by Mutual Information

A program was developed to sequentially select the coordinates of a subspace. Mutual information values were used to control the selection process. To establish the mutual information, two partitions of the sample patterns in the training set were required. Let $p(i, j)$ denote the fraction of patterns in the i-th cell of the first partition and the j-th cell of the second partition, $p_1(i)$ denote the fraction in the i-th cell of the first

partition, and $p_2(j)$ denote the fraction of patterns in the j-th cell of the second partition. Then the mutual information is given by:

$$\sum_i \sum_j p(i,j) \log_2 \frac{p(i,j)}{p_1(i)p_2(j)}$$

One of these partitions was provided by the actual classification of the sample patterns. The other partition chosen was an attempt to reflect the suitability of the subspace already selected, if augmented by a candidate coordinate. In this application, the value of the mutual information was between "zero" and "one." If $p(i,j) = p_1(i)p_2(j)$ for all i and j, that is, the actual pattern classifications were statistically independent of the second partition, the value would be "zero." If p(i/j) were always "zero" or "one," that is, the second partition gives complete information about the actual pattern classification, the value of the mutual information is given by $-\Sigma p_1(i) \log_2 p_1(i)$. Since two classes containing equal numbers of sample patterns were used, this value is "one."

The selection of the second partition was difficult, and the final choice represented several compromises with computational reasonableness.

Ideally, a property filter would be designed using distribution estimation for each subspace formed by augmenting the subspace already chosen by a candidate coordinate. The decisions of these property filters would then form the second partitions. To avoid the excessive computer time required to accomplish this, it was decided to design one property filter on the subspace already selected, and to refine the property filter's partition using the discrete gray scales of the candidate coordinates. As a second compromise with memory size, only the most significant two bits of the three bit gray scale were used. Thus the second partition has eight cells, formed from the one bit decision of the logic unit and the two bit gray scale of the candidate coordinate. As a final simplification, and perhaps the most weakening compromise of all, quadratic property filters were used to minimize computer time.

Coordinates for the subspace are selected from a 25 by 25 subaperture of the 75 by 75 NIMBUS patterns. It was felt

that the subaperture would be sufficiently large to contain the significant pattern features. The NIMBUS patterns were divided into nine subapertures. Due to the textural form of these cloud structures, the patterns within the nine subapertures should be of the same nature. All nine subapertures were used, so that the sets of training patterns contained 9000 examples of each class.

Once a subspace was selected, it was expanded to a set of 25 subspaces by translation. Consider the subspace to be in the central subaperture. Horizontal and vertical translations of 0, plus and minus 10, and plus and minus 25 raster elements were used. The subspace positioned within each of the original nine subapertures is included in the resulting set of 25 subspaces. A set of nine subspaces was also formed by using only the 0 and plus and minus 25 translations (corresponding to the original subapertures).

Results on task NVC were sufficiently poor with the nine subspace case, that the 25 subspace case for the selected subspace was not tested. As controls, two random subspaces in the 25 by 25 subaperture were selected for Task PVS and one for Task NVC, and sets of 25 translations formed as above. For task PVS performance achieved by applying distribution estimation to these random subspaces were almost as good as the performance of the same technique on the mutual information subspace. The results on Task NVC were poor. For this task, a subspace consisting of the eighth through seventeenth coordinates selected by the mutual information program was processed using 25 translations, and again the generalization performance was around 70 percent. Thus these tests have not demonstrated that these mutual information subspaces offer any advantage over randomly selected subspaces.

3.3.6.3 Distribution Estimation

A nonparametric distribution estimation technique was used to establish the switching surfaces for the subspaces. In particular, the binary property filter is defined as on for a test pattern if the density estimate (likelihood function) for the positive class is larger than that for the negative class at the test pattern vector. The problem of accurately recovering the unknown underlying distribution (or density) associated with a collection of samples has received considerable attention. E. Parzen[158]

has treated a general class of consistent estimators for the one-dimensional case. Most of his results have been extended to the multi-dimensional case by V. K. Murthy.[159] The mathematical results can be described very briefly. Let $\{x^k\}_{k=1}^M$ be a set of M identically distributed N-dimensional random vectors. The empirical distribution function F_M is defined by the expression

$$F_M(x_1, x_2, \ldots, x_N) \equiv \frac{1}{M} \left\{ \begin{array}{l} \text{number of observations } x^k \\ \text{such that } x_j^k \leq x_j,\ j=1,2,\ldots,N \end{array} \right\} \tag{1}$$

An estimator f_M for the N-variate density f is given by

$$f_M(x) \equiv \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \prod_{n=1}^{N} b_n \cdot H\{b_1(x_1 - y_1), \ldots, b_N(x_N - y_N)\}\, dF_M(y_1, \ldots y_N) \tag{2}$$

The function H satisfies the conditions

$$\int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} H(x_1, x_2, \ldots x_N)\, dx_1 \ldots dx_N = 1,$$

$$H(x_1, x_2, \ldots x_N) = H(\pm x_1, \pm x_2, \ldots, \pm x_N) \geq 0,$$

$$\lim_{\|x\| \to \infty} \{\|x\|\, H(x_1, x_2, \ldots, x_N)\} = 0, \text{ where } \|x\| \equiv \left\{ \sum_{n=1}^{N} x_n^2 \right\}^{1/2}$$

and $b_n > 0$ for $n = 1, 2, \ldots, N$. For the applications of interest it is also assumed that the density f of the underlying distribution F is everywhere continuous.

It can easily be shown that Equation (2) is equivalent to

$$f_M(x) = \frac{1}{M} \left\{ \prod_{n=1}^{N} b_n \right\} \sum_{k=1}^{M} H\{b_i(x_i - x_1^k), \ldots, b_N(x_N - x_N^k)\} \tag{3}$$

V. K. Murthy,[159] in extending E. Parzen's results for the one-dimensional case, has shown that if the b_n are functions of the sample size M and satisfy the conditions

$$\text{(a)} \quad \lim_{M\to\infty} b_n = \lim_{M\to\infty} b_n(M) = \infty \text{ for } n = 1, 2, \ldots, N$$

$$\text{(b)} \quad \lim_{M\to\infty} \frac{M}{\prod_{n=1}^{N} b_n} = \infty$$

then f_M is a consistent estimate of f at every point x. That is, if conditions (a) and (b) are satisfied then at every point x

$$\lim_{M\to\infty} E\{f_M(x)\} = f(x)$$

and

$$\lim_{M\to\infty} \text{var}\{f_M(x)\} = 0$$

The specific case under investigation involves

$$H(x) = \frac{1}{\pi^{N/2}} \prod_{n=1}^{N} \exp\{-x_n^2\}$$

so that Equation (3) becomes

$$f_M(x) \frac{1}{M\pi^{N/2}} \left\{ \prod_{n=1}^{N} b_n^{1/2} \right\} \sum_{k=1}^{M} \prod_{n=1}^{N} \exp\left\{ -b_n (x_n - x_n^k)^2 \right\} \qquad (4)$$

The notation used in Equation (4) is slightly different from that used in Reference 159 and entails substituting $\sqrt{b_n}$ for b_n in Equations (2), (3) and condition (b). Since in every practical case one deals with a finite number of samples, M, the problem is to obtain for a fixed M an estimate of the values for b_n. It is possible to introduce a property of the unknown underlying distribution (via the sample set $\{x^k\}_{k=1}^{M}$) by investigating an expression of the form

$$\frac{1}{b_n} = \frac{1}{M(M-1)} \sum_{i=1}^{M} \sum_{j=1}^{M} a_{ijn} \exp(-\rho_n a_{ijn}) \tag{5}$$

where

$$\rho_n > 0 \text{ for } n = 1, 2, \ldots, N$$

$$\lim_{M \to \infty} \rho_n = \infty$$

and

$a_{ijn} \equiv (x_n^i - x_n^j)^2$, i and j refer to the i-th and j-th sample vector.

The problem of determining the b_n is thus traded for the problem of determining ρ_n. It can be shown that in order for b_n to satisfy the consistency conditions (a) and (b), ρ_n is of the order of log M (i.e., $\rho_n / \log M \leq k$ for all $M \leq 1$). In obtaining this result the inequality

$$\frac{M}{\prod_{n=1}^{N} b_n^{1/2}} \geq M \prod_{n=1}^{N} \left[\exp\{-\rho_n \mu_n(M) + \log \mu_n(M) \right]^{1/2}$$

is derived, where

$$\mu_n(M) = \frac{1}{M(M-1)} \sum_{i=1}^{M} \sum_{j=1}^{M} a_{ijn}.$$

Choosing

$$\rho_n = \frac{\delta_n}{\mu_n(M)} \left[\log M + \log \{\mu_n(M)\}^{1/\delta_n} \right]$$

yields

$$\frac{M}{\prod_{n=1}^{N} b_n^{1/2}} \geq M \prod_{n=1}^{N} \exp\{-\frac{1}{2} \delta_n \log M\} = M^{1 - \frac{1}{2} \sum_{n=1}^{N} \delta_n}$$

so that the consistency conditions are satisfied with ρ_n as given above if

$$\sum_{n=1}^{N} \delta_n < 2.$$

Furthermore, it can be shown that if ρ_n is of the form

$$\rho_n = g_n M^{\beta_n} + d_m$$

where $\beta_n > 0$ and g_n, d_n are constants such that $\rho_n > 0$ for $M \geq 1$, then the consistency conditions are violated. Thus, in calculating the values of b_n the expression

$$\rho_n = \frac{\delta_n}{\mu_n(M)} \log (M \cdot \mu_n(M)^{\frac{1}{\delta_n}})$$

is used. The behavior of ρ_n (and hence of b_n) is influenced by the quantities $\mu_n(M)$ which yield an indication of the "spread" of the unknown distribution along each of the N axes. Thus for a fixed sample size M a distribution with large second central moments requires smaller values for b_n (and hence smaller values of ρ_n) than a distribution possessing small second central moments. This behavior is exhibited by ρ_n as given above.

Distribution estimates were made separately for each subspace. Thus, although the coordinate configurations of the subspaces in a system are simple translations of one another, each subspace has its own unique switching surface.

Attempts were made to simplify the distribution estimates by finding the local maxima of the density functions. It was hoped that the number of sample points used in the distribution estimation could be reduced by replacing those clustered around a mode by the modal point itself. Even with variations in clustering program parameters, only two modes could be found for the polygonal cell distribution, which would be insufficient to provide a real simplification.

It was hoped that 25 to 100 cluster points would be found, with nearly all of the sample points belonging to some cluster.

Having found such a set of cluster points, one may proceed to simplify the system in one of two ways. One may use only the cluster points to estimate the distributions, or one may use the cluster points plus the sample points not belonging to a cluster to estimate the distribution. When 1000 samples are being used to estimate the distribution, and when one finds two cluster points which represent perhaps a dozen sample points, the second alternative offers a one percent reduction in system complexity. This hardly seems worth the effort. The use of the first alternative with two cluster points implies that the distributions are simple and highly separable. A single quadratic property filter should do nearly as well, since four points are separable by a quadratic surface. Earlier experiments with DAID units indicate that this is not the case.

The subspace selection algorithm was applied to task PVS. The first eight coordinates were selected with no difficulty. The coordinates which were selected were widely separated in the subaperture. For coordinates nine and ten, the program tried to select coordinates already chosen. When this happens, the algorithm chooses the second best candidate, and so on. With these alternate selections, coordinates nine and ten of the subspace followed the trend of widely separated points. For the eleventh and subsequent selections, the program tried to repeat the fourth coordinate, and on being denied this, chose an adjacent point. Performance of the quadratic property filter in classifying the training patterns varied smoothly, reaching a maximum with ten coordinates. At that point it made 71.65 percent correct decisions on the 18,000 subaperture patterns (nine subapertures from each of 2,000 training patterns).

The first ten coordinates selected were used as the subspace. Nine translations were taken and switching surfaces assigned using distribution estimation. Using a majority vote of the nine property filters to determine the system decision, 86.75 percent of the test patterns were correctly identified. This is a quarter of a percent better than sequential MADALINE accomplished with 400 DAID property filters, and at least one percent better than any other system obtained with statistically derived properties.

The same subspace was then used in 25 translations. The system yielded 90.75 percent correct decisions, a substantial increase. The property filters themselves were 73.59 percent correct in 10,000 generalization decisions (25 subapertures from each of 400 generalization patterns). Using one of the randomly generated subspaces, the 25 property filters were 73.17 percent correct, and the majority vote system achieved 90.25 percent. For the second random subspace, the property filters were 73.65 percent correct, and the generalization decisions were 89.75 percent correct. Thus the subspace selection process did not add much to the system.

The subspace selection program was then run to choose a ten coordinate subspace for task NVC. Repetition of previous selections began with the fifth coordinate and continued thereafter. The best quadratic property filter classification of the sample patterns occurred with five coordinates, and was 60.67 percent. The ten coordinate quadratic unit gave 59.71 percent. The subspace selected was a rather strange one. The first nine choices came from the left half of the top row of the 25 by 25 raster. The tenth coordinate was about halfway down the last column.

Nine translations were taken, and a distribution estimation system tested. It achieved 66.5 percent on the test patterns — 84 percent of the noncumulus patterns, 30 percent of the solid cell patterns, and 68 percent of the polygonal cell patterns were correctly identified. Contributing to this uneveness might be the fact that equal numbers of cumulus and noncumulus patterns were used to estimate the distributions. There were, therefore, twice as many noncumulus patterns as there were solid cell or polygonal cell patterns. Solid cell patterns appear more similar to noncumulus than to polygonal cell patterns.

Due to the poor overall results, the 25 translation experiment was not run. The unusual subspace may have contributed to these results. The subspace selection program was continued to 23 coordinates. Except for two coordinates near the center of the subaperture, the remaining coordinates formed a loose cluster near the lower right corner of the aperture. Best classification with a quadratic property was 67.86 percent

with 17 coordinates, a figure much more in line with that achieved on task PVS. Coordinates 8 through 17 were used as a subspace, and 25 translations taken. The overall performance was 71.25 percent, with 98.5 percent of the noncumulus, 73 percent of the polygonal cell, and 15 percent of the solid cell patterns being correctly identified.

To complete the picture, a random subspace was selected, and 25 translations taken. Overall performance was 69.75 percent – 99 on noncumulus, 67 on polygonal cells, and 14 on solid cells.

3.3.6.4 Optical Experiments

Optical processing[161] offers some attractive features for pattern recognition. Great speed is possible due to the highly parallel nature of the processing. Processing in the spatial frequency plane (see Figure 41) offers freedom from translational variations. In addition, the textural nature of the cloud patterns suggests that a frequency plane analysis might be fruitful.

The two-dimensional Fourier transform of an input image is presented in the frequency plane. By placing a suitably designed transparency or mask in the frequency plane, it is possible to operate on the Fourier transform of the input image with the results displayed in the output plane. Alternately, it is possible to sample the frequency plane and display these results directly. The design of the property filters is then dependent on the spatial frequency content of the training set of data for each pattern class.

To investigate the possibility of designing property filters based upon the spatial frequencies of the patterns, some limited optical experiments were performed. A set of 72 black and white transparencies of the NIMBUS cloud imagery was made on 120 film. The data were selected to provide 24 transparencies for each of the three types of clouds used in previous experiments. These transparencies were then placed in the input plane of the objective lens as shown in Figure 42. A laser beam was used to illuminate the cloud pattern transparency. A microscope, with its input lens placed in the frequency plane, allowed the resulting spectrum of the image to be enlarged to ease the sampling task. A photomultiplier tube

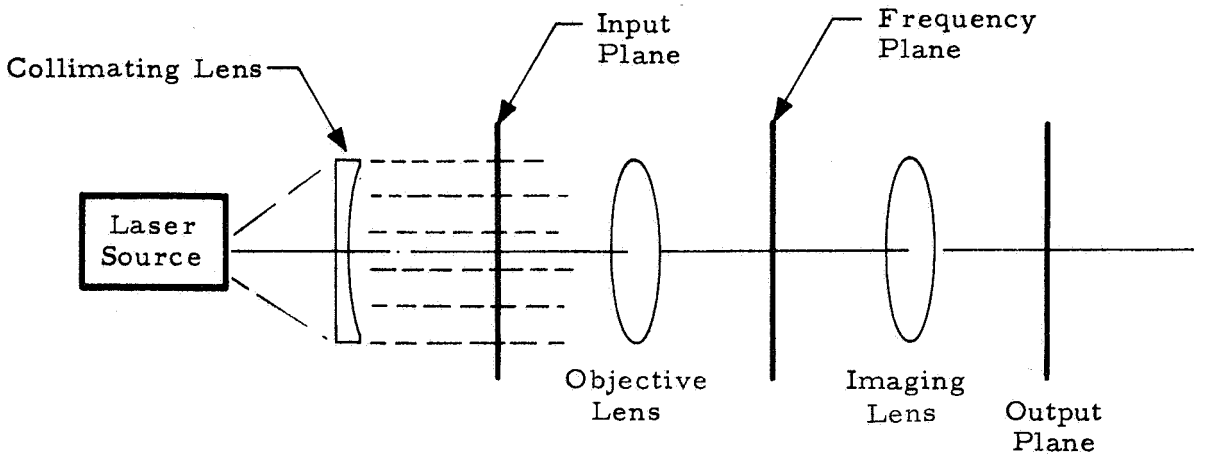


Figure 41. Optical Processing

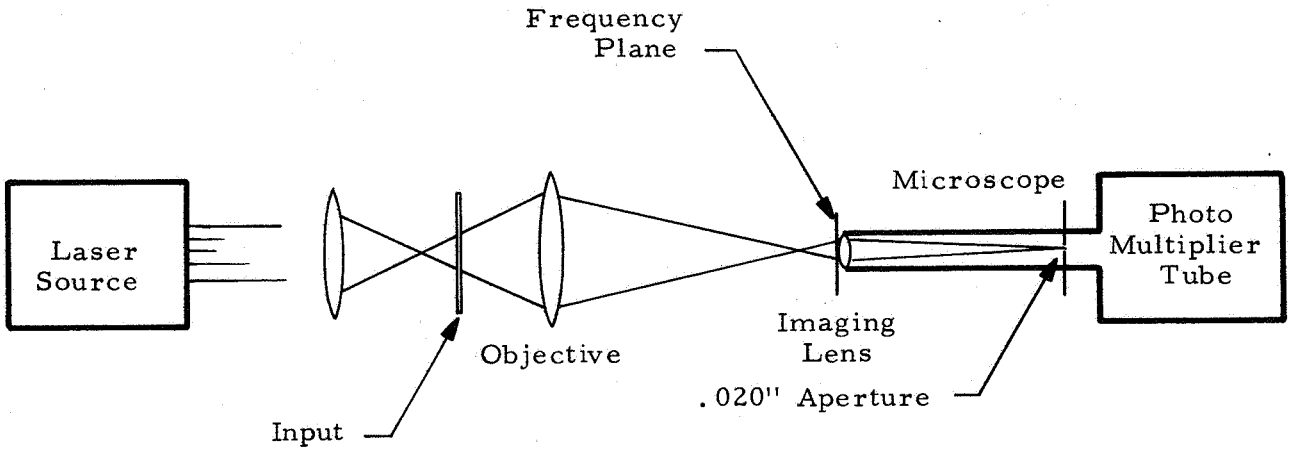


Figure 42. Modified Optical Processing

with 0.020 inch input aperture measured the irradiance in this enlarged frequency plane. The sampling was performed over nine rows and columns of points, each row and column was separated by 0.001 inch, resulting in 81 measurements of the spatial frequencies for each transparency. An example of the measurements made is shown in Figure 43.

The resulting data was then subjected to several analyses including the following:

1. Property filters were generated using the QSID procedure modified to accommodate six inputs per unit and to select one unit at a time from a population of 15 candidate units generated.
2. Property filters were generated using a further modification of this QSID procedure where the covariance matrices of the pattern classes were assumed to be equal, resulting in the specification of a linear discriminant for each filter.

As a result of the generation of these property filters, the patterns were actually classified by the QSID or the LSID (linear surface) procedures. Despite the small number of patterns represented, it is of interest to note (Table XII) that complete separability was readily achieved with very few property filters.

TABLE XII

CLASSIFICATION RESULTS USING OPTICAL PREPROCESSING

Technique	Task	Number of Patterns in Each Class	Number of Units Required for Complete Separation
QSID	PVS	24	3
LSID	PVS	24	5
QSID	NVC	24	2
LSID	NVC	24	3

Each property filter was then examined to attempt to determine what information, available in the spatial frequency domain, was selected. The LSID units were of particular interest since the linear weights

1 -32	-30	-23	-17	-12	-24	-29	-31	-32
10 -31	-30	-20	-1	16	-11	-27	-30	-31
19 -30	-31	17	23	26	12	-22	-30	-32
28 -30	-19	22	26	28	17	-21	-29	-32
37 -28	-16	21	25	27	10	-22	-30	-32
46 -28	-14	18	24	24	-2	-26	-30	-32
55 -27	-17	6	16	12	-18	-28	-31	-32
64 -29	-23	-7	2	-4	-25	-30	-32	-32
73 -31	-26	-22	-18	-22	-29	-32	-32	-32

C3399

Figure 43. Sampled Frequency Plane (Irradiance)

indicated a direct relationship between input signal and filter response. Tables XIII and XIV present the input connections, the weight assigned, and the frequency of usage of each connection. For the NVC task only 16 different inputs were used, while for the PVS task, which required two more property filters, 28 inputs were used. Only five of these input connections were selected for both tasks. The scattering of the connections across the array of inputs showed a trend toward selecting more outer elements for cumulo-polygonal and cumulo-solid cell classifications which may be interpreted as a tendency to select, for discrimination, the higher frequency content present in these patterns. No attempt was made to physically implement or to simulate these filters to attempt to perform a generalization experiment due to the difficulty in obtaining an adequate sample for comparative evaluation.

3.3.6.5 Summary

Summarizing these results, a technique for property filter generalization using sequential subspace selection and nonparametric distribution estimation was tested. On task PVS a substantial performance decline was experienced. In both cases, the subspace selection process appears weak. The results are probably due to the smaller representation of the solid cell and polygonal cell subclasses in Task NVC, and to the greater gray scale distribution differences between the polygonal cells and the solid cells or noncumulus patterns than between the solid cells and the noncumulus patterns.

TABLE XIII

PROPERTY FILTERS (NVC)

Logic Unit	LSID Connection	Weights
1	78	-.458E 00
	45	-.593E -01
	30	.179E 00
	44	-.399E 00
	70	.155E 00
	58	.546E 00
2	78	-.117E 01
	81	.143E 01
	9	.626E 01
	8	-.859E 01
	28	.244E 01
	71	.667E 00
3	41	.251E 01
	62	.536E 00
	17	-.228E 01
	40	-.282E 01
	74	-.693E 00
	28	.180E 01

Connection	Frequency of Usage
8	1
9	1
17	1
28	2
30	1
40	1
41	1
44	1
45	1
58	1
62	1
70	1
71	1
74	1
78	2
81	1

TABLE XIV

PROPERTY FILTERS (PVS)

Logic Unit	LSID Connection	Weights
1	13	-.779E -03
	67	.204E 00
	81	-.276E 01
	48	-.273E 00
	9	.697E 00
	59	-.355E 00
2	29	.352E 00
	55	-.606E 00
	66	.543E 00
	36	-.480E 00
	68	-.755E -01
	74	-.126E 01
3	61	-.751E 00
	62	.204E 01
	6	.341E 00
	7	-.122E 01
	69	.162E -01
	1	-.105E 01
4	26	.234E 00
	3	.682E 00
	12	-.291E 00
	45	-.463E 00
	63	.147E 01
	80	-.318E 01
5	4	-.223E 00
	60	.722E -01
	12	.321E 00
	35	.593E -01
	30	-.286E 00
	36	.119E 00

Connection	Frequency of Usage
1	1
3	1
4	1
6	1
7	1
9	1
12	2
13	1
26	1
29	1
30	1
35	1
36	2
45	1
48	1
55	1
59	1
60	1
61	1
62	1
63	1
66	1
67	1
68	1
69	1
74	1
80	1
81	1

3.3.7 Augmentation of Known Properties

3.3.7.1 Introduction

Figure 44 shows the philosophy for designing a recognition system. "Signal Conditioning" provides both formatting of the input patterns and (where possible) invariance to irrelevant pattern differences such as gray-scale changes, translations, etc. The "Known Properties" are included to exploit the designer's knowledge of the recognition task. They measure pattern characteristics which the designer feels are useful for classifying the patterns. If the known properties are sufficient for the "Decision Mechanism" being used, then the design effort is readily completed. If they are not, however, more property filters must be obtained. One way in which the additional property filters may be designed is by the statistical analysis of sample patterns (i.e. "Statistical Property" extraction).

The augmentation of a set of know property filters with a set of statistical property filters specifically designed to complement the known properties is a virtually untouched problem. Eight augmentation experiments are reported here. Augmentation was tried on both NIMBUS tasks (Tasks PvS and NvC), as well as all three lunar feature tasks (CvC, RvR, and CvR) at the reduced aperture sizes of 25 by 25 and 15 by 15.

3.3.7.2 Augmentation Technique

The method of augmentation was as follows. A linear decision mechanism for the known properties was first designed. For this purpose the iterative design algorithm (modified to process the continuous outputs of the known property filters) was used. This approach was used for three reasons (1) design times are short, (2) the designs are optimized in terms of the pattern losses used in the rest of the process, and (3) compensation for variation of means and variances between property filters is automatic.*

*In some other experiments using continuous data an error correction process was used. The data had to be normalized prior to the application of the algorithm. If this normalization had not been made, four million cycles through the patterns would have been necessary for the threshold to compensate for the means of the property filter values.

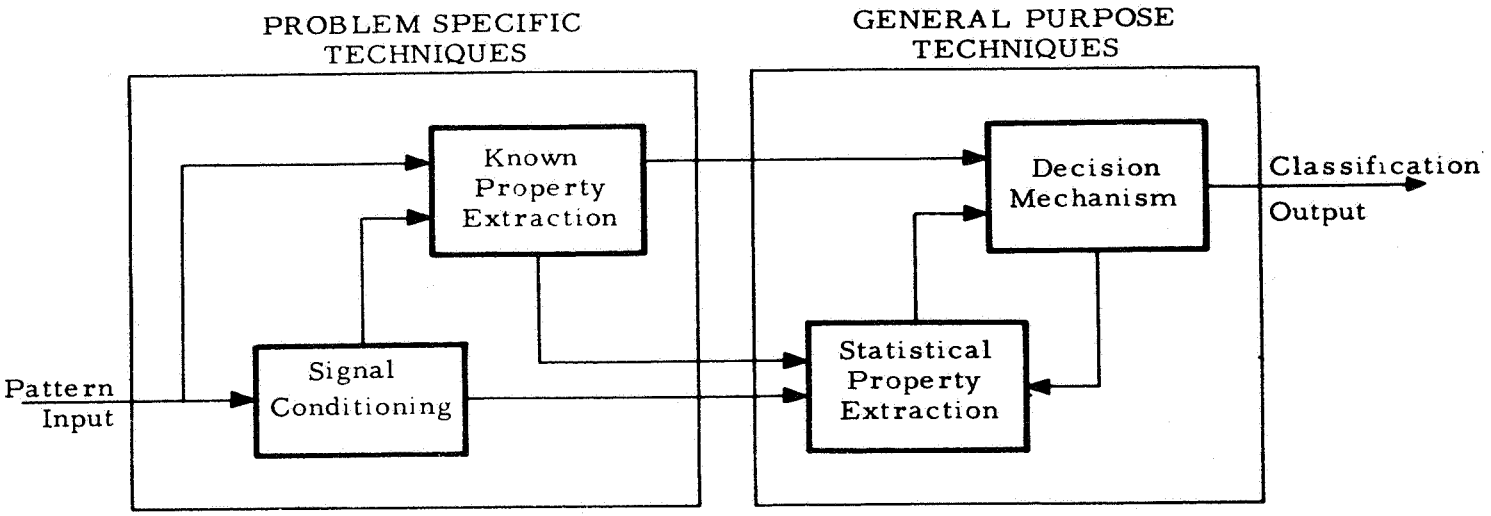


Figure 44. Pattern Recognition Mechanism

After this design was complete, each sample pattern in the training set was assigned a loss value. If D_i was the value of the linear decision function for the i-th pattern, and δ_i plus or minus one, depending on the true classification of that pattern, the loss for that pattern was given by

$$L_i = \exp \{ -\delta_i D_i \}$$

This portion of the system design was held fixed for the rest of the process.

The losses for the training samples were used as initial values for the sample pattern losses in the QSID program. This program selects statistically designed property filters to minimize the system loss. Thus the statistical property filters are selected to complement the know property filters.

After the set of augmenting statistical property filters was selected, the decision mechanism generated by the QSID program was discarded. That portion of the decision mechanism was redesigned by applying the iterative design algorithm, again using initial values for losses of the training patterns. Performance on the sample patterns was determined by adding the decision functions for the known and statistical property sets. Again, the use of the iterative design algorithm made the equal weighting of these two decision functions automatic.

3.3.7.3 Known Properties

A set of 29 known properties was designed by the NASA Technical Officer for this program, Mr. Eugene M. Darling, Jr. These properties, listed in Table XV, were specifically designed for recognizing cloud patterns. Systems using these known properties achieved 90.0 and 97.0 percent generalization performances on Tasks NVC and PVS, respectively. Since these known properties alone achieved excellent performance, there is little room left for improvement by augmentation.

Fifteen of these 29 known properties relate to general characteristics of a two dimensional brightness field. The remaining properties are specifically tailored to the cloud recognition problem.

TABLE XV

KNOWN PROPERTIES

1.	Mean Brightness		15.	Mean Cloud Segment	
2.	Brightness Variance		16.	Number of Clouds	
3.	Relative Frequency of Each Gray Level	0	17.	Mean Cloud Size	
4.		1	18.	Variance Cloud Size	
5.		2	19.	Relative Frequency of Cloud Size (i.e. number of elements in a 75 x 75 element aperture)	1-25
6.		3	20.		26-100
7.		4	21.		101-225
8.		5	22.		226-400
9.		6	23.		401-900
10.		7	24.		901-1600
11.	Information in the X-Direction (adjacent points)		25.		1601-2500
12.	Information in the Y-Direction		26.		2501-3600
13.	Mean Gray Level Area (connected regions of constant gray level)		27.		3601-4900
14.	Variance, Gray Level Area		28.		4901-5625
			29.	.80 contour area (auto-correlation function)	

The general properties are numbers 1 through 14, and 29. These 15 properties were also used in the lunar terrain recognition experiments. It was anticipated that the use of a small set of known properties, none of which was designed for the lunar problem, would result in lower performance levels more amenable to improvement by augmentation than was the case with the cloud pattern experiments.

Most of the entries in Table XV are self explanatory. The following definitions are offered for those which are not.

Information in the X and Y Directions

Let $P(I)$ be the unconditional probability of brightness I, $0 \leq I \leq 7$, calculated from a particular pattern.

$P(J/I)$ is the probability of brightness J, given that the brightness is I at the adjacent point.

$H_q(J/I)$ is the conditional information of J given I where q is either X or Y.

$$H_q(J/I) = \sum_{I=0}^{7} P(I) \sum_{J=0}^{7} P(J/I) \log P(J/I)$$

This is the equation defining properties 11 and 12

Mean Grey Level Area

The average size of connected areas of constant brightness, I for $0 \leq I \leq 7$.

Variance, Grey Level Area

The variance of size for connected areas of constant brightness, I for $0 \leq I \leq 7$.

.80 Contour Area

The area bounded by the y-axis and 0.80 auto-correlation contour.

3.3.7.4 Augmented Systems

The process for augmenting known properties was applied to eight recognition tasks. Table XVI presents the results obtained both with the known properties alone, and with known properties augmented by statistical properties.

TABLE XVI

GENERALIZATION PERFORMANCES

Task	Aperture	Known Properties	Known Properties plus Augmenting Statistical Properties	Percentage Decrease in Error Rate	Probability of Chance Occurrence
NVC	75 x 75	90.00	92.75	27.5	.083
PVS	75 x 75	97.00	94.25	-91.7	.029
CVR	25 x 25	92.00	97.00	62.5	.0009
	15 x 15	81.50	96.00	78.4	$<10^{-10}$
RVR	25 x 25	78.50	80.00	7.0	.300
	15 x 15	81.00	82.75	9.2	.260
CVC	25 x 25	71.50	85.75	50.0	.00003
	15 x 15	96.00	96.25	6.2	.426

Although results for networks using statistical properties only (i.e., no known properties) were available for most of these recognition tasks, these results were not included in Table XVI. A comparison between the performances of a statistical property system and an augmented known property system does not evaluate the augmentation process. The computer program which derived the augmenting properties is the same one that derived the original statistical properties. The only difference consisted of providing initial values for the pattern losses to insure that the augmenting statistical properties would not try to duplicate the efforts of the known properties. Thus, if the performance of the augmented systems is higher (or equal to) that of the statistical property systems, one concludes that the known properties are (or are not) measuring significant pattern features not accessible to the quadratic statistical property filters. This comparison evaluates the utility of the known properties in the augmented system. As the purpose of these experiments was to evaluate the augmentation process, statistical property systems should not be compared to augmented systems.

In assessing the performance differences shown in Table XVI, two factors should be given consideration - (1) an objective evaluation that such a difference might occur by chance, due to the random selection of the sample test patterns, and (2) a subjective evaluation of whether or not the difference matters. To aid in this latter consideration, the column "Percentage Decrease in Error Rate" is given in Table XVI. The column headed "Probability of Chance Occurrence" provides an indication of the probability that the difference occurred by chance. The method used in its computation is given below.

It is desired to test the hypothesis that the system (known property system and augmented known property system) are equal, against the (one sided) alternative that the augmented system is better. Assume that the pattern set (consisting of 400 patterns) used to test one system was selected independently of the pattern set used to test the other system, and that each system has the same error rate, r. Let r_K and r_A be the observed error rates for the known property system and the augmented system, respectively. Then the variance of $r_K - r_A$ is $r(1 - r)/200$. Under the hypothesis,

$$\alpha = \frac{\sqrt{200}\,(r_K - r_A)}{\sqrt{r(1 - r)}}$$

will have zero mean and unit variance; and will be very nearly normally distributed. To obtain the "Significance Level," r was estimated by $\frac{1}{2}(r_K + r_A)$, and a table of the cumulative normal distribution was consulted to determine the probability that a difference greater than or equal to observed difference would occur by chance. The normal approximation should be very accurate for moderate probabilities, and indicative of very small probabilities for the extreme values. For task PvS, the one case in which r_K was less than r_A, the "Probability of Chance Occurrence" is the probability that a difference greater than the observed difference $r_A - r_K$ would occur by chance.

This analysis is conservative. The test patterns actually used were the same for each system. One would therefore actually expect a smaller performance difference than with two independent sets of test patterns. To see the extent of this, suppose that associated with the i-th test pattern was a parameter, p_i. Let p_i be the probability that each system misclassifies the i-th pattern (i. e., the systems make independent classifications, each having a probability of $1 - p_i$ of being correct). Let p_i have a Beta distribution with parameters $\frac{r}{1 - r}$ and 1, over the possible set of test patterns. Then the expected error rate is r, and the variance of $r_K - r_A$ is $r(1 - r)^2/(200(2 - r))$. For task NVC, this analysis would give a "Probability" of .022, compared to the value of .083 given by the conservative analysis. The choice of parameters for the Beta distribution is arbitrary, and critically affects the results. For example, parameters 1 and $\frac{1 - r}{r}$ also give an expected error rate of r, but yield .00003 as the Probability of Chance Occurrence.

3.3.7.5 Summary

Summarizing the results of the augmentation studies, eight tests were made. On task PvS a significant performance decrease was encountered, and on task CVC at the 15 by 15 aperture an insignificant performance change occurred. These failures are not disappointing, as performances with the known properties alone are 97 and 96 percent respectively. On task CVR at both apertures, and on task CVC at 25 by 25,

very substantial reductions in the error rates accompanied the augmentation. The likelihood that these improvements occurred by chance is quite remote. Task NVC provides a borderline case. The reduction in the error rate is only one-third to one-half as large as those achieved in the better cases, and there is one chance in twelve that the improvement might occur by chance. Task RvR at both apertures provided the most disappointing results. The reductions in the error rates were relatively small, and quite likely to have occurred by chance.

In at least half of the test cases on which augmentation would be important to a designer, substantial performance increases were achieved. Thus, the augmentation process appears to be a valuable one, particularly when it is most appropriate — when the known properties alone do not produce very high performance levels.

3.3.8 Pattern Boundary Delineation

3.3.8.1 Introduction

The ultimate objective of the NIMBUS data recognition systems is to provide the information needed to draw a cloud map. To accomplish this, it is necessary to delineate the boundaries between homogeneous regions of cloud textures. Because the boundaries between cloud pattern types are not usually sharply drawn, it is expected that operational accuracy requirements would not be too high.

A system's ability to meet this standard would depend, first and foremost, upon its generalization capabilities. A system which had a high rate of errors would give rise to many spurious boundaries. With a very low rate of errors, occasional mistakes may be removed by smoothing.

Assuming a system of high generalization capability, there are several factors that influence the accuracy with which boundaries may be delineated. One of these is the extent of overlap between adjacent sampling apertures. A large degree of overlap might be desirable from the standpoint of accuracy. However, the number of apertures which must be processed for a given coverage varies inversely as the square of the distance between adjacent aperture centers. Thus, concerning the amount of spaceborne processing to be performed, and the amount of data to be transmitted, the degree of overlap should be minimized.

The amount of overlap required to yield a given accuracy of boundary delineation depends on two factors. First, there is the performance of the system when the boundary is present in the aperture, that is, when clouds of two different types are contained in the aperture. The second factor is the accuracy with which a boundary can be interpolated between a pair of transition apertures. If the recognition system's decision is the only quantity considered, then the boundary must be placed halfway between the centers of the transition apertures. But if the magnitudes of the inputs to the response units are considered, the possibility of improving the accuracy of boundary delineation by interpolation exists. It is these two factors, the effect of a boundary in the aperture and the accuracy of interpolation, which form the subject of the experiments considered in this section.

3.3.8.2 Approach

To investigate the boundary delineation problem, patterns with artificially induced boundaries were generated. This was accomplished by creating patterns which are a montage of two patterns selected from the generalization files. The recognition systems were then tested on the montage patterns.

For these experiments, new sets of quadratic property filters were designed for Tasks NVC and PVS, using the QSID program. The decision function was supplied by the iterative design algorithm. These new recognition systems did not give better performance than the earlier DAID designs, but needed considerably less than the 400 property filters of the earlier systems to obtain the same performance.

The QSID program required 190 and 115 property filters for 100 percent classification on Tasks NVC and PVS respectively. When the decision function was reassigned by iterative design, these numbers were reduced to 150 and 90 property filters, respectively. The best generalization performances came at 100 and 50 property filters, respectively. These latter numbers were selected for the boundary

Task	QSID Property Filters	Classification	Generalization
NVC	100	94.35	86.00
PVS	50	95.30	87.50

delineation investigation. Figure 45 shows the generalization performance as a function of the number of property filters.

The montage patterns were generated by a computer program. This program accepted, as inputs, two patterns and a main direction, and would produce 100 montage patterns as output. The main direction could be right, left, up, or down. If the main direction was right, for example, the first output pattern would be the first pattern of the pair, the second output pattern would be the same, except that the rightmost column would be replaced by the rightmost column of the second input pattern, and so on until all 75 columns had been replaced. Following these first 76 patterns,

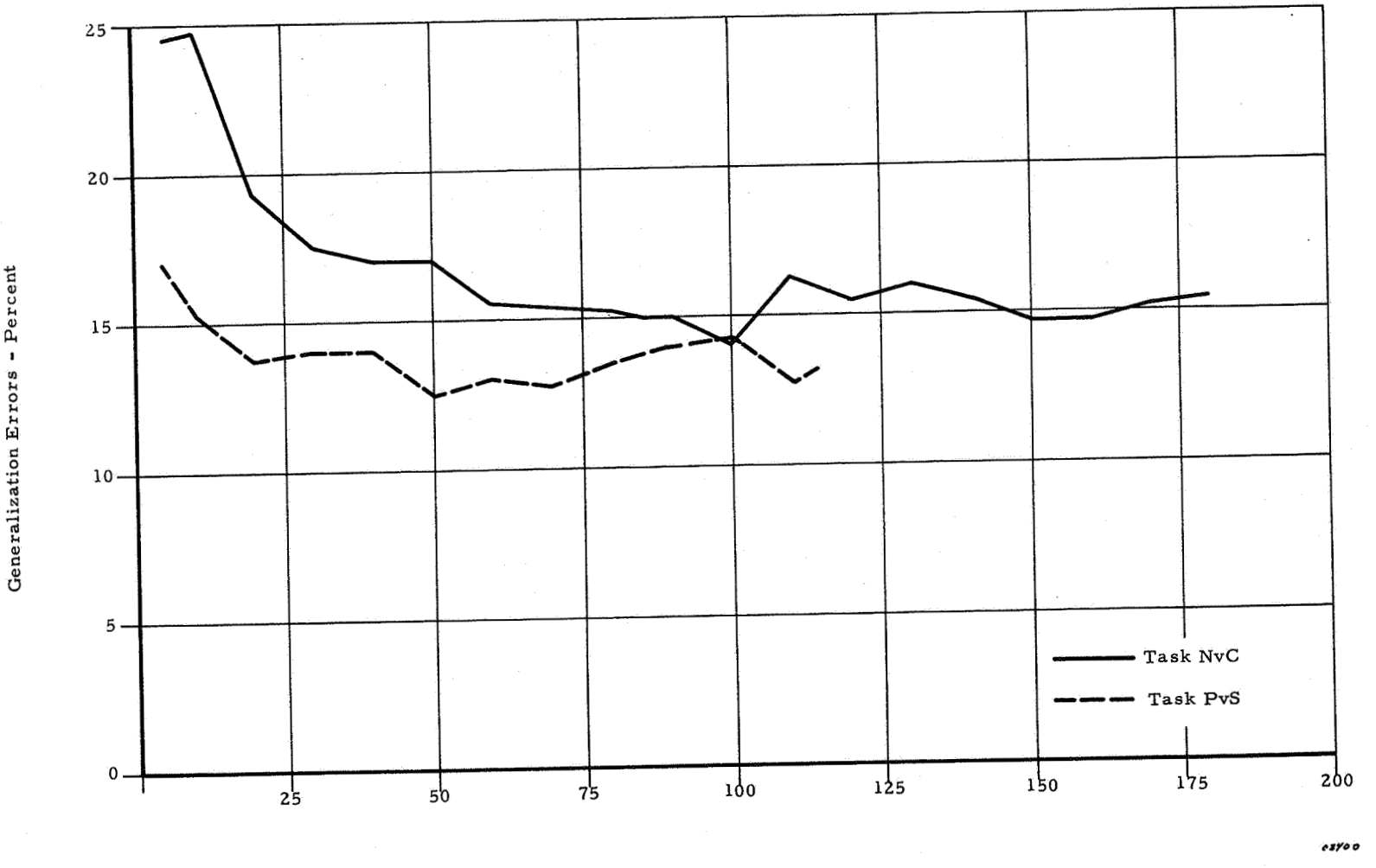


Figure 45. Number of Property Filters

eight selected montages from each of the other three directions were produced. Data from these latter 24 montages were not examined. Figures 46 through 49 illustrate the first, twenty-sixth, fifty first, and seventy-sixth patterns in a montage sequence.

A total of 144 different pairings of patterns were considered. Usually, all four directions were considered for each pairing, as early experiments indicated that different directions for a single pairing produced results as diverse as different pairings would. In all, 464 combinations of pairing and direction were run, producing 46,400 montage patterns.

For the first 76 patterns in each montage sequence, the computer produced a plot of the input to the recognition system's response unit. Figure 50 shows this plot for the montage sequence illustrated in Figures 46 through 49.

Ideally, the recognition system would compute a discriminant (input to the response unit) of a fixed magnitude whenever a homogeneous pattern is present in the aperture. Again, ideally, the computer plot for a montage sequence would appear as a straight line. Under these conditions, the extent of the montage could be determined from the value of the discriminant of a single pattern. Unfortunately, the original patterns are not that well defined, and a high percentage are not pure prototypes. In addition, the recognition system does not produce a constant magnitude output. The straight line plot, however, remains desirable. With this, one may interpolate linearly to position the boundary.

Test patterns giving rise to incorrect decisions were not included in this study. The reamining patterns were classified as low (discriminant magnitudes between 0 and 4) or high (discriminant magnitudes between 4 and 10). All combinations of high against high, low against low, and high against low and all combinations of pattern classes, including pairs of patterns from the same class, are represented in the 464 montage sequences.

Of the 46,400 discriminant values computed, 11,136 were ignored. These were, for the most part, duplicates of other values

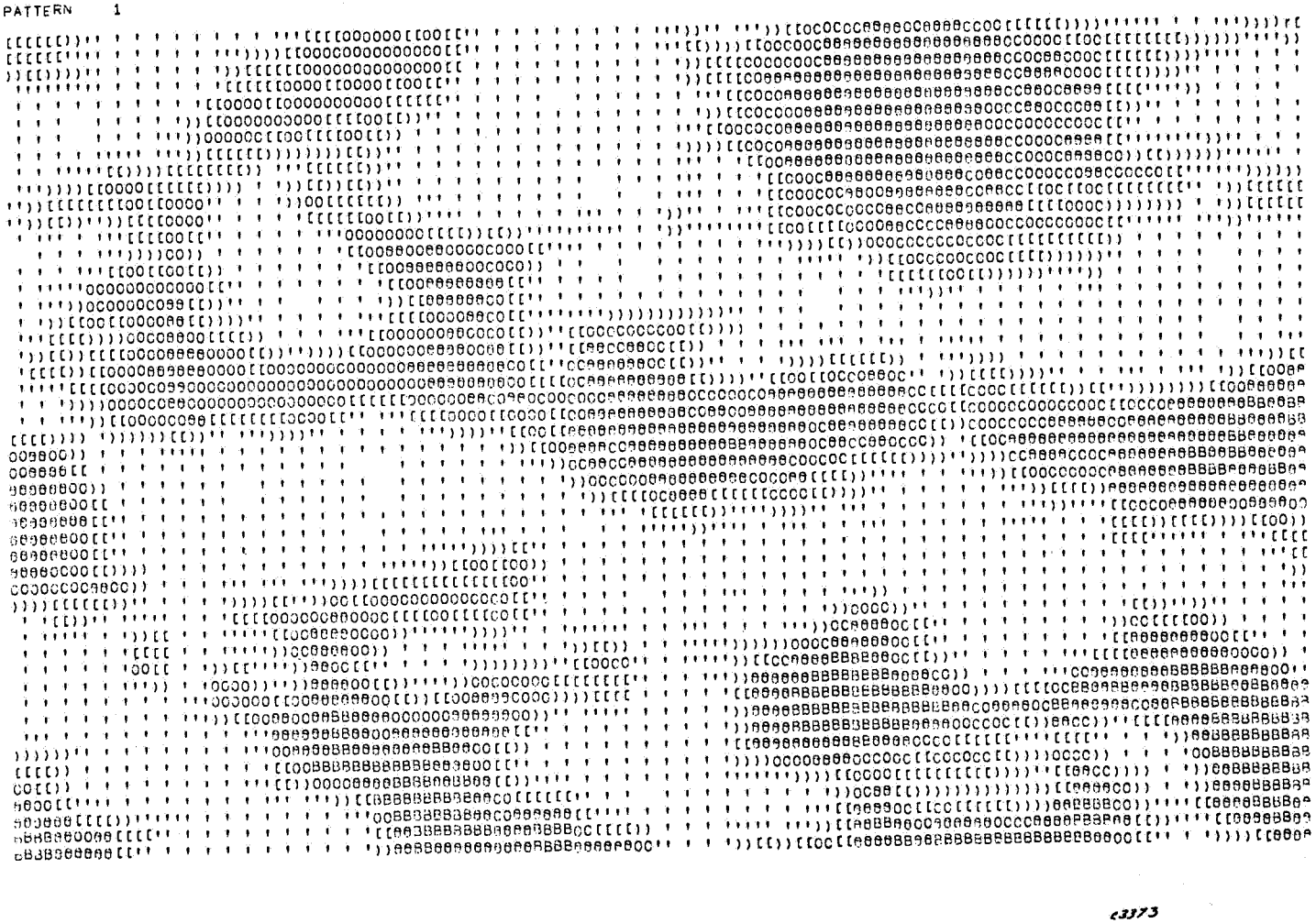


Figure 46. Montage Sequence

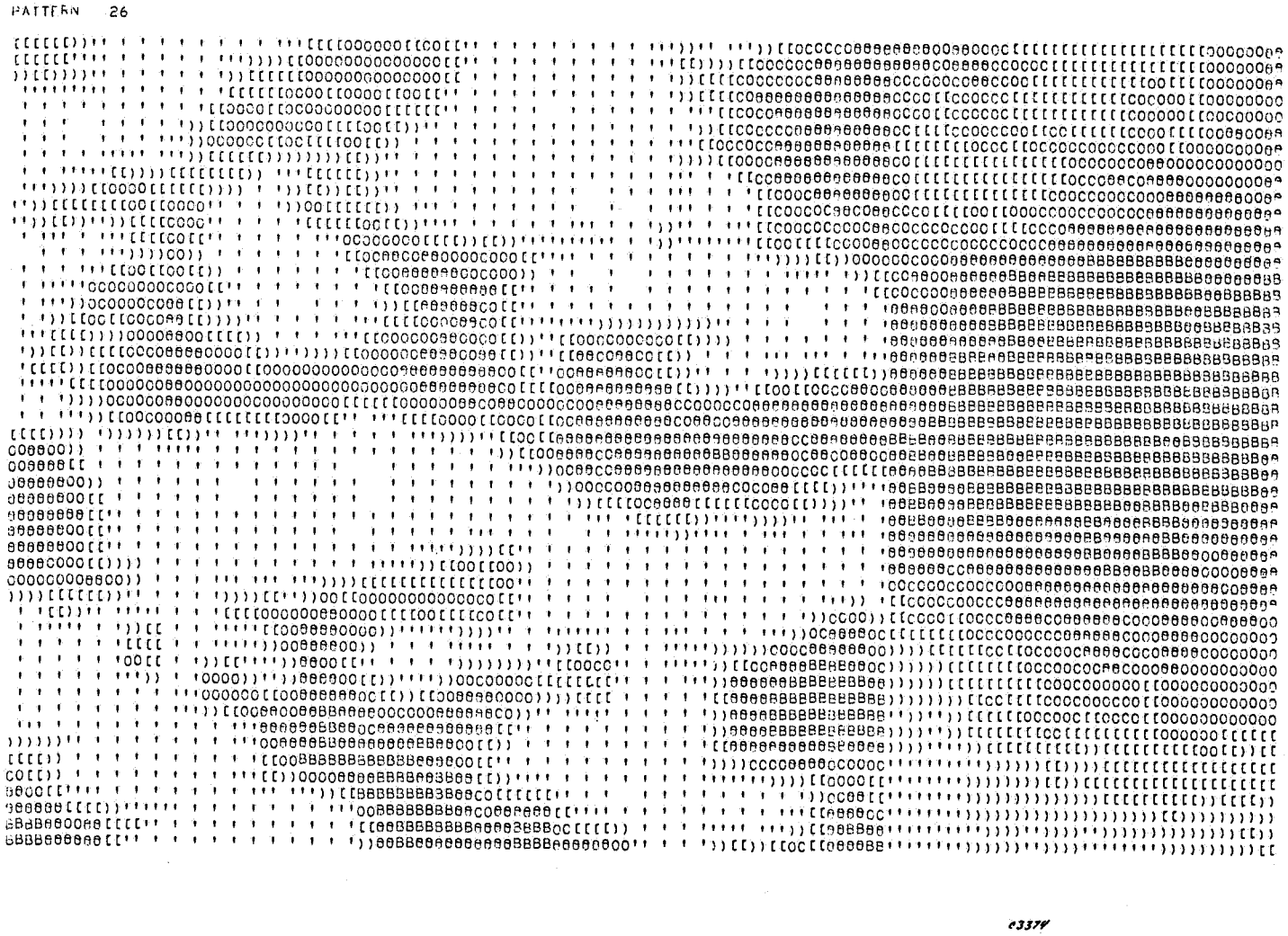


Figure 47. Montage Sequence

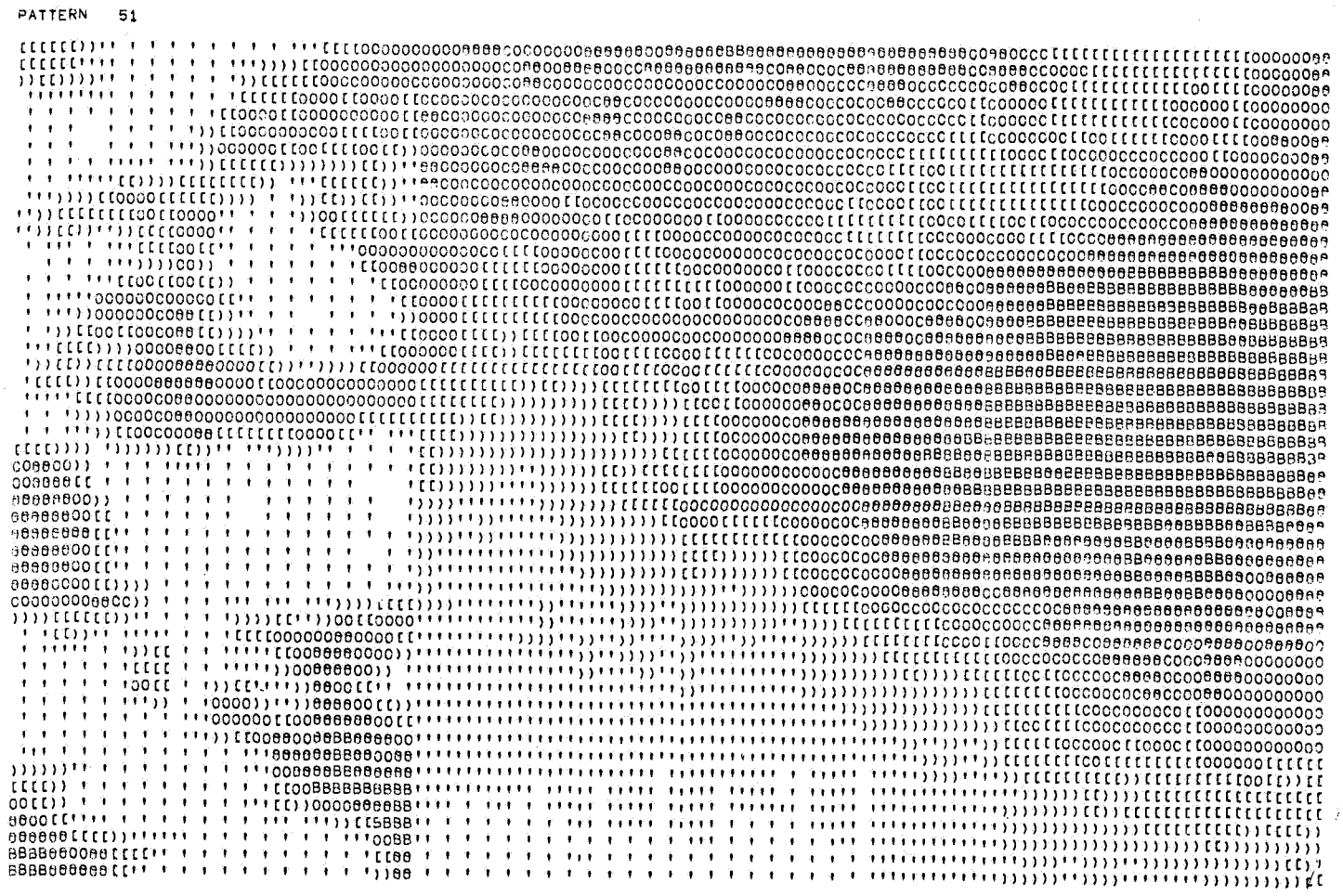
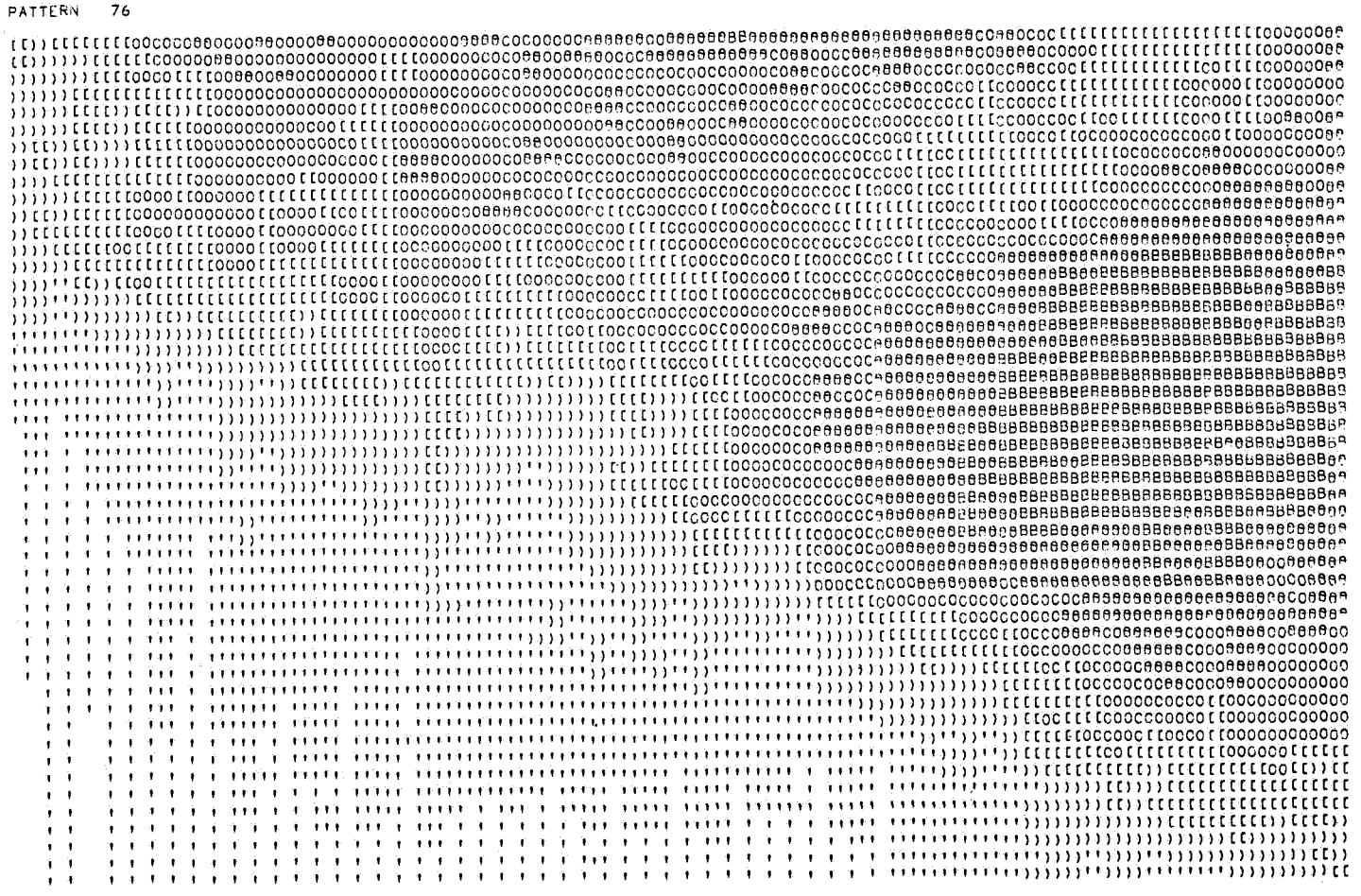


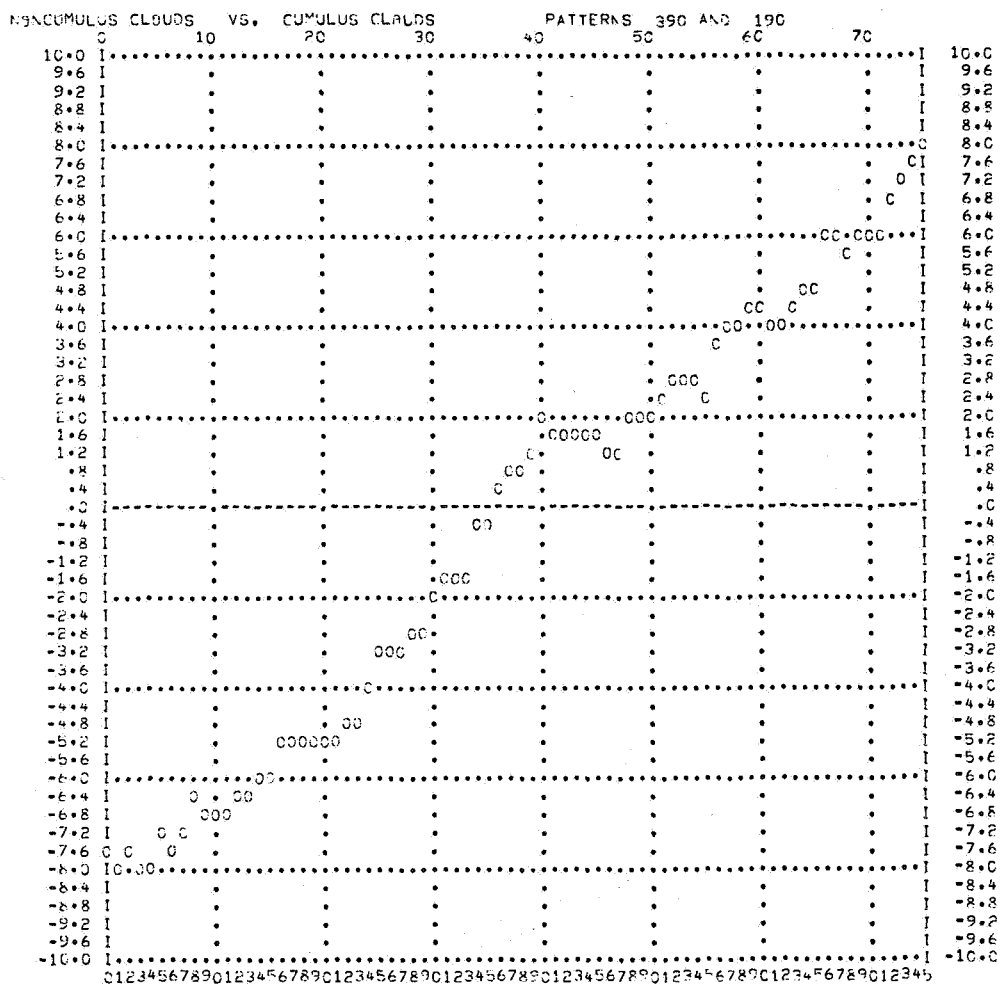
Figure 48. Montage Sequence



c3376

Figure 49. Montage Sequence

Amount of Overlap (raster element)



Response Unit Input

c3377

Figure 50. Discriminant Plot of Sequence

and correspond to the last 24 patterns in each sequence of 100. Of the remaining 35, 264 discriminants, 20,672 were subjected to visual examination for calibration purposes.

The remaining 14,592 discriminants represent the 192 montage sequences involving a high pattern of one class paired with a high pattern of another class. These data were computer analyzed to determine the degree of fit of the plot to a straight line.

3.3.8.3 Calibration

Montage sequences not included in the computer analyses were considered to be calibration runs. These runs encompass 272 montage sequences, or almost 59 percent of the sequences.

In the montage patterns, unlike real patterns, the boundary is present as a sharp discontinuity. The presence of this discontinuity could result in erratic performance of the property filters whose input connections straddle the discontinuity. The sequences meant primarily to determine whether or not this effect is present were those involving two "high" patterns of the same class. If the effect were present, one would expect to find a "sagging" towards zero in the computer plots. This effect, as illustrated in Figure 51, was found in a small minority of cases. These cases were counterbalanced by another small minority in which the curves bowed away from zero. The majority of cases showed unbiased fluctuations about a straight line connecting the endpoints, as illustrated in Figure 52. Thus, it is concluded that the property filter's input fields are sufficiently well distributed thoughout the aperture, and that the nature of the property filters is such that the presence of a discontinuity in the aperture does not create a serious disturbance.

With only the sequences involving a high pattern of one class and a high pattern of another class entering into the computer analyses, the question arises whether the results could be considered valid for other conditions. A large number of sequences invloving high-low and low-low pairs were processed, both for pattern pairs from the same class and for pattern pairs from different classes. The computer graphs were judged subjectively. It was judged that the "random" fluctuations about

Amount of Overlap (raster elements)

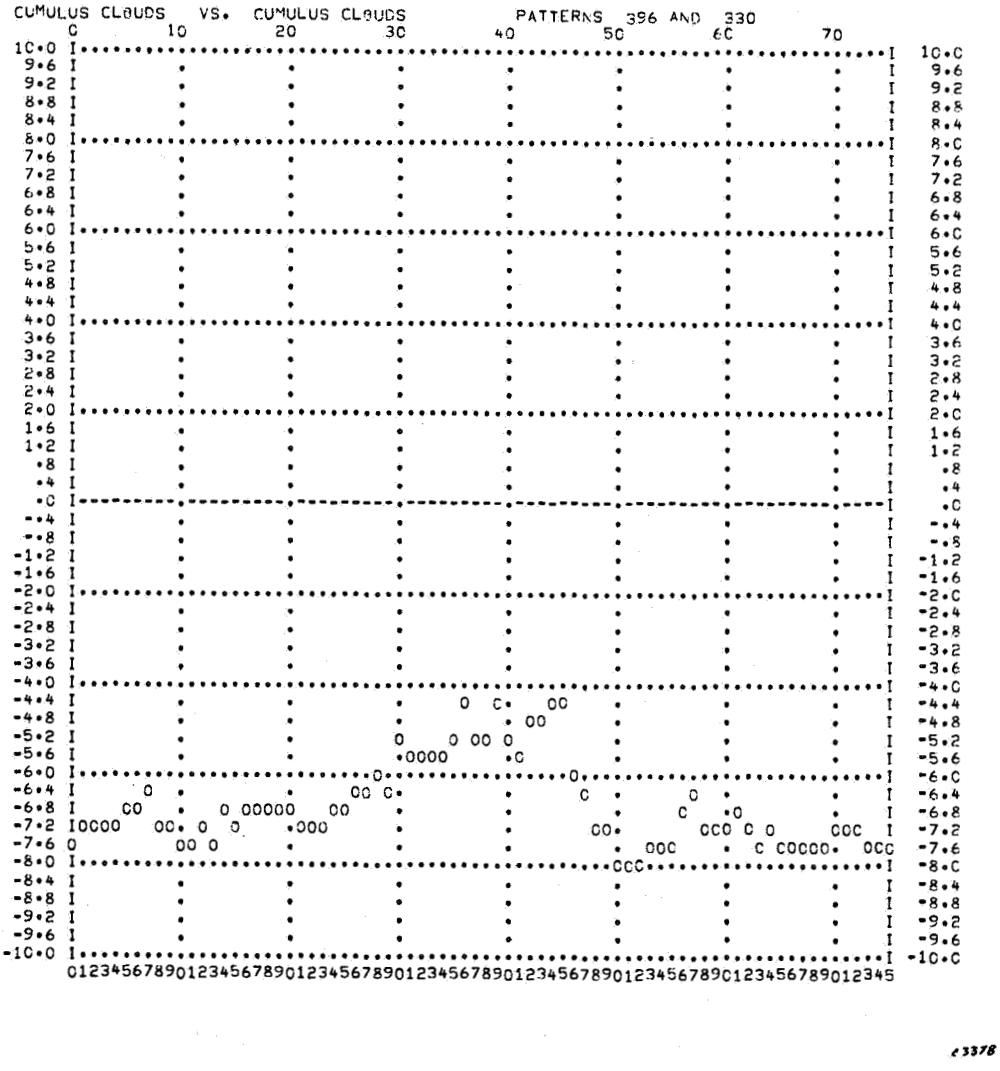


Figure 51. Discriminant Plot (same Pattern Class)

Amount of Overlap (raster elements)

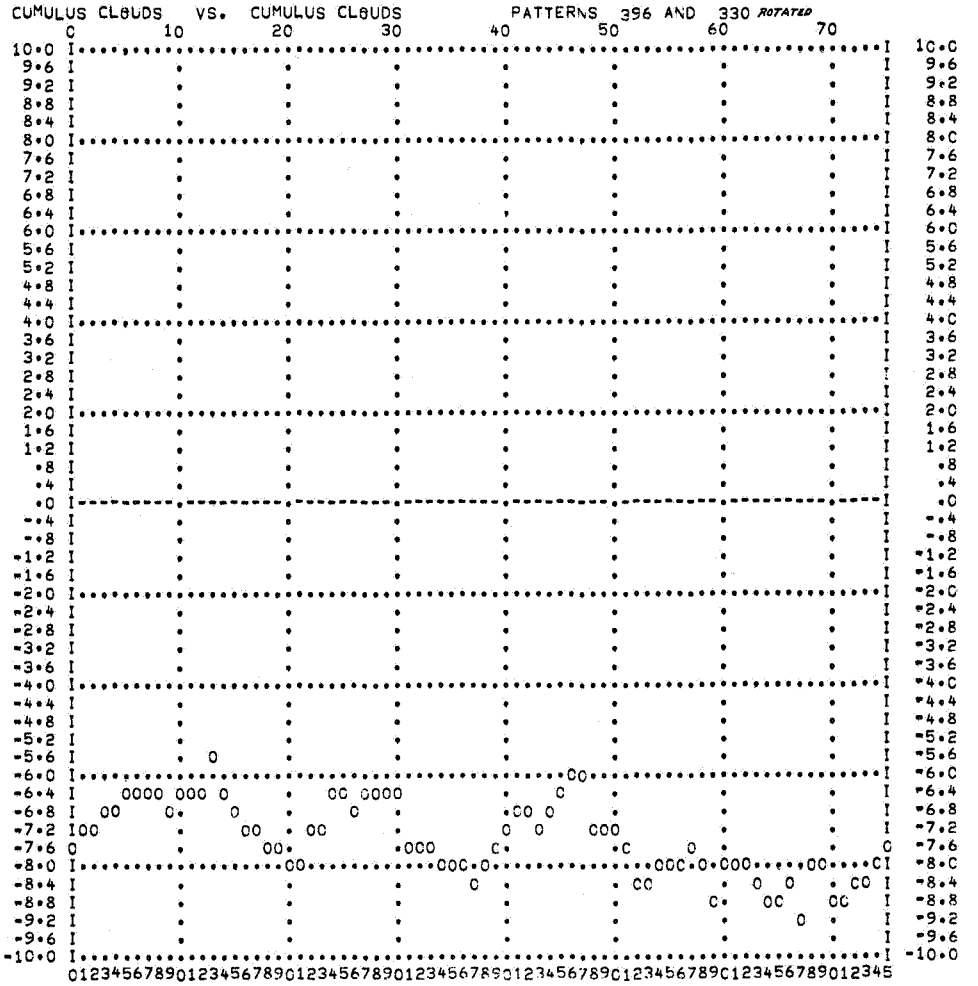


Figure 52. Discriminant Plot (same Pattern Class)

the underlying curve were the same for these calibration as for the noncalibration sequences, but that the underlying curves were probably more linear for the calibration sequences. This is probably due to the fact that the endpoints in the calibration plots are closer together than in the noncalibration plots.

3.3.8.4 Accuracy of Interpolation

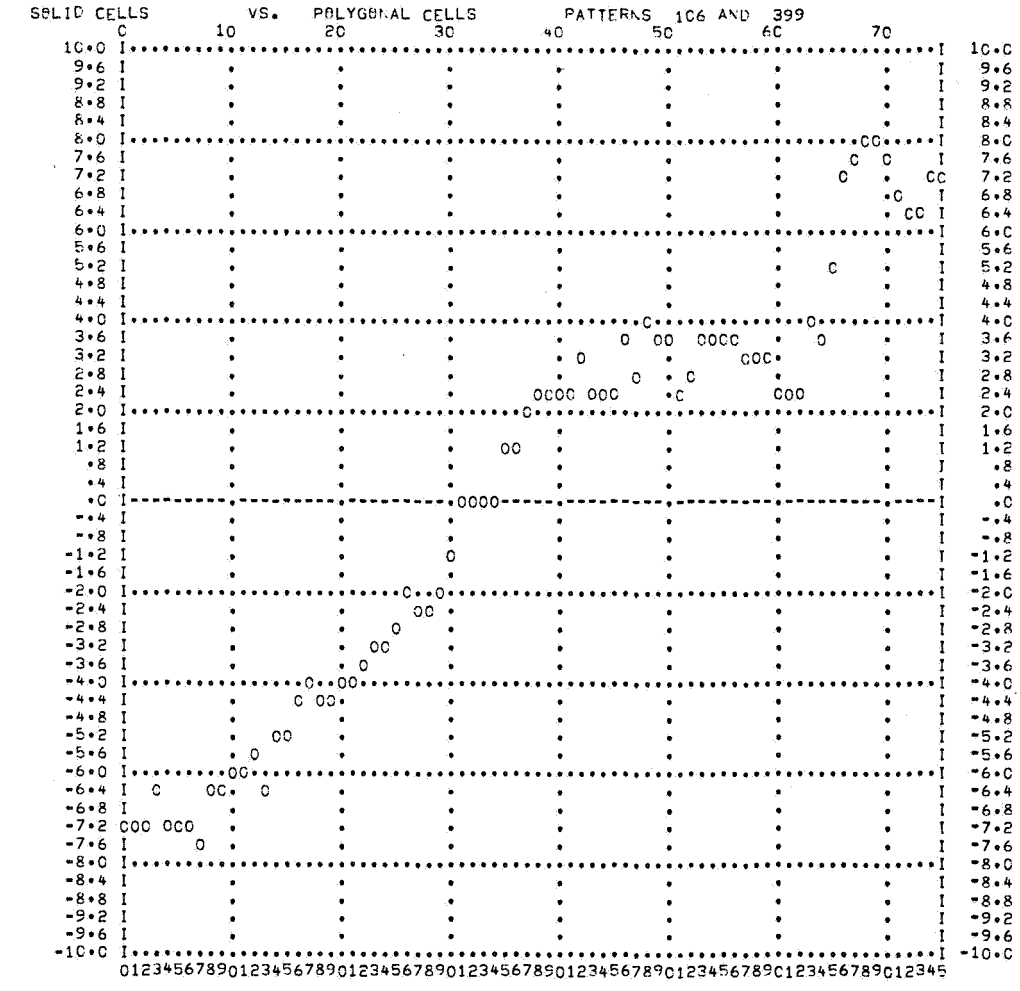
There were 192 pairings of "high" patterns for which the sequence of discriminant values were keypunched, 96 each for Tasks PVS and NVC. Several analyses relevant to linear interpolation were performed. Figures 53 through 56 are illustrative of the data processed and were selected to show the variety of plots encountered.

Straight lines were constructed joining the endpoints of the sequences (i.e., the discriminants for the "pure" patterns). The rms deviation of the other 74 times 192 discriminants about these lines was 1.94. This represents about 15 percent of the average full range. The rms deviations were also a function of the extent of the montage. Figure 57 presents these results. The deviations are largest (about 2.4) near the center of the plots, as might be expected. If the lines had been constructed by least squares, smaller deviations would have been observed, and the smallest deviations would likely not have been at the left side of the curve.

The value of the discriminants at the points where the constructed straight lines crossed the zero axis were noted. The absolute values of the deviations at these points had an average value of 2.00 and a standard deviation of 1.41. The median deviation at this point was 1.51.

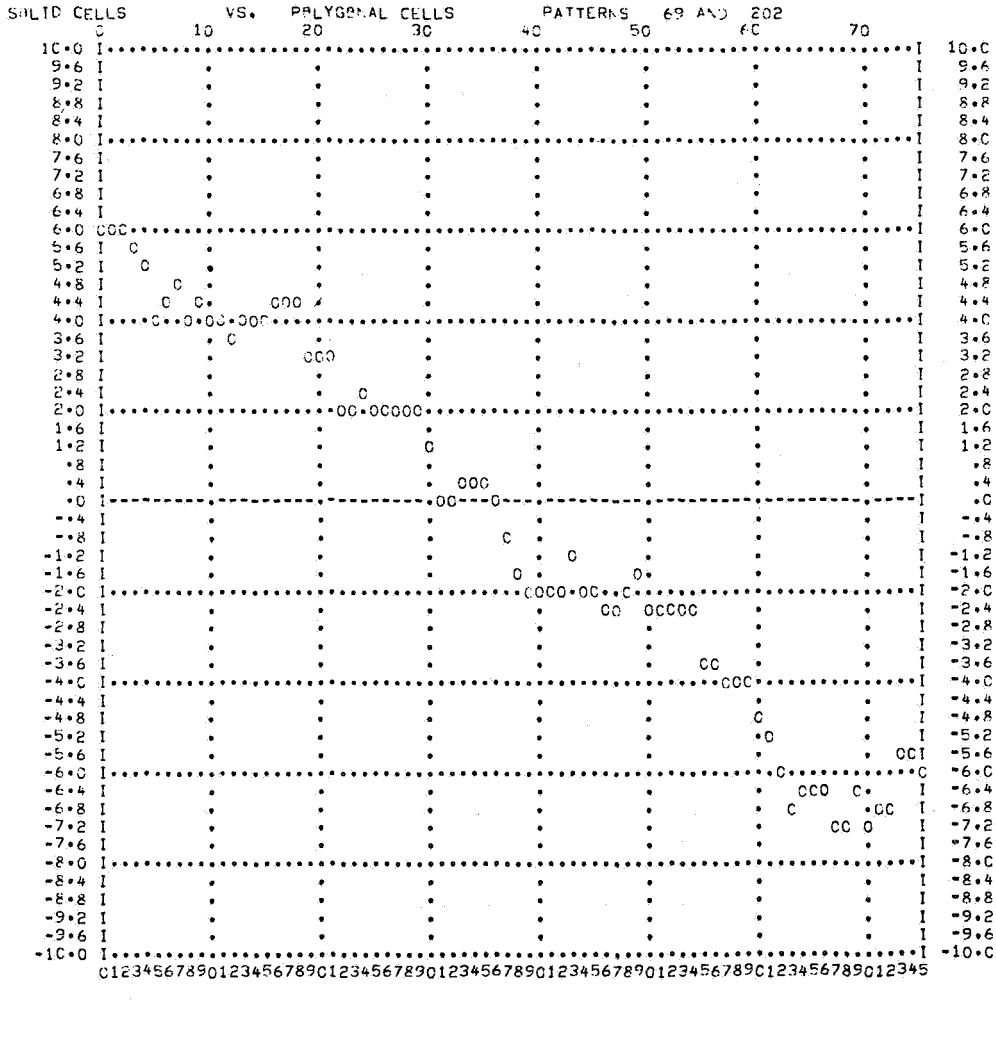
The average number of times a curve crossed the zero axis was 2.24. In almost two thirds of the cases, there was only a single crossing. The maximum number of crossings was 9. The average distance from the crossing of the straight line to the nearest crossing of the discriminant curve was 7.95, about twelve percent of full scale (full scale is 75). The median distance was 6.6.

The foregoing data are given to provide a feeling for the effectiveness of a straight line fit. The major results of the boundary delineation task are given by the analyses below. The assumption



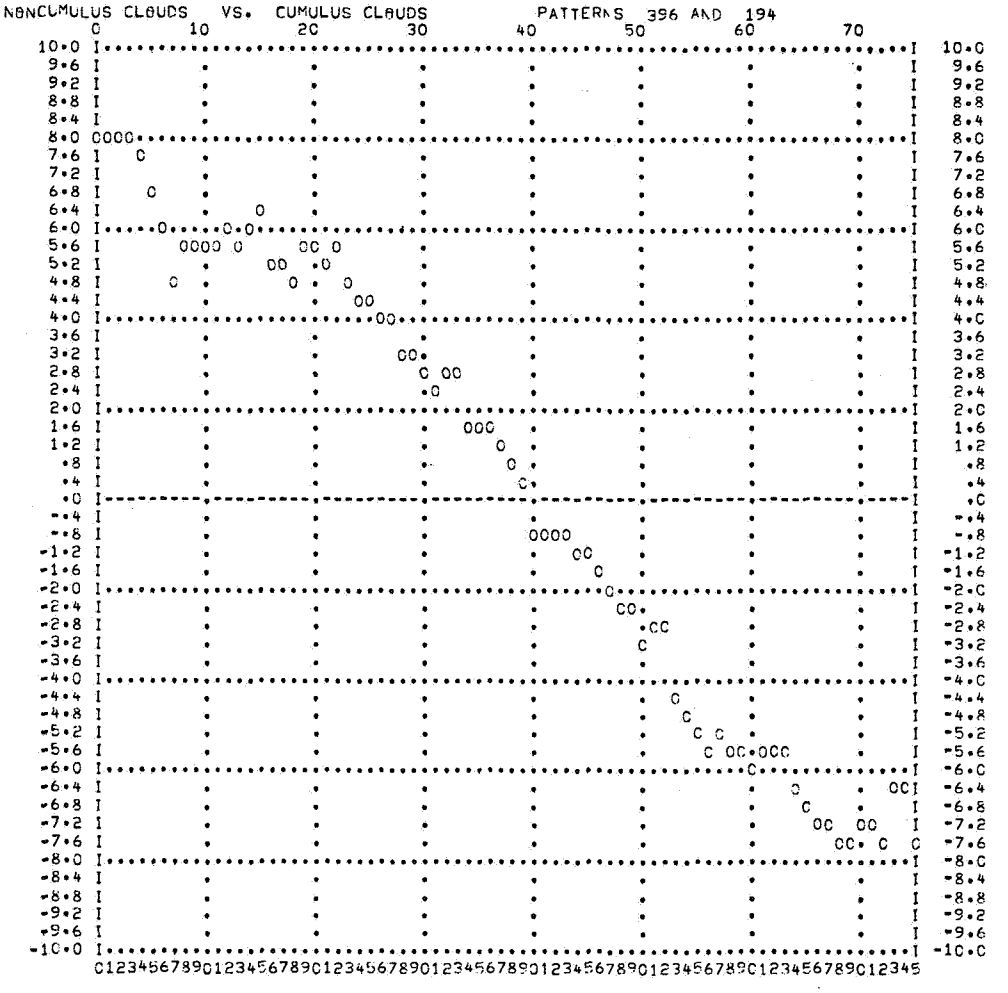
c3380

Figure 53. Discriminant Plot SVP



c5381

Figure 54. Discriminant Plot SVP



c3382

Figure 55. Discriminant Plot (NVC)

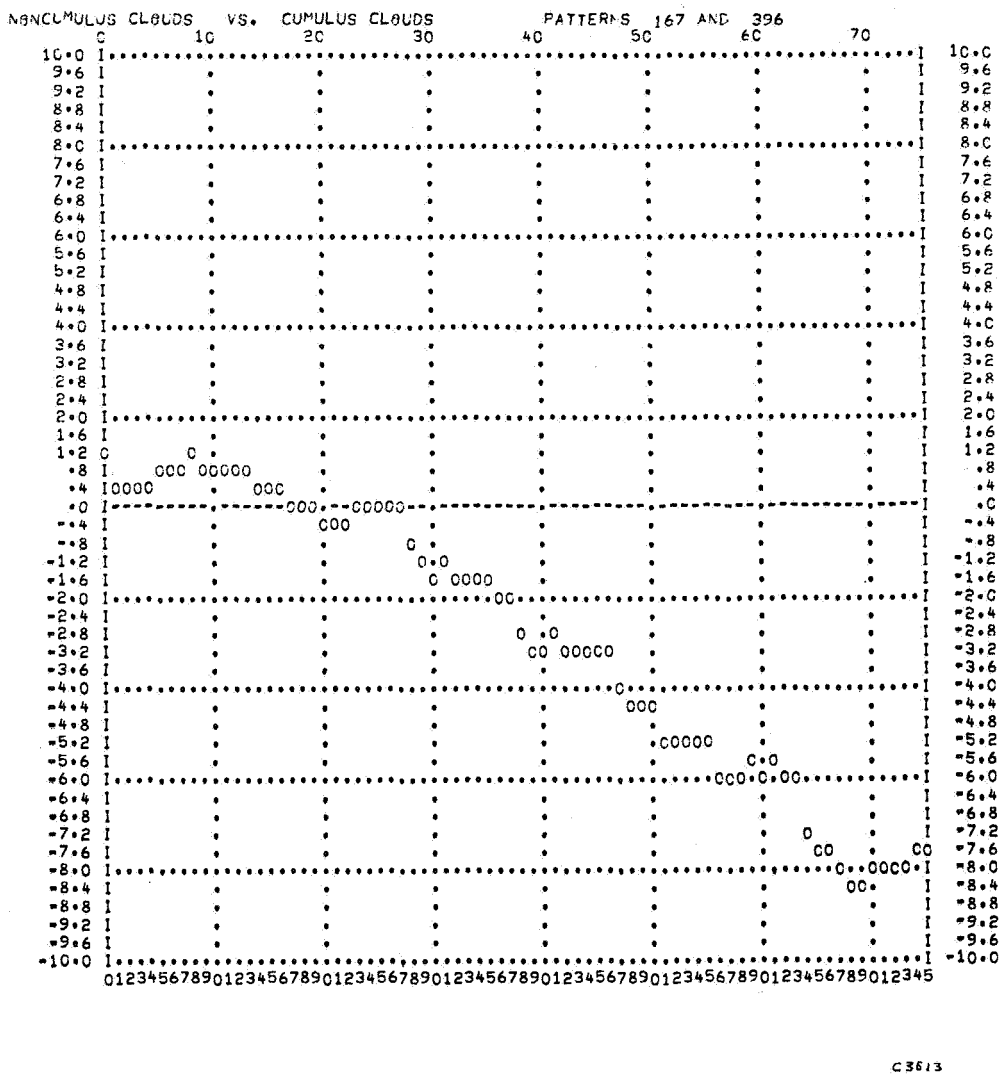


Figure 56. Discriminant Plot (NVC)

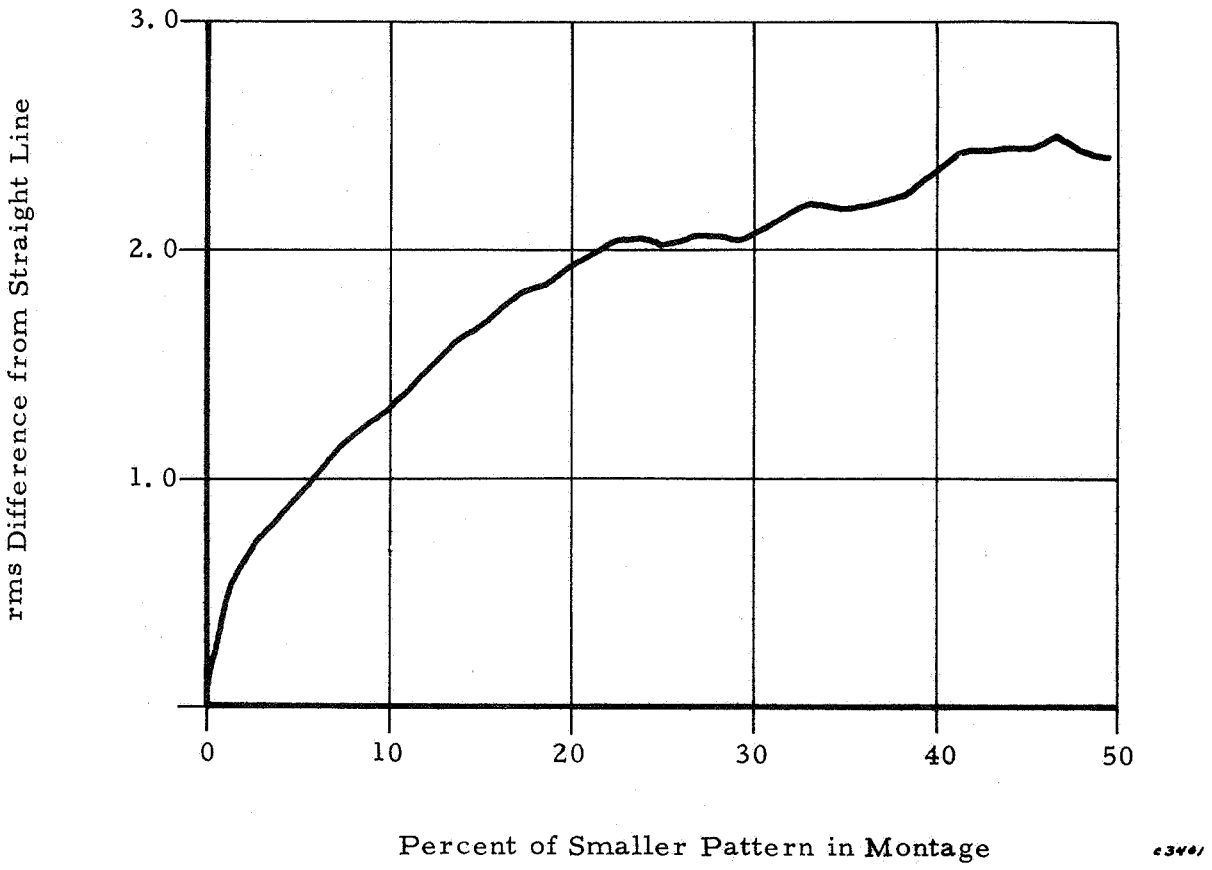


Figure 57. Deviations of Discriminant Value from Straight Line

was made that the recognition system first passes over a homogeneous region of clouds during which the discriminant would be constant (equal to the first discriminant in a sequence of 76). It then passes over a boundary, which would give rise to the next 74 discriminant values. Finally, it passes over a second homogeneous cloud region, where the discriminant is again constant (equal to the last of the 76 values). With this assumption, the graphs were extended on each end by horizontal lines. Distances from 1 to 75 between successive apertures were considered. For each distance, all possible transition pairs (successive decisions which are different) were found. The error in boundary placement was then computed for two cases — linear interpolation on the discriminant values, and linear interpolation on the decision (in which the boundary is placed halfway between the centers of the transition pair). The results are presented in Figures 58 through 60.

It can be seen from these figures, considerable advantage results from using the discriminant values rather than the decision itself. If the average accuracy requirement were ten raster elements (corresponding to about 23 miles at the resolutions used), this could be achieved with an aperture spacing of 72 raster elements using the discriminant values, while the use of the decisions only would require a spacing of no more than 10 raster elements. The latter case involves the processing of more than 50 times as many apertures than the former does. Using the discriminant values for interpolation, the accuracy achievable does not depend strongly on the spacing between apertures, within the range tested, the entire range of accuracies falling between 8.58 and 10.22 raster elements (11.5 to 13.5 percent of the aperture size). The maximum accuracy is achieved with spacings of 42 to 46 raster elements, not far from the spacing of 38 selected for the hardware studies prior to these experiments.

3.4 Conclusions and Recommendations

1. Using only statistically derived property filters, recognition systems were designed to give more than 84 percent correct generalization decisions on all five tasks. For three of the tasks performances near 85 percent were obtained, on one task 96 percent was obtained, and on the fifth task, 99.5 percent was achieved. To accomplish this, the size of the sampling

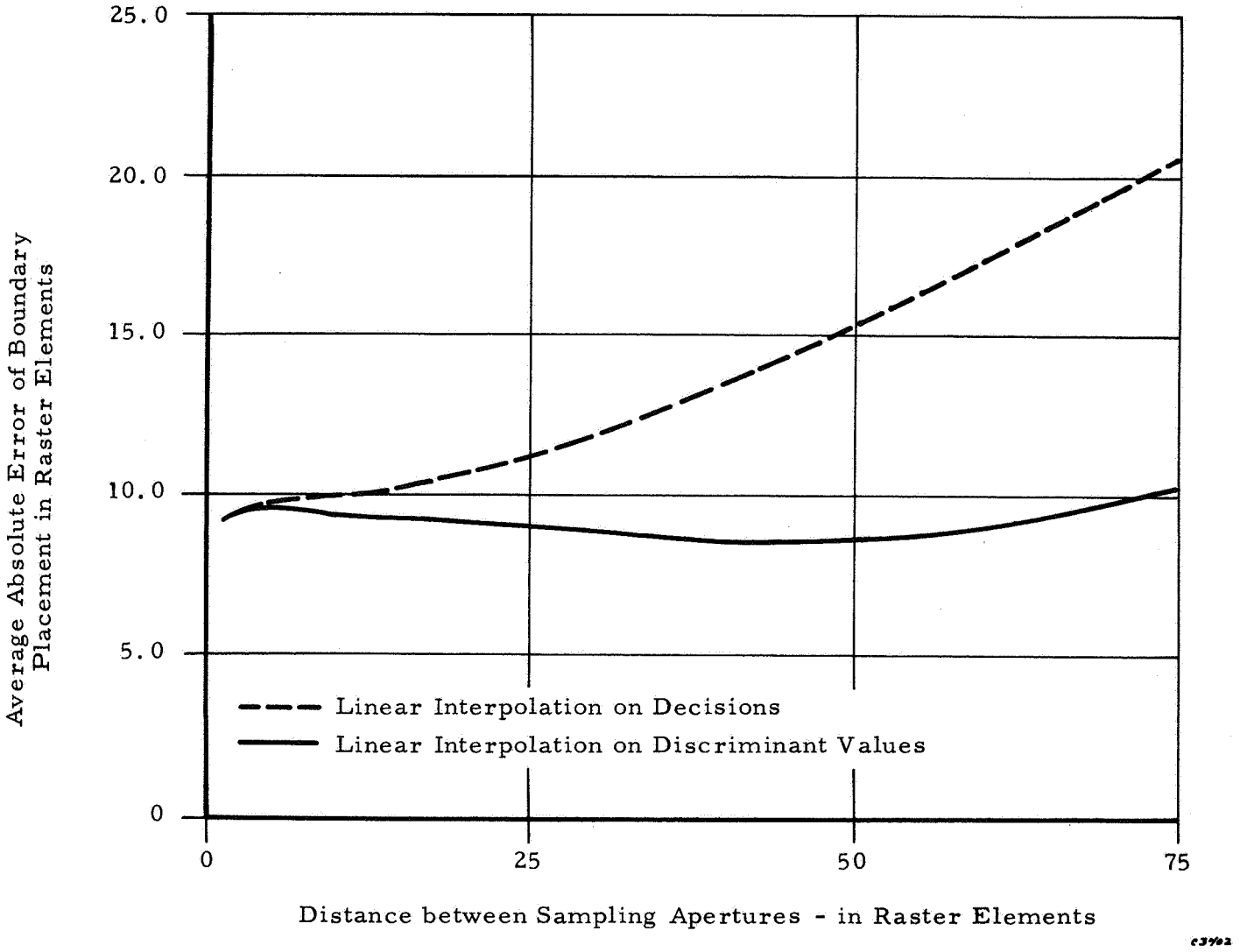


Figure 58. Interpolation of Boundary Location (NVC)

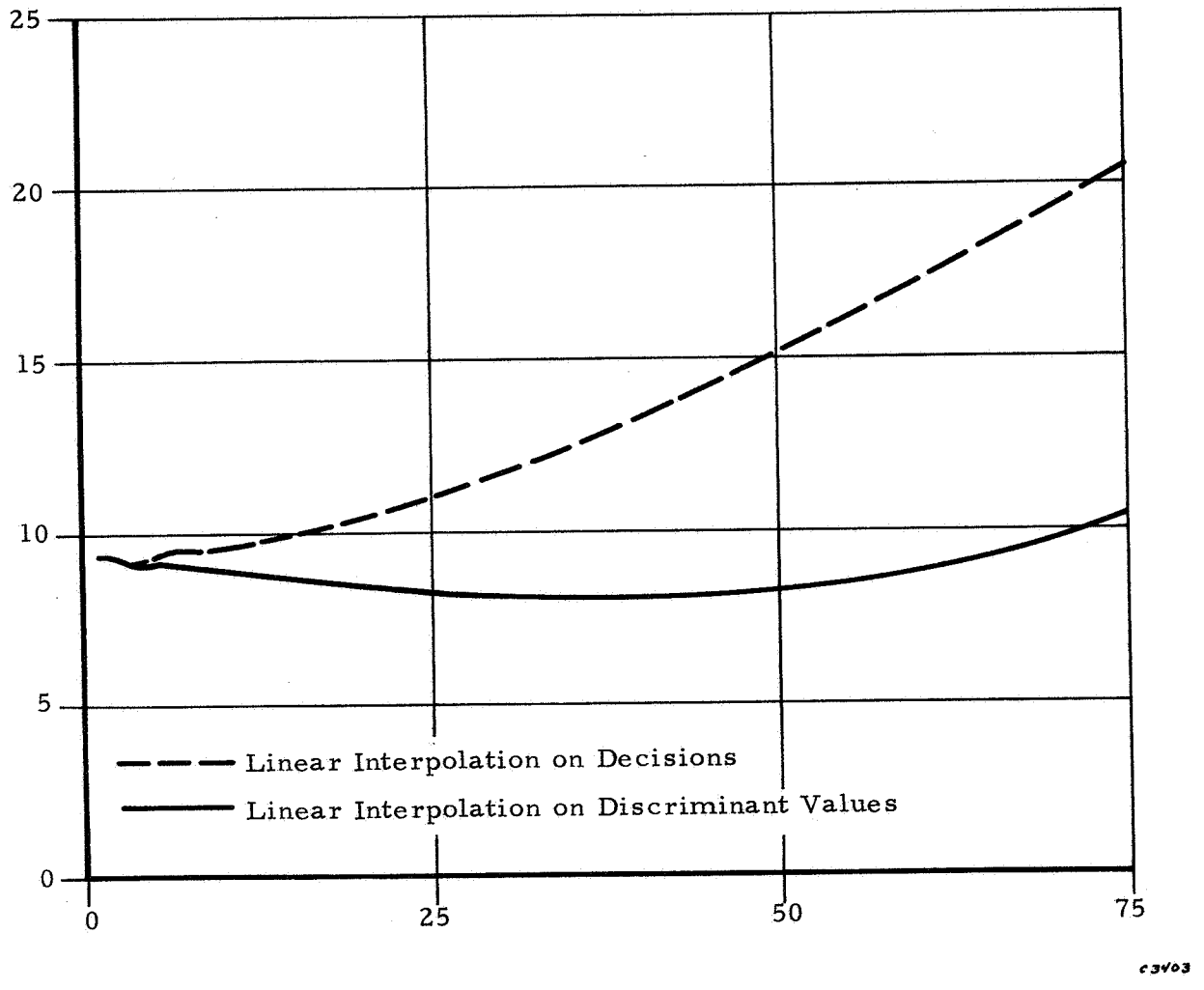


Figure 59. Interpolation of Boundary Location (PVS)

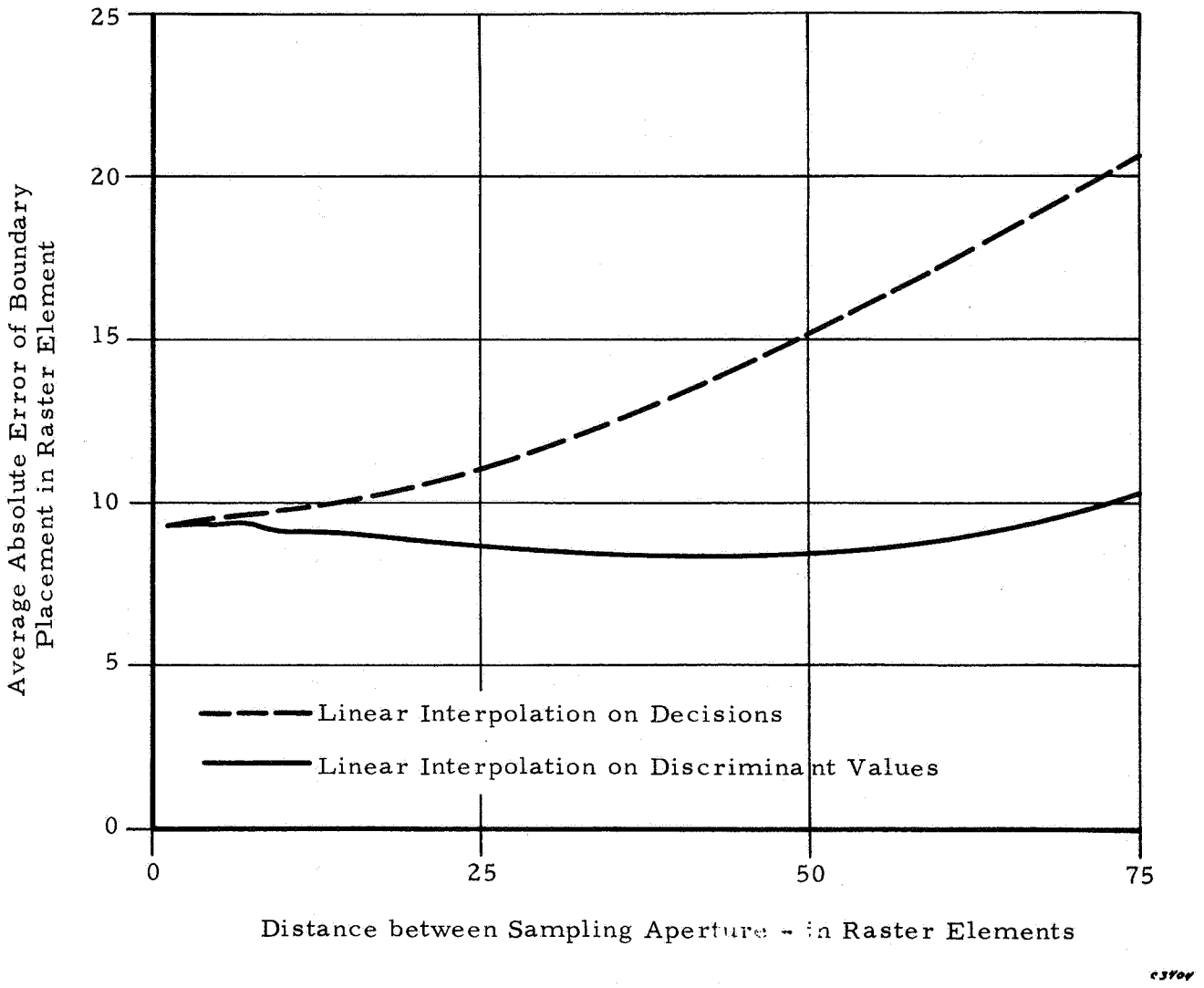


Figure 60. Interpolation of Boundary Location (Combined)

aperture has to be reduced for two of the tasks. The importance of choosing the proper aperture size was strongly emphasized on one task, for which performance rose from 64 percent with a 50 by 50 aperture to 96 percent with a 15 by 15 aperture.

Two of the tasks, for which 86 and 87.5 percent were obtained, were cloud pattern recognition tasks. Using only 15 known properties recognition systems were designed which achieved 90 and 97 percent correct decisions on these tasks. Augmenting the logically designed property filters with statistically designed property filters raised the 90 percent figure to 92.75 percent.

In view of the quality of the sample patterns, many of which are difficult for a human observer to identify, it is felt that the performances obtained are sufficiently good to warrant the further development of a space-borne recognition system.

2. By interpolating on the discriminant functions generated by the recognition systems, boundaries between cloud pattern types could be located to an average accuracy of 8.6 to 10.2 raster elements (corresponding to 11.5 to 13.5 percent of the aperture size). This accuracy was relatively constant for all distances between sampling apertures up to 75 raster elements – the no overlap case. These figures do not include spurious boundaries introduced by incorrect recognition system errors. To obtain comparable accuracies when the discriminant functions are ignored and only the decisions are used, the spacing between consecutive apertures can be no more than 14 raster elements.

It is concluded, therefore, that for boundary delineation the discriminant function should be used rather than the decision, itself. If this procedure is followed it is not necessary to process overlapping apertures in order to accurately locate boundaries between regions of different homogeneous textural patterns.

3. A set of 15 known properties, produced very good results on the tasks for which they were designed and, indeed, performed moderately well on tasks for which they were not designed.

Eight experiments to augment the known properties with statistically designed property filters were performed. In two experiments, in which performance with the known properties alone was excellent, augmentation lead to no change or a decline in performance. Of the other six experiments, augmentation yielded two cases in which the improvement was small and unlikely to be statistically significant; one case of a moderate improvement which was significant at the 92 percent level; and three cases of substantial improvement, significant at more than a 99.9 percent level.

Where possible known property filters should be designed for each recognition task. Augmentation of these known properties with statistically designed properties should then be attempted and the resulting performance evaluated. The probability of registering improvement with this procedure appears to be about 50% with the improvement being accomplished where it is needed most – where known property performance is low.

4. In the experiments performed, the statistical property filters with quadratic switching surfaces consistently performed better than those with linear switching surfaces, even though considerably more linear units were used. The linear units were designed using the first sample moments, and are therefore based only on average gray scale differences. The quadratic units were based on the first two sample moments, and are based on two-point contrasts as well as gray scale differences. It is possible to approximate quadratic units with linear units by using more than one threshold. (160) Had this been done, it is likely that the performance with linear units could be brought up to the level of the quadratic units at the expense of a larger property filter set.

Amongst the eight algorithms (plus three modifications) for designing decision functions, no one technique showed a consistent advantage over all others. On all problems for which high performance levels were obtained by any method, the forced learning, Bayes' weights, and mean square error algorithms failed to provide high performance. Distribution estimation is too complex a technique to be considered in view of the fact that its performance is no better than average. The remaining algorithms were iterative design, error correction, piecewise linear, and MADALINE. The failure of the latter two, which yield nonlinear decision functions, to outperform the

linear techniques is probably due to the method by which the property filters were selected — a method which emphasizes linear separability of the sample patterns.

5. Attempts to improve the original methods of designing decision functions and statistical property filters were, by and large, not successful. The added algorithms, and modifications to earlier algorithms, for the design of decision functions lead to shorter and more stable design runs, but not to significantly better performances. A method for selecting subspaces for property filters deterministically, and establishing the switching surfaces nonparametrically lead to a moderate performance increase on one cloud pattern task, and a substantial performance drop on the other.

The authors feel that better methods for the generation of property filters can lead to more effective recognition systems, since our present ability to design decision functions far exceeds our capability to extract a meaningful set of pattern properties.

4.0 HARDWARE FEASIBILITY STUDY OF AN 'ON BOARD' RECOGNITION SYSTEM

4.1 Introduction

In order to evaluate the performance of the three spaceborne recognition systems reported here, it has been assumed that the system is operating on imagery similar to that obtained by the Automatic Picture Transmission (APT) System [162] aboard an earth-orbiting meteorological satellite. The recognition system classifies 75 x 75 element subsections of a picture into one of a prescribed number of cloud (or background) classes. A code designating the class is either stored aboard the spacecraft or transmitted to earth, or both.

The usefulness of performing on-board processing is apparent when one considers even the relatively modest data link and data acquisition requirements of the APT System. The APT vidicon camera system, covering a region on the earth of about 1050 x 1050 miles with a 700 line resolution, generates approximately 22×10^6 bits of information per orbit (i.e., 15 daylight pictures/orbit with seven gray levels). If an on-board recognition system, similar to that to be described, were employed, with an optics system covering the same ground region, only 42×10^3 bits of information would be generated per orbit. This figure allows for the transmission of 15 bits to specify the system's response unit input (i.e., discriminant function) each time a classification is made. The result is a substantial reduction in transmission bandwidth, data storage, and subsequent data processing requirements.

The 'on board' recognition system will employ a wide-angle lens having a field of view of 1050 x 1050 n. miles from a 500 n. mile altitude similar to the APTS. Cloud cover classifications are made on 175 x 175 n.m. subsections of this area. With the satellite in a polar orbit, the east/west scanning of each subsection is accomplished electronically by means of the input system, while the north/south scanning results from the movement of the satellite (Figure 61). An overlap of 50% east/west and north/south is

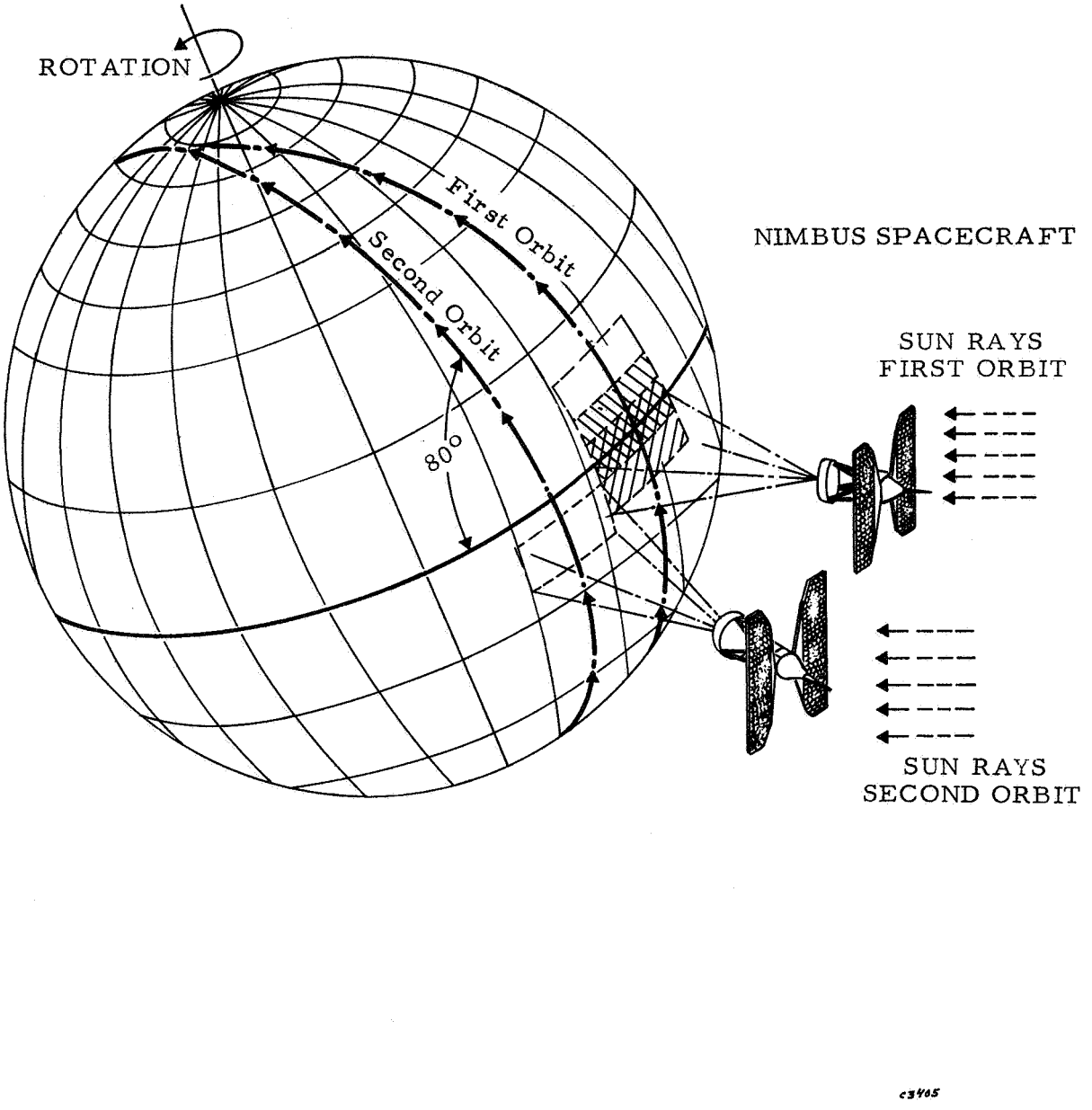


Figure 61. North/South Scanning By Satellite Movement
(50% overlapping strips)

introduced in order to increase the accuracy with which the boundaries between different cloud classes may be determined.*

The east/west scanning of the input system yields 11 overlapping subsections covering a region on the earth of 175 by 1050 n. miles (Figure 62). Classifications are performed on each subsection. To obtain the 50% overlapping subsections in the north/south direction the satellite must traverse a ground distance of 87.5 miles. The ground speed of the NIMBUS (500 n.m. orbit) is approximately 3.61 n. miles/second. Therefore the eleven east/west overlapping subsections must be processed in about 24 seconds. All of the implementation approaches investigated and discussed in this section require less than this time to perform the required processing.

The pattern recognition system may be divided into two parts — (1) an input unit and (2) a decision network. The general structure of such a recognition device is illustrated by Figure 63.

To perform the cloud pattern classification task as discussed in Section 3, it has been determined that a sensory resolution of 75 x 75 elements for each subsection can be utilized. (163) Image storage and shuttering is not required if a subsection classification can be performed in less than 0.645 sec., the time required to move one sensory resolution element. This requirement is satisfied for all of the systems considered in the feasibility study.

The input unit consists of an image dissector tube with associated horizontal and vertical deflection coils, focusing coils, x and y deflection amplifiers, three digital-to-analog converters and corresponding flip-flop registers, and a signal conditioning amplifier for the dissector tube output. Except in the parallel/analog implementation of the decision network, a four bit analog-to-digital converter is also considered to be a component part of the input unit.

---

*In Section 3.3.8 of this report, it is shown that for abrupt transition boundaries, if the discriminant value (response unit input sum) are used as a means for interpolating to find the boundary location, little benefit is gained from providing this amount of overlap. If in actuality 50% overlap is not required, the system may be modified to yield potentially a 75% reduction in the computing duty cycle, with commensurate savings in power consumption.

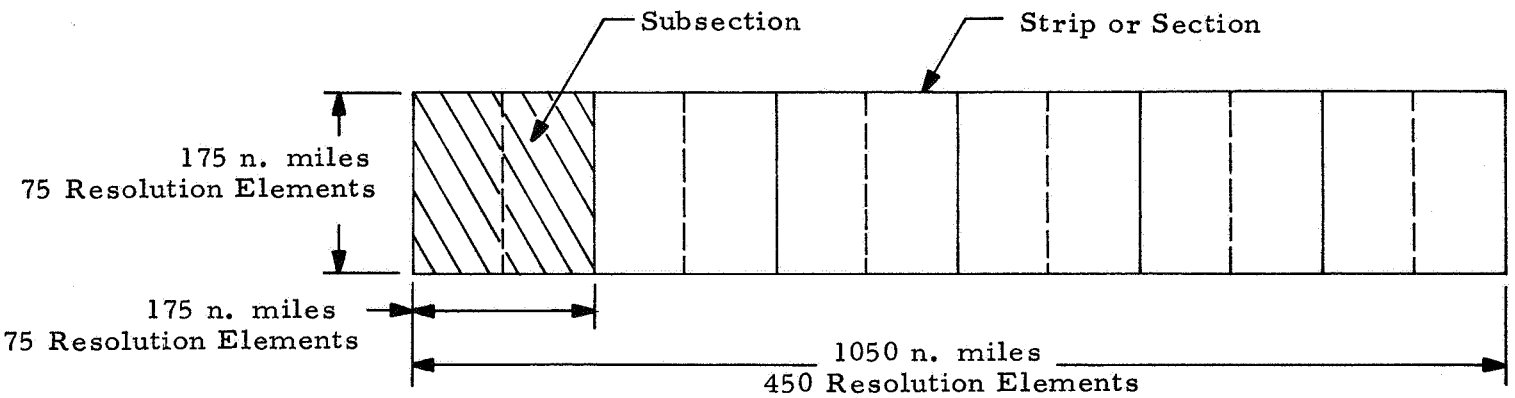


Figure 62. East/West Scanning By Input System
(50% overlapping subsections)

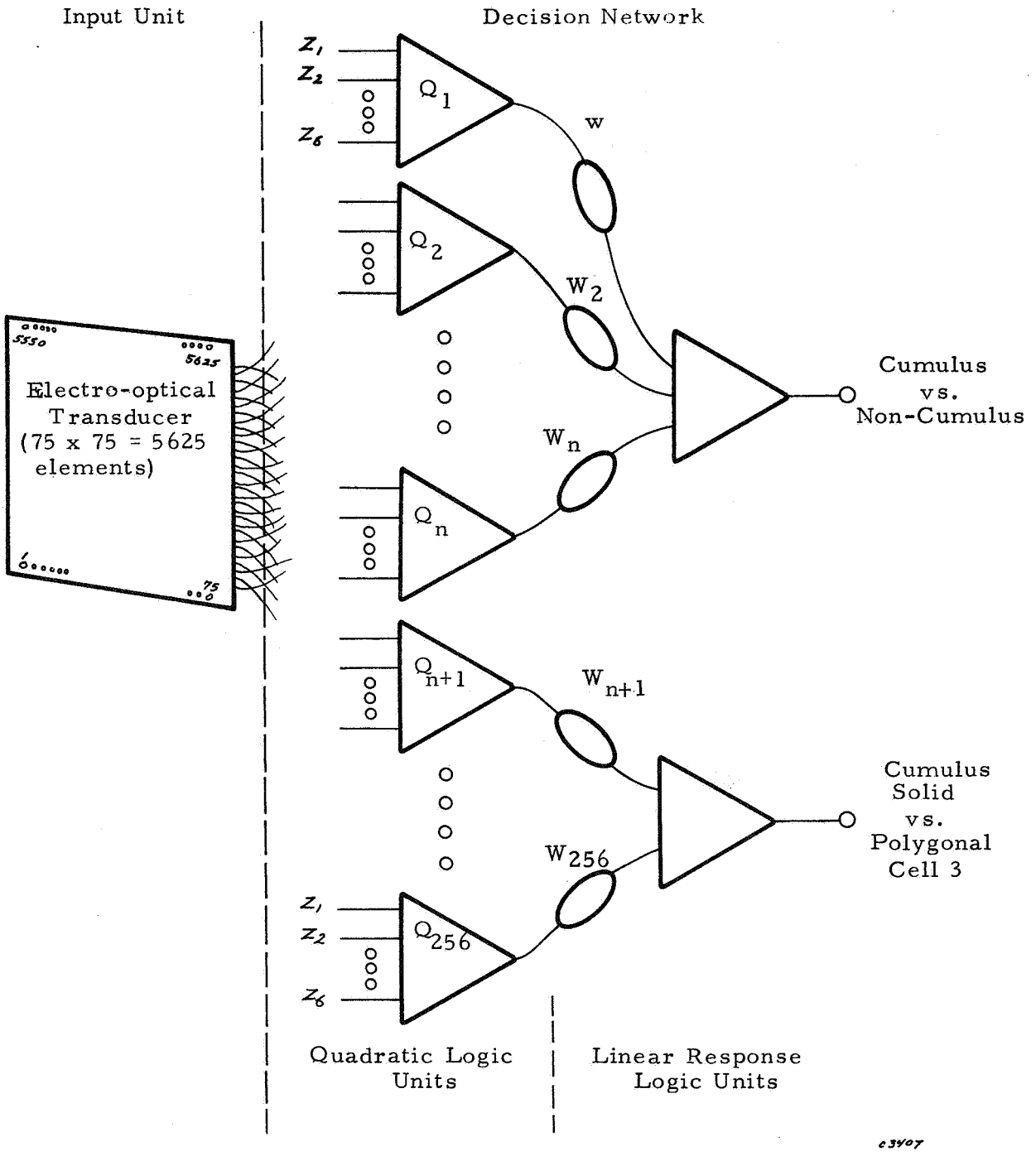


Figure 63. Recognition System Block Diagram

The decision network is made up of 256 first layer quadratic logic units and two second layer response units. Each quadratic unit has six inputs, z_1 through z_6, which are a function of the light intensity at the corresponding coordinates (x_1, y_1) through (x_6, y_6), where (x_i, y_i) is one of the 5625 points of the subsection being processed. One second layer response unit receives inputs from the first N quadratic units while the second response unit receives input from the remaining 256-N quadratic units. The first response unit discriminates between cumulus and non-cumulus cloud cover, while the second response unit discriminates between cumulus, polygonal cells and cumulus, solid cells. In the simulation studies reported in previous sections, only 150 quadratic units are required to accomplish these classification tasks. The hardware feasibility study uses 256 quadratic units, hence the estimated size, weight and power consumption figures tend to be conservative.

Each of the 256 quadratic logic units implements the following algebraic function

$$a_o + \sum_{i=1}^{6} z_i \left(b_i + \sum_{j=1}^{i} c_{ij} z_j \right) \tag{1}$$

If the function is zero or positive then the quadratic unit output Q_m is 1, otherwise $Q_m = 0$. The above expression contains one threshold (a_o), six linear weights (b_i), and 21 quadratic weights (c_{ij}). These 28 coefficients, as well as the coordinates corresponding to each z_i, are completely specified in advance by training the recognition system on a sample set of patterns by means of a computer simulation program (Section 3.0).

The purpose of the two linear response units is simply to test the inequalities.

$$\sum_{m=1}^{N} Q_m w_m + \theta_1 \geq 0 \text{ and } \sum_{N+1}^{256} Q_m w_m + \theta_2 \geq 0 \tag{2}$$

Three techniques for implementing the recognition systems were evaluated in this study. The three systems considered are (1) parallel/analog,

(2) sequential/hybrid, and (3) sequential/digital. The term "parallel" refers to the fact that all 256 quadratic computing elements are separately implemented. The term "analog" indicates that the quadratic unit's z-inputs are analog voltages. In the sequential approach the general quadratic expression (involving six inputs) is implemented once, in hardware, or by means of a computer program. The coefficients and input coordinates are specified and the expression is evaluated sequentially for each of the 256 quadratic units. The term "hybrid" refers to the fact that the z-vector inputs are supplied in both analog and digital form to the quadratic computing element. The term "digital" implies that the z-vector inputs (as well as weights and thresholds) to the quadratic computing element are all binary.

Table XVII contains a summary of each system and the major subsystems, along with the results of the hardware feasibility study.

All of the systems require some form of memory to store the connectivity of the system, the quadratic and linear weights, and the thresholds. It should be noted that in the sequential/hybrid approach the diode matrix can be replaced by a core memory which should yield a smaller weight and volume, but will increase the power dissipation and cost. Similarly, the diode matrix could replace the core memory in the sequential/digital approach if a small writing buffer memory were included. Another memory technique which appears useful is a transparency containing digital weights, thresholds, and access information.

Processing time in each case could be reduced at the cost of additional power consumption by decreasing the time required for the image dissector to access a data point.

If a computer with sufficient memory capacity is an integral part of an existing satellite system, the possibility exists for implementing the sequential/digital approach using that computer, thereby requiring only the addition of the input system. A further advantage of the sequential/digital approach is that it allows augmentation of the statistically designed properties (logic units) with known properties (see Section 3.3.7) extracted from the input pattern by special digital subprograms. This is the only system configuration which affords such flexibility.

TABLE XVII

SUMMARY OF IMPLEMENTATIONS STUDY

System	Subsystems				Processing Time/Strip (seconds)	Duty Cycle	Volume (Cubic Ft.)	Weight Power	Average Pound Dissipation (watts)	Advantages
	Input Device	Computing Element		Memory						
		Type	Quantity							
Parallel/Analog	Image Dissector	Analog Multipliers	6 x 256 1536	Fixed Resistors and Hardwired Connections	6.29	26%	3.7	50	180	None*
Sequential/Hybrid		Hybrid Multipliers	33	Diode Matrix	2.25	9.3%	2.1	32	37	Simplicity, Cost, Power, Size, Weight
Sequential/Digital		Arithmetic Unit	1	Cores	2.45	10.2%	.9	26	30	Flexibility, Power, Size Weight

* (1) Best speed if photosensor array is substituted for image dissector (less than 110 msec/strip).

(2) With stored coordinate data and image dissector (100 μsec/data point) the parallel analog system speed may also be increased (less than 1.8 seconds/strip).

The implementation study reveals that, with current engineering technology and commercially available components, a useful sequential/hybrid or sequential/digital recognition system could be constructed with weight, size, and power requirements comparable with those of the satellite-borne APT System. In view of the classification accuracies obtained (Section 3.0), and the results of this hardware feasibility study, the development of an on-board recognition system for cloud cover mapping appears to be practical.

4.2 Input System

It is possible to construct the light sensitive portion of the input unit with an array of 75 x 450 (33,750) discrete photocells. This approach is not used since the size of such an array of discrete elements would be at least 7.5" x 45.0".* If a monolithic photosensory array is used to replace the array of discrete sensor elements the size problem can be alleviated. However, current technology in monolithic arrays does not yield adequate uniformity between sensors to allow discrimination of eight gray levels. In general, the primary reason for employing such an array of light sensory elements is in the interest of generating rapid responses from the recognition system. As previously described, however, without image storage the system is required to classify a new subsection every .645 seconds, the time interval corresponding to a satellite displacement of one resolution element. This allows ample time for each data point to be acquired sequentially. To generate a subsection classification it is necessary to acquire an input set of data representative of the light intensity of a maximum of 5625 points. Allowing 100 μsec per data point, .5625 seconds at most are required to access data. In the sequential approach only the required data points (i.e., those resulting from the design simulation) are accessed. For the case under consideration 256 logic units are to be mechanized, each with 6 inputs utilizing at most 1536 input points, so that only .1536 seconds are required to access data. In any event the access time is less than the .645 seconds available. Thus it is

*For example with the use of TI LS-600 NPN Planar Silicon light sensors having a center-to-center spacing of 0.1".

possible to employ a compact lightweight vidicon or image dissector tube to sequentially acquire data under the assumption that any point may be interrogated within 100 μsec.

An image dissector rather than a vidicon has been selected. A vidicon has the property that in reading out a data point the data value at that point is temporarily destroyed. On occasion two adjacent logic units may employ the same input data point, hence if the vidicon were employed and the same point interrogated in rapid succession the voltage representative of the light intensity would vary even though the input light intensity corresponding to the data point is constant. This disadvantage is overcome with the image dissector. In addition the vidicon requires from 1 to 4 watts of heater power which is not required by an image dissector. A further advantage of the image dissector tube is that the resolution of the tube is, to a first order approximation, controllable by the selection of tube aperture size. For these reasons an image dissector tube appears ideally suited as an optical-to-electrical transducer for a video pattern recognition system.

4.2.1 Image Dissector Tube

The schematic of an image dissector tube is shown in Figure 64. The photocathode of the tube releases electrons which are collimated by a magnetic focusing field in the drift tube. The direction of electron drift is dependent upon the magnetic focusing and deflection fields. An aperture permits electrons to pass into the photomultiplier. The area of the aperture is representative of approximately an equal area on the photocathode, the location of the selected photocathode area being dependent upon the horizontal and vertical magnetic deflection field. The electrons passing through the aperture are multiplied by a photomultiplier with a conventional dynode structure. The dissector tube selected is the ITT model F4011. The pertinent information concerning the tube and the associated deflection and focusing coil is specified in Table XVIII.

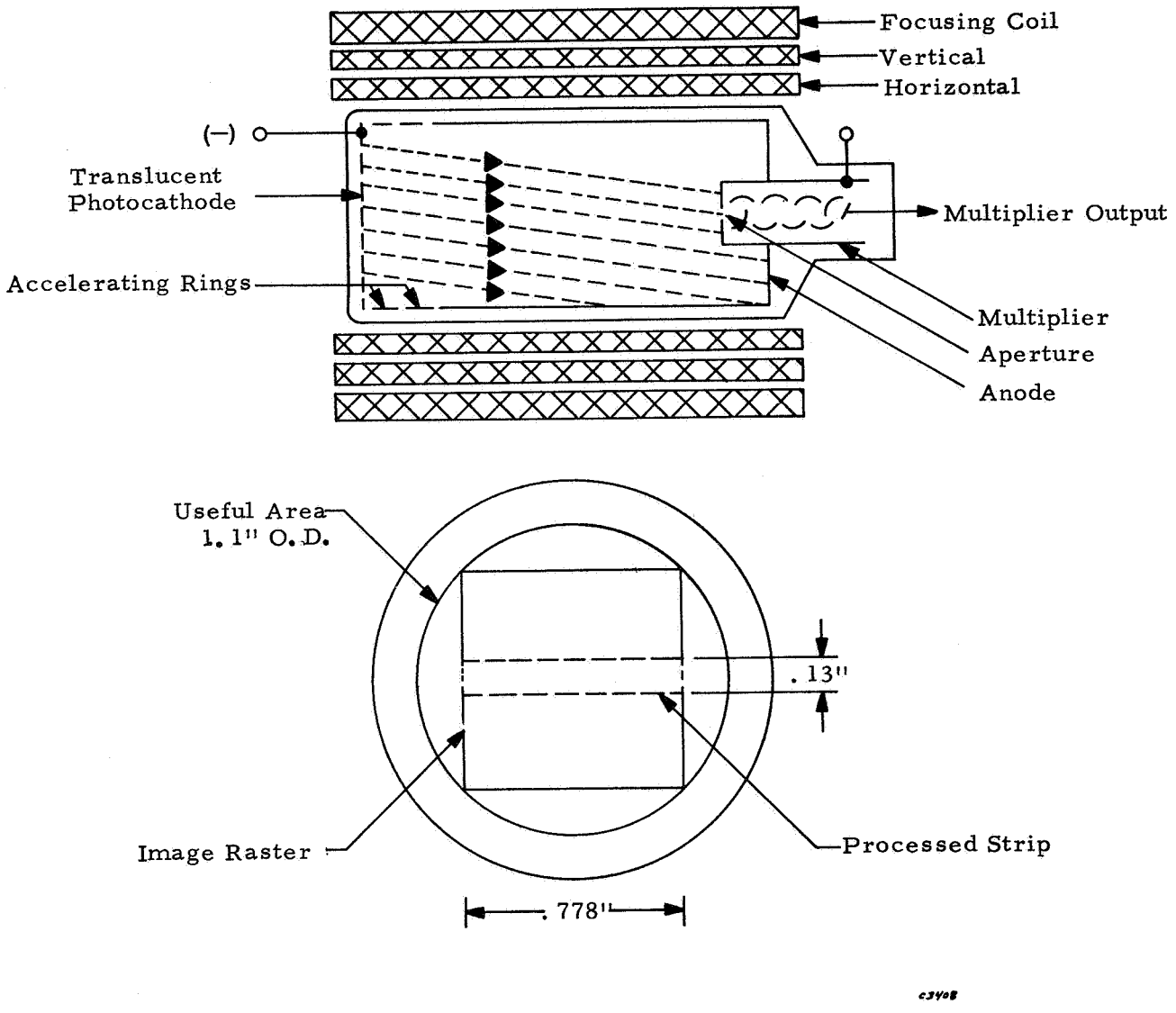


Figure 64. Image Dissector Tube

TABLE XVIII

DISSECTOR SPECIFICATION

Tube Diameter	1.5 inches
Useful Photocathode Diameter	1.1 inches
Selected Aperture = $\left(\frac{1.1''}{\sqrt{2}}\right)\left(\frac{1}{450}\right)$ =	1.724×10^{-3} inches
Anode-to-Photocathode Voltage	2400 volts
Photocathode Sensitivity (typical)	50μa/lumen
Photocathode Spectral Response	S11
Maximum Photocathode Current Density	10 μ a/cm^2
Photomultiplier Gain (typical)	2×10^6
F4508 Deflection Coil Sensitivity (typical)	150 ma/in
Inductance	3 mh
Resistance	11 ohms
F4506 Focusing Coil (voltage & current)	28 volts @ 40 ma
Overall Dimensions with Coils	8.2" x 3.1" O.D.
Weight with Coils	3.9 pounds

4.2.2 Input Optics

The focal length of the input lens required to give a ground coverage of 1050 by 1050 nautical miles from an altitude of 500 nautical miles corresponds to a lens having a 108-degree field-of-view. (162) Since the linear dimensions of the image dissector, for a square format contained within the useful area, is 0.778 inches, and since the image plane to lens distance for an object at infinity corresponds to the focal length, then for the case considered $f \simeq \frac{.778}{2} / \tan \frac{108^\circ}{2} = 0.389/\tan 54^\circ = 0.283$ inches or 7.2 mm. A controlable iris can be used to adjust the input light intensity if required, however for purposes of this discussion the lens will be used at an F number of 8.

4.2.3 Maximum Dissector Current

The dissector tube selected should not be operated at a photocathode density in excess of 10 $\mu a/cm^2$, however the output anode current from the dissector should be as large as possible under conditions of maximum illumination since the dissector signal-to-noise ratio improves in proportion to the square root of the output anode current. The maximum output from the sun above the earth's atmosphere at mean solar distance is 12,680 lumen/ft^2. (164) The maximum albedo or reflectance from the earth is 0.9. (164) Hence the maximum light flux density input to the lens is 11,410 lumens/ft^2. Since the lens is used at an F number of 8 and has a nominal transmittance of 0.7, then the illumination on the dissector is 31.2 lumen/ft^2. *

The useful area of the dissector is $\pi(0.55)^2$ $in.^2$ = 0.95 $in.^2$ = $(6.6)\ 10^{-3}$ ft^2 (or 6.1 cm^2), hence the light flux on the dissector is nearly 0.2 lumens. The photocathode sensitivity is specified by the manufacturer with respect to a tungsten source having a color temperature of 2870°K. The sun has a much higher color temperature and, based on the luminous efficiency curve, radiates about four times the power per lumen as the tungsten source. Since typical photocathode sensitivity of the dissector for the tungsten source is 50 μa/lumen it is estimated that sensitivity with the solar source will be about 200 μa/lumen. The maximum photocathode current is 40 μa (product of dissector light flux and photocathode sensitivity). Since the useful dissector area is 6.1 cm^2 the maximum photocathode current density is 6.6 $\mu a/cm^2$ or 42.1 $\mu a/in.^2$ Notice that this is 34% less than the

---

*The dissector illumination is computed from the equation: (165)

$$E = \frac{\pi}{4} \frac{B}{n_f^2 (1+m)^2} \cdot \tau$$

where πB = light flux density input to the lens (πB = 11,410 lumens/ft^2)

n_f = F number

m = linear magnification (m=0 for the case considered)

τ = transmittance of the lens = 0.7

maximum allowable current density as specified by the manufacturer. The selected dissector aperture diameter of .0017 inches (see Table XVIII) yields an aperture area of 2.34×10^{-6} in.^{2} Hence the current through the aperture is 98.5×10^{-6} μa. Multiplying this current by the typical photomultiplier gain of 2×10^6 yields a maximum output current of 197 μa.

4.2.4 Dissector Signal-to-Noise Ratio

The primary source of noise in an image dissector is due to the random shot noise present in the photocathode emission. The effect of the photomultiplier is to increase the noise coming through the aperture by a factor of 1.15 for the dissector being considered. The formula for shot noise [166] may be applied to the output current of the image dissector.

$$I_n = k\sqrt{2\, e\mu\, I_D \Delta f} \qquad (3)$$

where $k = \sqrt{\frac{\delta}{\delta - 1}} = 1.15$, δ = per stage gain of the photomultiplier

e = charge of one electron = 1.6×10^{19} coulombs

μ = overall gain of the photomultiplier = 2×10^6

I_D = average dissector output current

Δf = bandwidth of the system

The useful bandwidth of the system can be no greater than one half the sampling frequency, so that

$$\Delta f = \frac{f_s}{2} = \frac{1}{2\Delta t} \qquad (4)$$

where f_s = sampling frequency = 10K Hz

Δt = time duration between adjacent samples = 100 μsec

(3) may be expressed as

$$I_n = k\sqrt{\frac{e\,\mu I_D}{\Delta t}}$$

This yields

$$\frac{I_D}{I_n} = \frac{1}{k}\sqrt{\frac{I_D \Delta t}{e\mu}} \qquad (6)$$

At the maximum dissector current 197 μa the corresponding value of I_{D/I_n} is 216, an extremely good signal-to-noise ratio.

4.2.5 Deflection Amplifier Requirements

The selected deflection coils have an inductance of 3 millihenries and sensitivity with the F4011 of 150 ma/in. The sampling time is 100 μsec per data point. This places certain restrictions on the maximum voltage swing, output current and output load the deflection amplifier must drive. The requirements for the horizontal deflection amplifier will be considered since it represents the worst case. This amplifier must be capable of supplying enough current to cause a deflection of ± .389 inches. At the stated sensitivity this requires ± 58.4 ma. Each subsection corresponds to a length .13 inches on a side (see Figure 64) requiring a current of 19.5 ma. One resolution element corresponds to changing the current .26 ma. To cause a deflection to within 1/2 of a resolution element requires that

$$\Delta t = \frac{L}{R_L} \ln \frac{19.5}{.13} = 5L/R_L \qquad (7)$$

The resistance in series with the deflection coil, neglecting the internal resistance of the coil (11 ohms), must be at least

$$R_L \geq \frac{5L}{\Delta t} = 150\Omega$$

With this value the voltage corresponding to maximum deflection is ±8.75 volts.

The deflection amplifiers have the capability of summing several inputs. In the x-direction this allows subsection biasing. This feature offers a significant advantage in the sequential/hybrid and sequential/digital decision network implementation since it is then necessary to store

only seven bits of coordinate information (75 positions) rather than nine bits (450 positions) which would be required if periodic subsection biasing were not employed.

The Y deflection amplifier also has the capability of summation and allows correction to be made to cause incoming data to be retained in a fixed position independent of the satellite movement. This correction is made with the advancement to each new subsection. The correction corresponds to 0.9 of a sensory element for the parallel/analog system (35 millivolts at the deflection amplifier output). In the sequential/hybrid and sequential/digital approach the correction corresponds to approximately 1/3 of a sensory element or 13 millivolts at the deflection amplifier output.

4.2.6 D/A Converter

The function of the D/A converter is to allow data accessing by means of digitally stored numbers, or in the case of the parallel/analog system by means of counters. The D/A use a standard R/2R summation ladder and field effect transistor switches; each switch controlled by the state of a flip-flop register or counter.

4.2.7 A/D Converter

A simple four bit A/D converter is required by the sequential implementations. The converter encodes the output voltage of the image dissector for 16 equally spaced gray levels. Since it may be difficult to define the minimum and maximum gray level precisely and the gray level intensities are not necessarily linearly spaced, the system allows for 16 gray levels rather than eight. If it is found that extra quantization is not required, the last bit of the converter can be disregarded. A conversion method suitable for the sequential/hybrid and sequential/digital implementations is illustrated in Figure 65. The input impedance to the converter is made large relative to 25KΩ, the dissector anode load resistor. This may require a unity gain buffer amplifier. The full scale input to the converter corresponds to 5 volts (25K x 200μa). The unit does not require a sample and hold circuit since the data is sampled and held by virtue of the D/A

action on the image dissector. Read-out is always done after the input voltage has stabilized in both sequential/hybrid and sequential/digital systems. The converter is capable of encoding a zero to full scale change in 3.2 μ seconds.

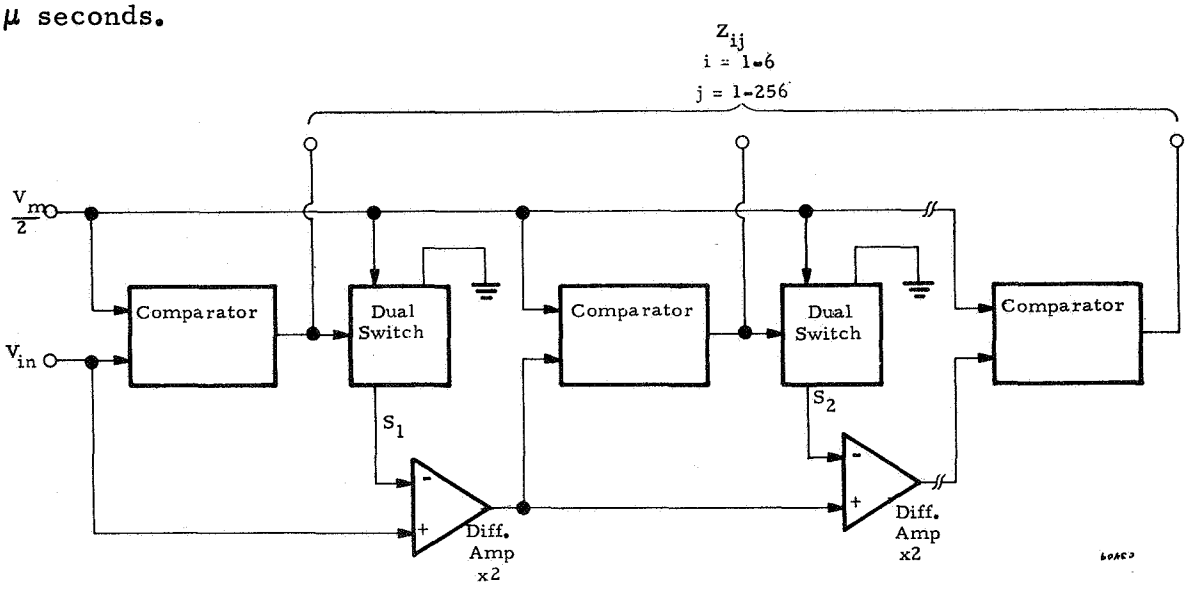


Figure 65. A/D Converter

4.3 Parallel/Analog System

A block diagram of this approach is shown in Figure 66. The input function and performance of the image dissector is as described in Section 4.2.1 and 4.2.3. The deflection amplifier performance has been described in Section 4.2.5 and the D/A's in 4.2.6.

The X counter advances from count one (binary zero) to count 75 (binary 74) while the Y counter remains fixed. At the end of X = 75 the Y counter advances from count one to count two. This process continues until all elements within a subsection have been interrogated. As this process is taking place the particular coordinate position being interrogated is decoded and is used to initiate sampling of one of the 1536 of the available 5625 points read-out. Only about 50X and 50Y locations are required to be decoded. The maximum size of the decoder network is 150, 7-input "AND" gates. Each

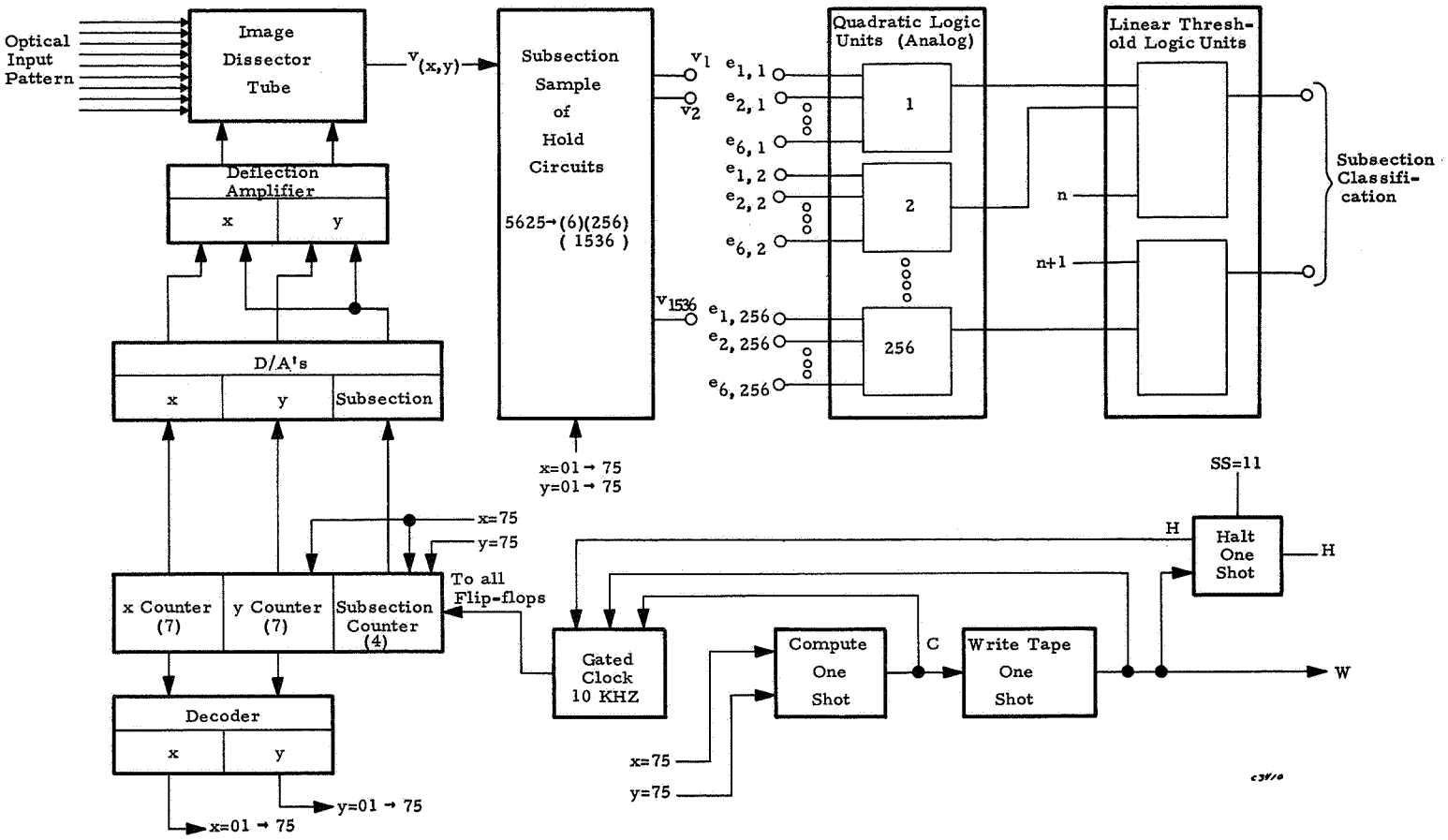


Figure 66. Visual Data Recognition System Using a Parallel/Analog Decision Network

sample and hold circuit contains a 2 input "AND" gate. Once the counter has advanced to the position X = 75, Y = 75 of a particular one of the eleven horizontally overlapping subsections the compute one-shot is fired. After a time delay of approximately 5 msec, a subsection classification appears at the output.

The compute one-shot causes the gated clock to be inhibited so that no counters advance during this interval. The compute one-shot's function is simply to allow time for all of the quadratic and response units to settle. When the compute one-shot turns "off," the read or write tape one-shot fires indicating to other equipment that a new subsection has been classified and the classification may be transmitted or stored on tape or both. The duration of the write one-shot is also set equal to 5 msec. Following the write tape one-shot's turn off, the gated clock is released and on the next clock pulse the X and Y counters return to X = 1 and Y = 1 and the subsection counter advances. On the occurrence of the eleventh subsection following turn off of the write one-shot, the halt one-shot is fired and remains in this state for a duration of 17.8 seconds halting further data processing. This allows time for the satellite to move sufficiently to cause 50% overlap of the next group of subsections to be classified. The clock is a 10 KHZ clock, hence the time required to acquire data will be 562.5 milliseconds whereas the time required to perform a classification and readout is only 10 msec. The response circuitry is arranged so that an analog voltage representative of the input sum of a response unit may be monitored.

4.3.1 Sample and Hold Circuits

The general structure of the j-th sample and hold circuit is shown in Figure 67. The major requirements for switch S is that it be a high speed switch, having a low "on" resistance and a high "off" resistance and be capable of bilateral current flow. The latter requirement is sometimes deleted at the expense of a shunt dumping switch in parallel with the capacitor C.

The overall sampling accuracy is dependent on the ratio of the hold time to the sample time for a fixed source and load resistance, and is independent of C.

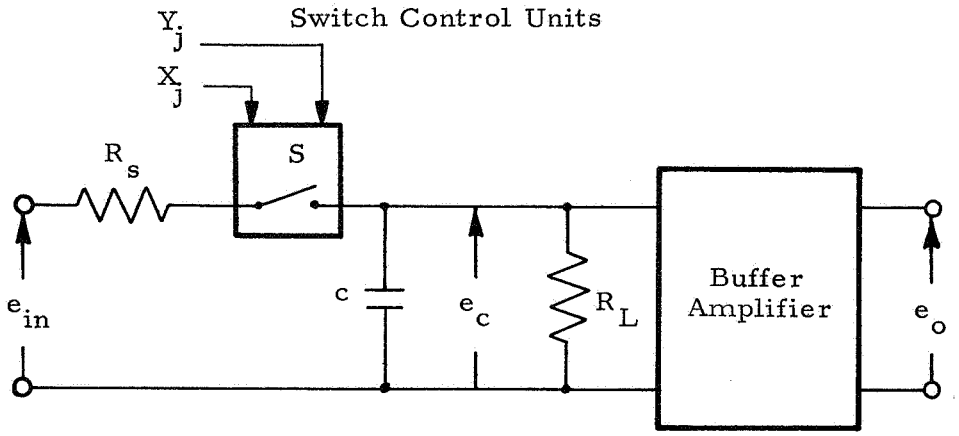


Figure 67. Typical Sample and Hold Arrangement

If

t_S = sampling time = dissector dwell time = 100 μsec

t_H = hold time = 572.5 msec

a = required sampling or hold accuracy

then

$$t_S = R_S C \ln \frac{1}{a}, \text{ and } t_H = R_L C \ln \frac{I}{1-a} \cong R_L C \cdot a$$

so that

$$\frac{t_H}{t_S} = \frac{R_L}{R_S} \cdot \frac{a}{\ln \frac{1}{a}}$$

Consider a system requiring 16 uniformly spaced gray levels, that is, a spacing between gray levels of 6.25%. Then it is reasonable to take a = 1.5% which gives a combined sampling and hold error of 3%. For the situation under consideration this gives $R_L/R_S = 1.6 \times 10^6$ Ω.

A reasonable value for a solid state switch is $R_S = 50$ ohms; this requires R_L = 80 megohms. Substituting the value of R_S or R_L yields C = .47 microfarads.

By constructing the buffer amplifier in a unity gain configuration and using a matched pair of field effect transistors in the input stage of the amplifier, it is possible to realize input resistances of greater than 100 megohms.

The power dissipated by the control gating circuits may be made sufficiently small so as to be neglected relative to the dissipation of the buffer amplifier. The buffer amplifier should be capable of accepting and outputting a zero to five volt signal. High level signals are conducive to good operation of the first layer logic units which these amplifiers supply. This indicates that it will be difficult to construct a buffer amplifier using less than 5 milliwatts.

4.3.2 Analog Quadratic Logic Unit

The performance of these logic units is described by equation (1), Section 4.1. A block diagram of the connective structure indicated by equation (1) is shown in Figure 68. All inputs (Z_1 through Z_6) are analog voltages.

Since 256 of these quadratic logic units are required by the recognition system, it is important to keep the weight and power of each as small as possible. Further, it is important to maintain the response time, t_R, of each unit as small as possible in order to obtain the high data processing rates normally associated with a parallel analog recognition system. The servo multiplier technique using ganged potentiometers operates accurately but has the disadvantages of large weight and size and a poor frequency response. The disadvantage of using the Hall generator technique is its weight and the power required to obtain a useful output. For example, a Hall multiplier using conventional toroidal magnets for a six input quadratic logic unit will weight approximately 3.0 pounds. A system requiring approximately 256 quadratic units would weight 0.4 tons.

A method that overcomes the disadvantages of servo-multipliers and Hall multipliers uses the logarithmic relationship between the transistor emitter current and the base to emitter voltage. This relationship is extremely uniform from transistor to transistor if the devices currently used for differential amplifier applications are employed. Figure 69 is a plot of V_{BE} vs. i_e for 1/2 of an MD1122. The graph was obtained from measurements on five differential transistors (10 junctions). The variation of the base to emitter voltage for any emitter current from 2μa to 100μa is less than $\pm$.020 volts. This variation will cause a maximum absolute error over the current range from 1μa to 1000μa of less than $\pm$5%.

The operation of the weighted analog multipliers used for the quadratic logic unit is described with the aid of Figure 70. Each weighted multiplier used by this logic unit is identical in its principles of operation. Initially v_x, v_y and v_B are shorted and the offset pot adjusted so that $v_{BE} = 0$. This accounts for any D.C. offset appearing across the input

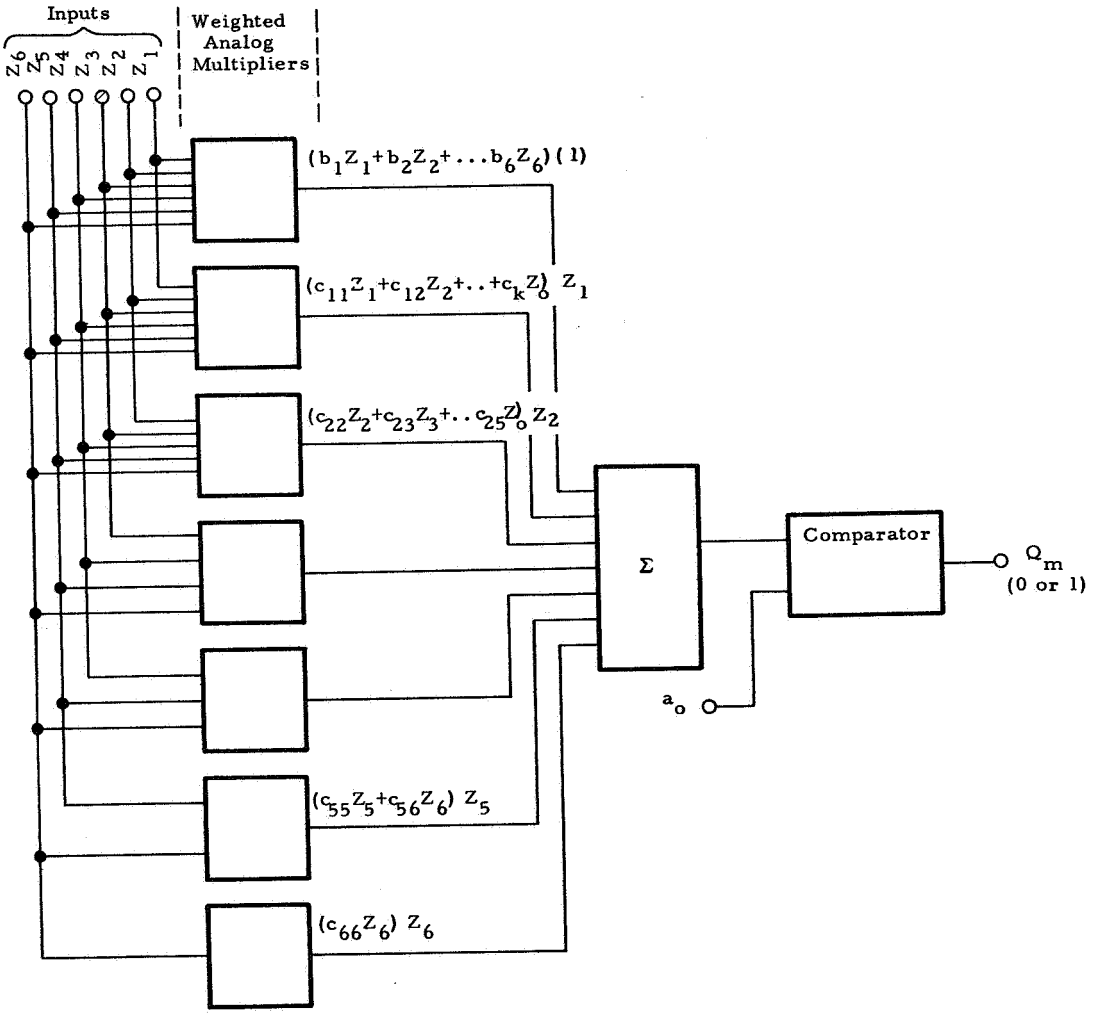


Figure 68. Quadratic Logic Unit Connective Structure

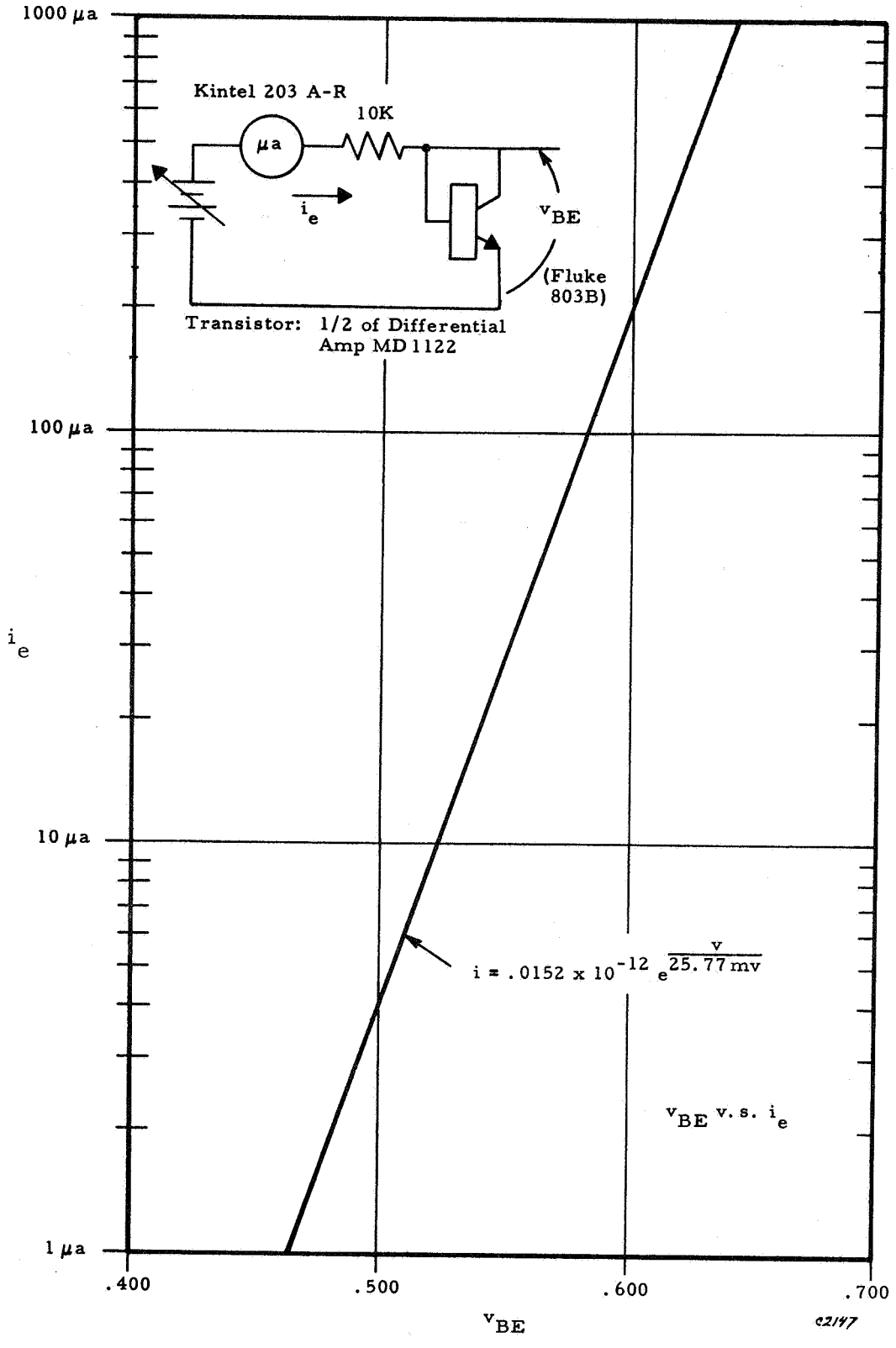


Figure 69. Plot of Emitter Current vs. Base to Emitter Voltage

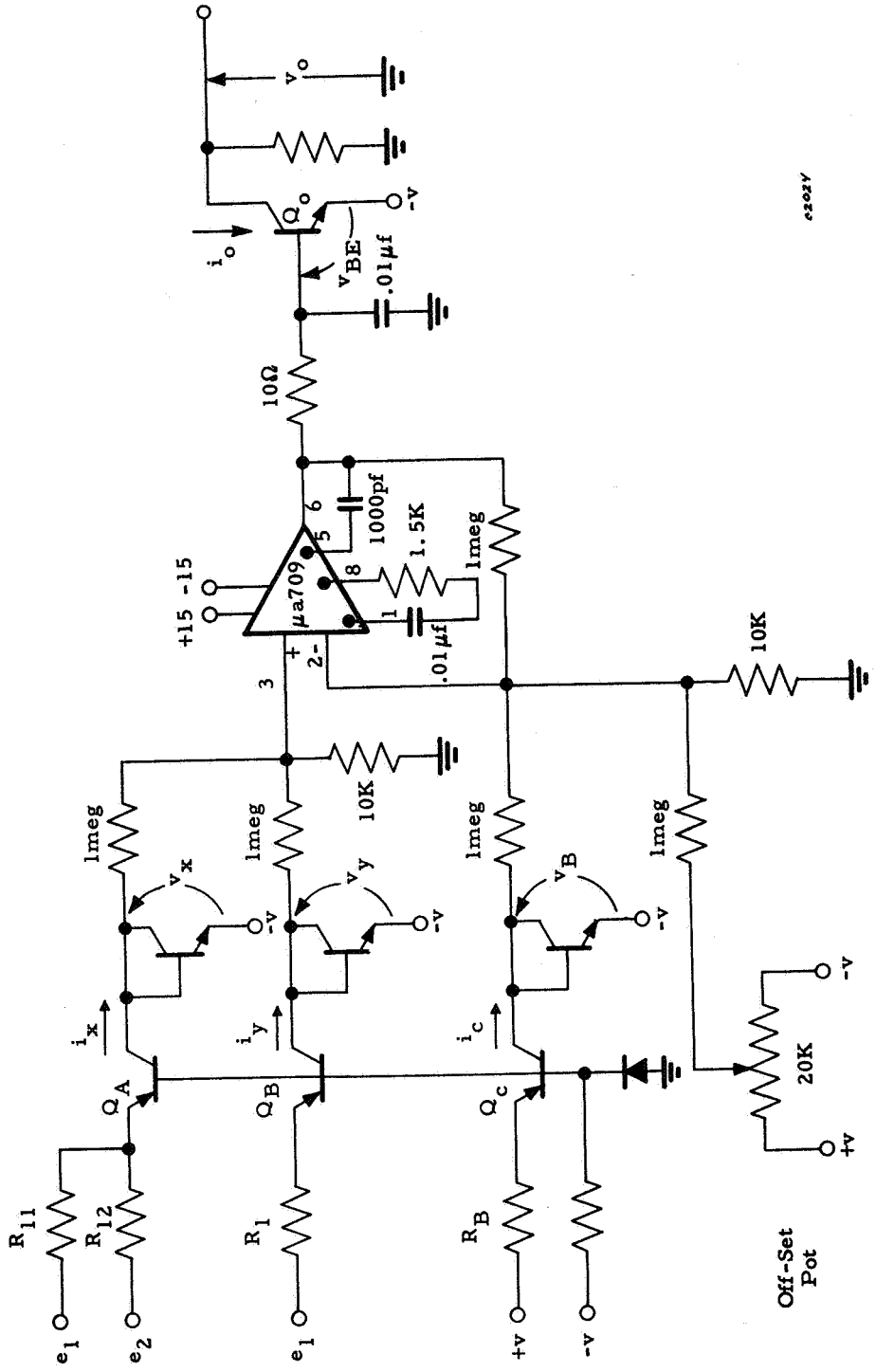


Figure 70. Weighted Multiplier Schematic

terminals of the μa709 integrated circuit amplifier. In an actual system, where size and weight are important the offset pot may be replaced by two fixed resistors.

The μa709 amplifier is connected as a differential amplifier so that

$$v_{BE} = v_x + v_y - v_B$$

This expression may be rewritten using the semilogarithmic relations between voltage and current to yield

$$\frac{KT}{q} \ln \frac{\alpha_o i_o}{i_R} = \frac{KT}{q} \ln \frac{i_x}{i_R} + \frac{KT}{q} \ln \frac{i_y}{i_R} - \frac{KT}{q} \ln \frac{i_c}{i_R}$$

or

$$i_o = \alpha_o \frac{i_x i_y}{i_c}$$

but

$$i_x = \alpha_a \left(\frac{e_1}{R_{11}} + \frac{e_2}{R_{12}} \right)$$

$$i_y = \alpha_B \frac{e_1}{R_1}; \quad i_c = \alpha_c \frac{v}{R_B}$$

and

$$v_o = -i_o R_L = -\alpha_o \frac{i_x i_y}{I_c} R_L$$

hence the output voltage is given by

$$v_o = -e_1 \left(\frac{\alpha_a \alpha_B \alpha_o}{\alpha_c} \right) \left(\frac{R_B R_L e_1}{R_{11} R_1 v} + \frac{R_B R_L e_2}{R_{12} R_1 \cdot v} \right)$$

If Q_a, Q_B, Q_c, and Q_o are high beta transistors having a $\beta_{min} = 100$ and a $\beta_{max} = 200$

then

$$.98 < \frac{\alpha_a \alpha_B \alpha_o}{\alpha_c} < .99$$

so that

$$v_o \cong -e_1 \left(\frac{R_B R_L}{R_{12} R_1} \frac{e_2}{v} + \frac{R_B R_L}{R_{11} R_1} \frac{e_1}{v} \right)$$

by equating like terms it is clear that

$$c_{ij} = \frac{R_B R_L}{R_{ij} R_1} \cdot \frac{1}{v}$$

Notice that if the voltage supplying the resistor R_1 is set equal to v, then linear terms of the quadratic logic are obtained with

$$b_i = \frac{R_B R_L}{R_{ij} R_1}$$

Negative weights are obtained by inverting the polarity of the corresponding voltage applied to R_{ij}, provided the sum of the input voltages is positive.

In designing this weighted multiplier it has been assumed that the $i_{Rx} = i_{Ry} = i_{RC} = i_{Ro}$. This condition can be met most readily if the transistors are mounted on a common heat sink or even to a higher degree if they are mounted on a common header. That is, a unit containing matched transistors mounted on a single T0-5 header.

To allow summing the individual products, see Figure 68, it is only necessary to mutually connect the output collectors of all the individual weighted multipliers and terminate these collectors in the common load resistor R_L.

The output comparator may be readily constructed using an integrated circuit μa 710 which is housed in a T0-5 case.

A six-input logic unit would require a volume of less than 7 cubic inches and weigh approximately 0.1 pound. The power dissipated by each logic unit using the conventional components listed above is approximate (80 milliwatts) (n+1). For n = 6 the dissipation is 0.56 watts. Future integrated circuit amplifiers could afford significant reduction in this power dissipation.

4.3.3 Linear Threshold Response Units

The response layer is constructed with linear threshold logic units. The equation defining the performance of these units is equation (2) of Section 4.1. The unit is constructed using an integrated circuit comparator and resistor summing network (see Figure 71). Inspection of the circuitry reveals that

$$w_m = \frac{R_p}{R_m} \qquad \text{where } R_p = \frac{1}{\frac{1}{R_1} + \frac{1}{R_2} + \cdots \frac{1}{R_n} + \frac{1}{R_\theta}}$$

$$\theta = (v)\,\frac{R_p}{R_\theta}$$

Negative weights are obtained by inverting the Q lines. The amplifiers are implemented using integrated circuits, one μA709 and one μA710.

4.3.4 Summary

Cloud cover classification from a satellite does not require a high classification speed; for this reason the quadratic unit redundancy is not justified. The major portion of the time used by this system to perform a subsection classification is spent accessing data points (562.5 msec). Given all data points a response is obtained from the decision network in less than 5 msec. The speed advantage of this network cannot be fully exploited without the use of a photosensor array. The parallel/analog system described in this section was designed to minimize size, weight, and power consumption. Speed was not the critical factor. However, if the speed requirements were critical the data coordinates could be stored in a diode matrix similar to that described in Section 4.4.2 allowing the unit to access only the 1536

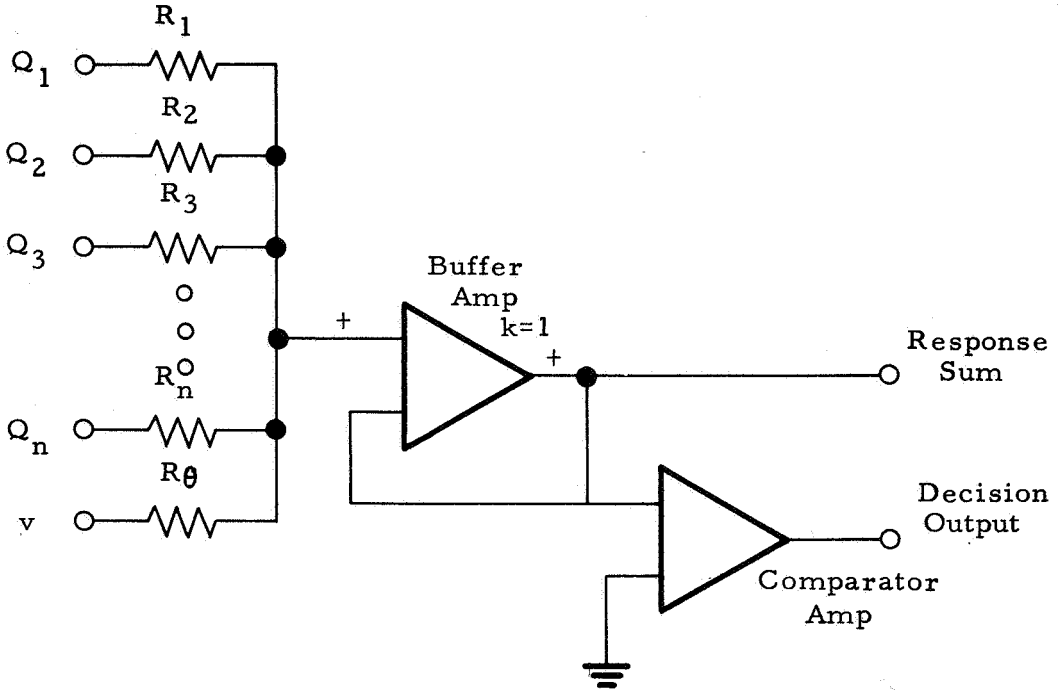


Figure 71. Linear Threshold Logic Unit

points required by the quadratic logic units. With this approach the time required to process a strip of eleven subsections could be reduced from 6.3 seconds to about 1.8 seconds with less than ten percent increase in weight and a slight increase in power consumption.

Table I on page xvii tabulates the estimated power, volume and weight as well as the recommended processing time/strip and duty cycle using this, the sequential/hybrid and the sequential/digital approaches.

4.4 Sequential/Hybrid System

The sequential/hybrid approach (Figure 72) accesses six data points, computes a quadratic unit's activity, adds and stores the response weights as determined by the quadratic unit's activity and then repeats the process until 256 quadratic units and the two resultant response sums have been generated.

Only one quadratic computing element is used. This unit must be provided with a means for changing the weights and the threshold to correspond to the particular quadratic unit being computed. This is accomplished by means of hybrid multipliers and a stored program.

The cyclic control of the system is as follows. Assume the lower and higher order binary counters (Figure 72) are at zero and the subsection counter of the input unit is at zero, so that c_o and t_o are true. This causes the stored program to read out the two response units' thresholds (40 bits) and the coordinates (X_1, Y_1) of the first data point of the first quadratic unit (14 bits). On the next clock pulse the threshold data is entered into the response sum registers and the location data are entered into the X and Y D/A registers of the input unit and C_1 and t_1 come true. The occurrence of t_1 causes the first data point to input to one of the six sample and hold circuits where it is retained in analog form. Also as a result of t_1's coming true on the next clock, the A/D converter output representing Z_1 will be transferred to a multiplier register contained by the quadratic logic computing circuitry. C_1 causes the second coordinate value of the first quadratic unit to be read-out of the stored program. The data accessing sequence proceeds. Then when C_5 comes true on the next clock pulse the sixth data location point of a quadratic logic unit is entered into the X and Y D/A register of the input unit. When C_6 and t_6 come true the last data point is retained by its associated sample and hold under control of t_6, and C_6 causes the 27 digital weights and a threshold to be read-out in parallel to the quadratic logic computing circuit. On the next clock pulse t_7 occurs and the digitized Z_6 is transferred to its respective quadratic logic register and the weights and threshold are load into their respective registers contained by the quadratic logic computing circuitry. Six bits including sign are used for digitally representing quadratic weights and threshold. During t_7 and C_7 all data inputs Z_1 through Z_6, in both analog

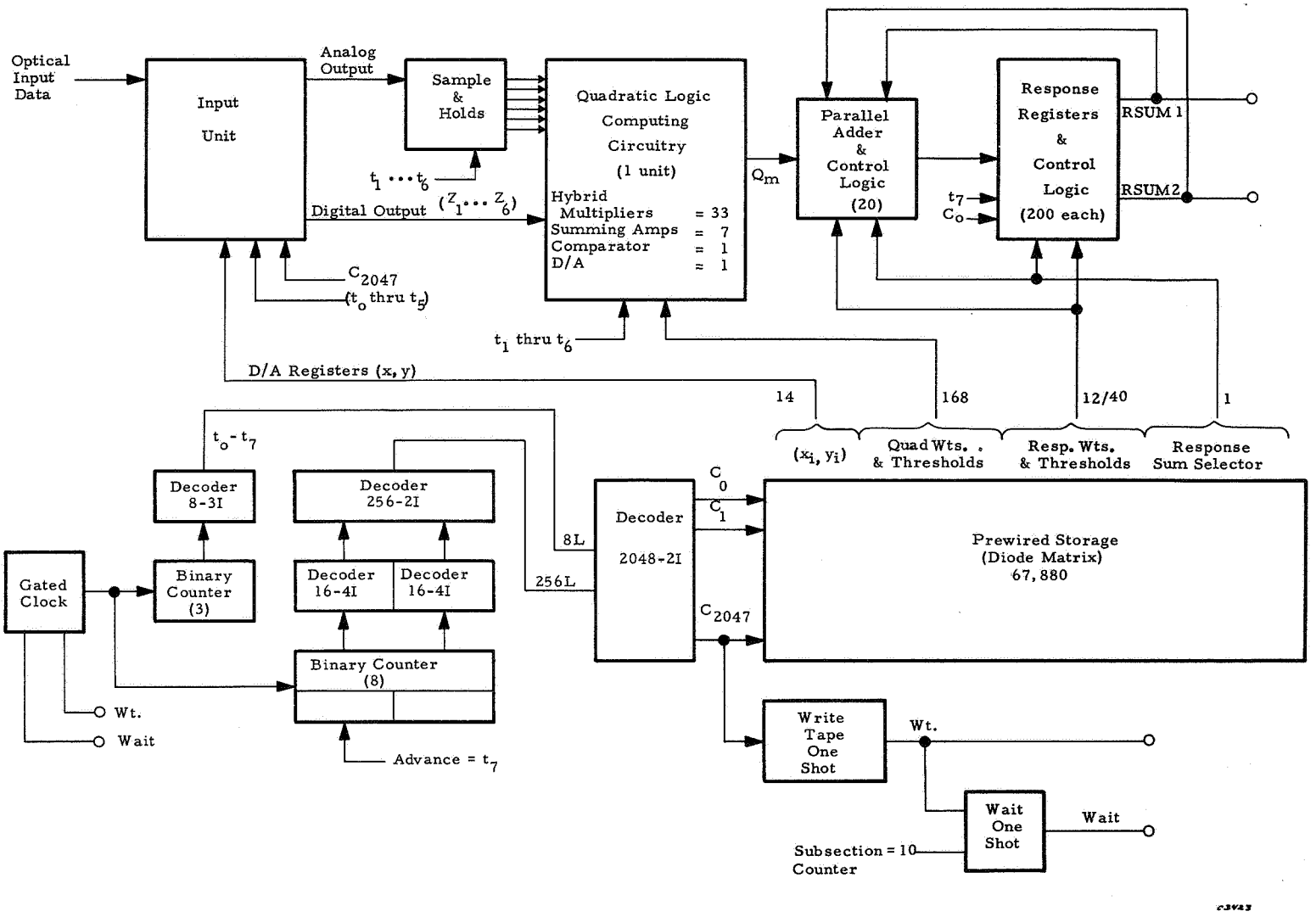


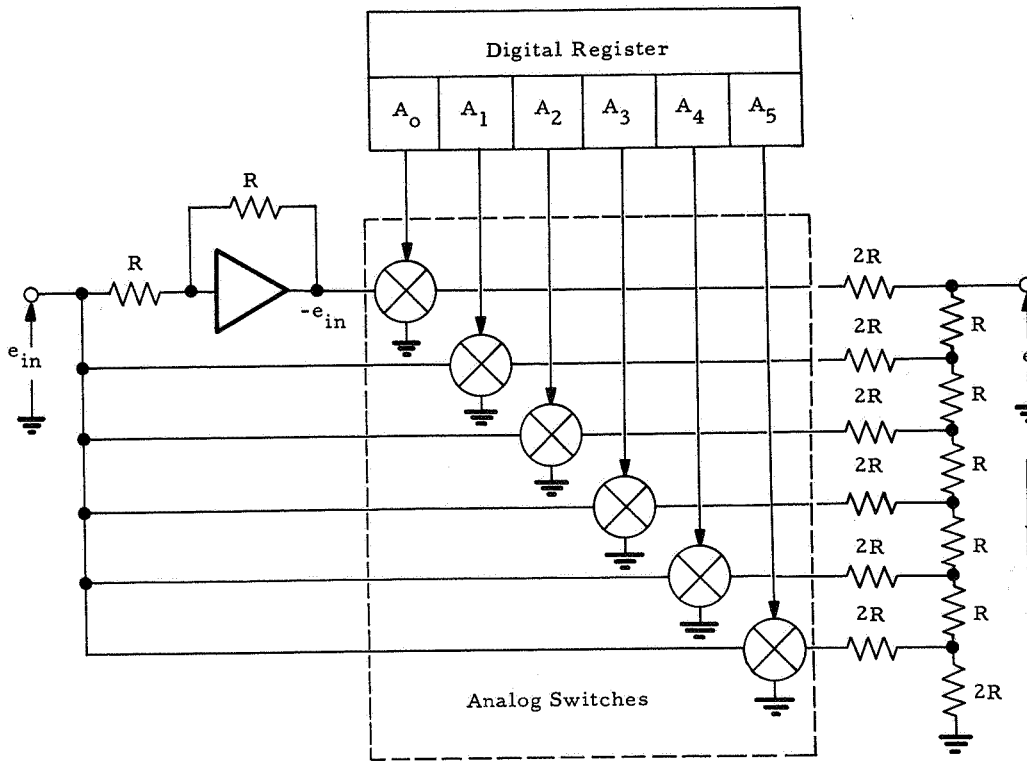
Figure 72. Visual Data Recognition System Using a Sequential/Hybrid Decision Network

and digital form, and the 27 weights and the threshold of the first logic unit are contained by or supplied to the quadratic logic computing circuitry. Thus during t_7 the activity of m^{th} quadratic unit Q_m is monitored. Due to the occurrence of C_7 the response weight corresponding to first quadratic unit is read from the stored program (12 bits) and one extra bit to define which of the two response sums is to be added to the response weight providing $Q_m = 1$. Following t_7 the lower order binary counter returns to t_o and the higher order binary counter advances while the resulting response sum is transferred to the appropriate response register. This sequence, except for inputting the response sum threshold, is repeated, 256 times. On the occurrence of the trailing edge of the last program count, C_{2047}, a write tape one shot is energized. This provides a signal indicating that a subsection has been classified and causes the gated clock to be momentarily inhibited and the subsection counter contained by the input unit to advance one count. In this fashion all 11 subsections of a strip are classified. On the occurrence of the trailing edge of C_{2047} and with the subsection counter equal to 10 the gated clock is inhibited until the next strip comes in view. Each clock interval corresponds to 100 μ sec. Neglecting the time used to readout response sums, the wait cycle will be 21.95 seconds. The time required to access and compute a subsection is .2048 second hence one strip corresponds to 2.25 seconds neglecting the time to readout each classified subsection.

4.4.1 Quadratic Logic Unit Computing Circuitry

The major building block of this computing element is the hybrid multiplier. This unit multiplies an analog signal by a digital coefficient. The circuit of a hybrid multiplier is shown in Figure 73. Each multiplier is essentially a D/A converter having a variable input voltage. Each of the six analog switches shown in Figure 73 is controlled by the output of the digital flip-flop register. The switches commutate either $\pm e_{in}$ or ground to respective bit positions of the D/A ladder. Negative digital multipliers are represented in two's complement.

Figure 74 illustrates the arrangement of the hybrid multipliers, summing amplifiers, D/A and comparator used to implement the quadratic logic unit. The inputs of the first column of multipliers are analog voltages originating from the six sample and hold circuits and digital weights from the stored program. The second layer multipliers use intermediate



$$e_o = e_{in}\left(-\frac{A_o}{2} + \frac{A_1}{4} + \frac{A_2}{8} + \frac{A_3}{16} + \frac{A_4}{32} + \frac{A_5}{64}\right)$$

where $A_i = 0$ or 1

	Digital Multiplier						Product
	A_o	A_1	A_2	A_3	A_4	A_5	e_o
31	0	1	1	1	1	1	(31/64) e_{in}
+1	0	0	0	0	0	1	(1/64) e_{in}
0	0	0	0	0	0	0	(0) e_{in}
-1	1	1	1	1	1	1	(-1/64) e_{in}
-31	1	0	0	0	0	1	(-31/64) e_{in}
-32	1	0	0	0	0	0	(-32/64) e_{in}

Note: Negative Digital Multipliers are Represented in Two's Complement

c3114

Figure 73. Hybrid Multiplier for Multiplying an Analog Signal Voltage by a Digital Coefficient

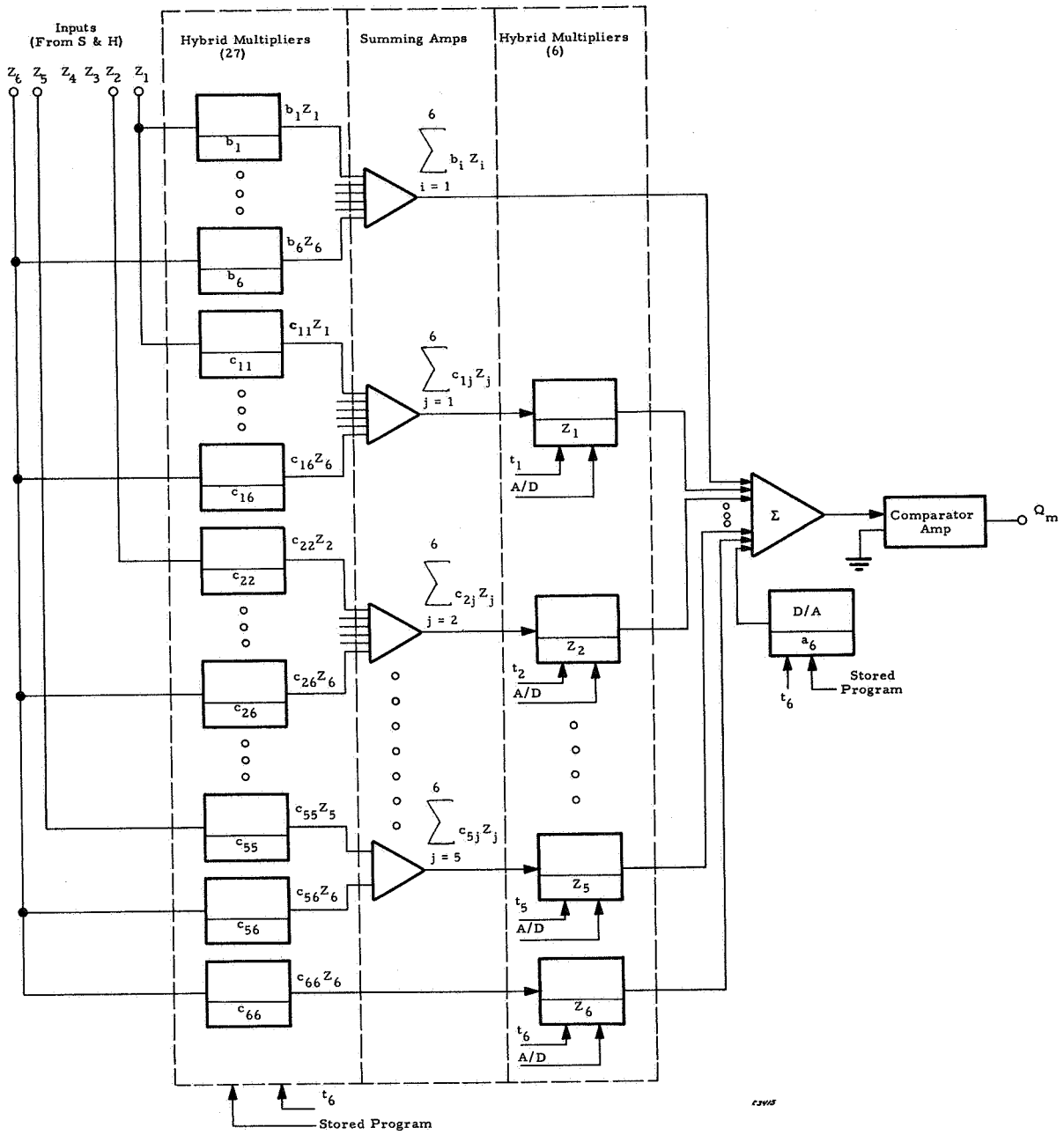


Figure 74. Quadratic Logic Computing Circuit Arrangement

sums for analog inputs and digital values corresponding to the Z's. Each of the 27 weights uses a six-bit register, each digital Z a four-bit register and the threshold a_o contained by the D/A register is six bits. The unit requires a total of 192 flip-flops and switches, 34 D/A ladder networks, and 35 integrated circuit amplifiers.

All quadratic weights are stored as six bits including sign. This weight accuracy is selected so the original data (four bits) will not be compromised. However with the quadratic weights selected the linear weights correspond to 10 bits including sign and the threshold corresponds to 14 bits including sign. In the prewired program linear weights and the threshold of the quadratic unit are represented as the most significant five bits plus sign and are rounded. The summing amplifiers in the quadratic unit computing circuit are therefore used to perform scaling as well as linear summation.

4.4.2 Prewired Storage

The storage medium selected for this implementation is a diode matrix. The diode matrix may be thought of as 1536 rows which interrogate the contents of 14 columns (X & Y data), 256 rows which interrogate 168 columns (a_o, b_i's and c_{ij}'s) and 256 rows which interrogate 13 columns (w_m and response sum selector). The matrix must provide locations for 67,880 diodes. The presence of a diode indicates a one in this bit position, absence a zero. Approximately 34,000 diodes are required to store the required data assuming an equal distribution of one's and zero's. Using miniature diodes such a memory could be constructed in volume less than 900 cubic inches.

This type of memory has the advantage that large or small blocks of stored data may be unloaded in parallel. This property can not be readily attained using conventional coincident current core memories since the word length of such memories is normally fixed at 40 bits or less. It can not be concluded on this basis that coincident core storage is not practical for the sequential/hybrid approach. Core storage with read/write capability could provide a method for storing all the response sums in an orbit (approximately 2772 response sums, 20 bits each-maximum). This capability cannot be obtained using the read only diode matrix. In addition

recognition system design changes may be made readily by reading new weights, thresholds and data coordinates into the core memory. However, the diode matrix does provide economy when compared to coincident current core storage. Other forms of prewired storage could have been employed such as permanently wired transformer coupled storage or ferrite rods. The diode matrix provides significant simplicity relative to other memory techniques.

4.4.3 Input Unit

The input unit uses the image dissector, deflection and focusing coils, deflection amplifiers, with X, Y and subsection D/A's previously described (Section 4.2). The input unit contains a four bit A/D as previously described (Section 4.2). Data accessing is done by directly loading the X and Y D/A registers, rather than counting as in the parallel/analog approach. The subsection D/A register acts as a counter and is advanced as a result of C_{2047}.

4.4.4 Parallel Adder and Control Logic

The inputs to the parallel adder are called A_o through A_{19} and M_o through M_{19}. The adder output sum is designated S_o through S_{19}. The least significant bit position corresponds to S_o. The sign bit is contained in S_{19}. Negative numbers (A and M lines) are presented to the adder in two's complement.

Let $RS1_o$ through $RS1_{19}$ and $RS2_o$ through $RS2_{19}$ represent the contents of the first and second response sum registers respectively. Let RU be the bit indicating which response unit is to be input to the adder. This bit is supplied by the stored program. Then in Boolean form.

$$A_o = (RS1_o \cdot RU) + (RS2_o \cdot \overline{RU})$$

$$A_{19} = (RS1_{19} \cdot RU) + (RS2_{19} \cdot \overline{RU})$$

The other inputs to the adder (M lines) are conditioned by Q_m and the twelve bit response weight w_{mo} through w_{m11}

$$M_o = w_{mo} \cdot Q_m$$

$$\vdots$$

$$M_{10} = w_{m10} Q_m$$

$$M_{11} = w_{m11} Q_m$$

$$\vdots$$

$$M_{19} = w_{m11} Q_m$$

Notice that sign of the response weight, w_{m11} is spread from M_{11} through M_{19}. This allows the 20 bit response sum to be added in a standard fashion to the 12 bit response weight.

The adder performs in a conventional manner.

$$S_o = A_o\overline{M}_o + \overline{A}_o M_o \qquad C_1 = A_o M_o$$

$$S_1 = A_1M_1C_1 + \overline{A}_1\overline{M}_1C_1 + \overline{A}_1M_1\overline{C}_1 + A_1\overline{M}_1\overline{C}_1 \qquad C_2 = A_1M_1 + A_1C_1 + M_1C_1$$

$$\vdots \qquad\qquad \vdots$$

$$S_{19} = A_{19}M_{19}C_{19} + \overline{A}_{19}\overline{M}_{19}C_{19} + \overline{A}_{19}M_{19}\overline{C}_{19} + A_{19}\overline{M}_{19}\overline{C}_{19}$$

The size of the weights prevent overflow in the sum, however overflow can be checked providing the signs of the two numbers to be added are alike. Overflow can not occur with opposite signs. The logic for such an overflow check is

$$\text{overflow} = \overline{A}_{19}\overline{M}_{19}S_{19} + A_{19}M_{19}\overline{S}_{19}$$

4.4.5 Response Register Logic

The purpose of this logic is to load the response registers with their respective thresholds at the proper time and to enter the adder sum into the register dictated by the response unit selector bit (RU) at the appropriate time. Accordingly the response register's logic is

$$1^{RS1_o} = S_o(RU)t_7 + (\theta 1_o)C_o \qquad O^{RS1_o} = \overline{S}_o(RU)t_7 + (\overline{\theta 1}_o)C_o$$

$$\vdots \qquad\qquad \vdots$$

$$1^{RS1_{19}} = S_{19}(RU)t_7 + (\theta 1_{19})C_o \qquad O^{RS1_{19}} = \overline{S}_{19}(RU)t_7 + (\overline{\theta 1}_{19})C_o$$

$$1^{RS2_o} = S_o(\overline{RU})t_7 + (\theta 2_o)C_o \qquad O^{RS2_o} = \overline{S}_o(\overline{RU})t_7 + (\overline{\theta 2}_o)C_o$$

$$\vdots \qquad\qquad \vdots$$

$$1^{RS2_{19}} = S_{19}(\overline{RU})t_7 + (\theta 2_{19})C_o \qquad O^{RS2_{19}} = S_{19}(\overline{RU})t_7 + (\overline{\theta 2}_{19})C_o$$

4.4.6 Counters and Decoders

The lower and upper counters of the system count in a straight binary fashion. Since this logic is quite widely known the set/reset equations for these flip-flops will not be discussed. The lower counter produces binary counts 0 through 7. The upper counter yields binary counts 0 through 255. As shown in the block diagram the decoding requires the following "AND" gates

Quantity	No. of Inputs
8	3
32	4
2304	2

This gating may be easily accomplished using conventional diode logic due to the low clock rate.

4.4.7 Sample and Hold

The sample and hold circuits are implemented as previously described in Section 4.3.1 of the parallel/analog system, except that sample control is accomplished with a single input timing level (t_1 through t_6) rather than a pair of timing levels corresponding to an accessed coordinate (X_1 Y_1 through $X_{75}Y_{75}$).

4.4.8 Summary of the Sequential/Hybrid Approach

A summary of the estimated size, weight and power and required processing time per strip and system duty cycle is given in Table XVII, Section 4.1 based on the preceding preliminary design.

This approach appears highly effective provided that the decision network uses only the statistical type property filters.

The results of the study are highly dependent upon the memory selected. If the quadratic weights and thresholds are loaded four at a time into the quadratic unit computing circuit, while a data point is simultaneously accessed, a core memory could be used with a 38-bit word length using nearly the identical counter control arrangement without requiring the decoders. Such a system would yield reduced size and extend the flexibility of the system if the core memory used has a write capability. In addition the write capability would provide a direct means for electronically transferring the results from the computer simulation to the on board sequential/hybrid recognition system.

The digital registers of the hybrid multipliers are loaded with fixed weights from the diode matrix in order to implement the sequential/hybrid system. If these registers were, in addition, provided with an up/down counting capability, the weights and thresholds could be modified as specified by a training algorithm. This type of modified hybrid multiplier could prove quite useful for future applications requiring self-learning recognition systems.

4.5 Sequential/Digital

Figure 75 is a block diagram of the sequential/digital system, The system may be divided into two parts, a compact general purpose computer and an input unit. The GP computer is used to control the input unit and to compute and store the 22 responses associated with a strip (11 subsections). The GP is divided into three component parts — a memory, an arithmetic unit and a control unit.

Many general purpose computers have been designed, developed and are available commercially. The term general purpose as used here

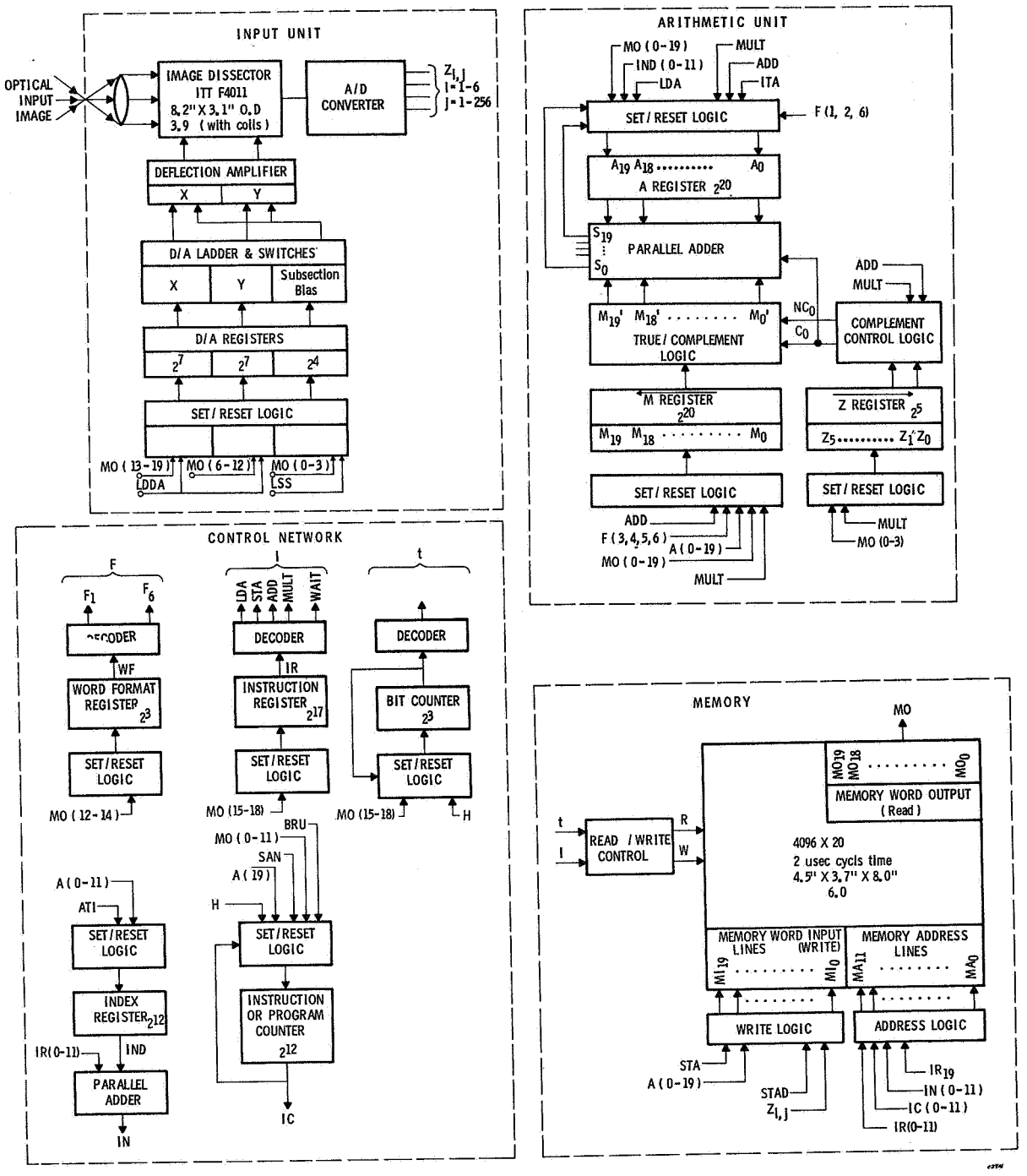


Figure 75. Visual Data Recognition System Using a Sequential/ Digital Decision Network Implementation

refers to the fact that the sequence of the operations executed can be readily changed by altering a stored program. On the other hand a computer designed to be particularly efficient for a special problem using a special input medium is normally considered to be a special purpose computer. The objective in performing a preliminary design of a computer, rather than using available commercial machines, is to specify clearly the minimal hardware required for the special problem at hand and to design a computer compatible with the input unit and capable of operating in real time.

The factors considered which yield the minimal computer hardware are (1) operations may be performed using only fixed point arithmetic, (2) the only arithmetic operations required are addition and multiplication, (3) the multiplier is always 4-bits representing the encoded 16 grey levels, (4) the memory is packed to yield a minimum storage capacity, and (5) the minimal control unit requires only 13 instructions.

This system uses a 2μsec clock. All registers are altered only on occurrence of a clock pulse yielding a synchronous computer. To obtain the 2μsec clock, a 1μsec crystal controlled oscillator is driven into a toggle flip-flop. The normal clock differentiates the positive-going output of the true toggle flip-flop output. The false side of this flip-flop is differentiated on the positive-going output to yield and out-of-phase clock.

4.5.1 Input Unit

The image dissector; deflection and focus coils; deflection amplifiers; D/A ladders, switches and registers; and four-bit A/D converter are implemented as described in Section 4.2. Data are accessed under program control. The data access rate corresponds to 100$\pm$2μsec/point as in the other implementations considered.

The X, Y, and subsection registers are loaded directly from the memory output under control of the two instructions LDDA and LSS. The memory output register lines are designated MO_o through MO_{19}.

Accordingly the set/reset equation for the X, Y and subsection register, SS, are

$$1^{X_1} = (MO_{13})(LDDA)t_6; O^{X_1} = (\overline{MO}_{13})(LDDA)t_6; 1^{Y_1} = (MO_6)(LDDA)t_6; O^{Y_1} = (\overline{MO}_6)(LDDA)t_6$$

$$1^{X_2} = (MO_{14})(LDDA)t_6; O^{X_2} = (\overline{MO}_{13})(LDDA)t_6; 1^{Y_2} = (MO_7)(LDDA)t_6; O^{Y_2} = (\overline{MO}_7)(LDDA)t_6$$

$$\vdots \qquad \vdots \qquad \vdots \qquad \vdots$$

$$1^{X_{64}} = (MO_{19})(LDDA)t_6; O^{X_{64}} = (\overline{MO}_{19})(LDDA)t_6; 1^{Y_{64}} = (MO_{12})(LDDA)t_6; O^{Y_{64}} = (\overline{MO}_{12})(LDDA)t_6$$

$$1^{SS_1} = (MO_0)(LSS)t_6 \qquad O^{SS_1} = (\overline{MO}_0)(LSS)t_6$$

$$\vdots \qquad \vdots$$

$$1^{SS_4} = (MO_3)(LSS)t_6 \qquad O^{SS_4} = (\overline{MO}_3)(LSS)t_6$$

The X and Y flip-flops are labeled so the sum of the subscripts of the flip-flops in the true state may be added to yield the associated rectangular coordinates of a point in a subsection.

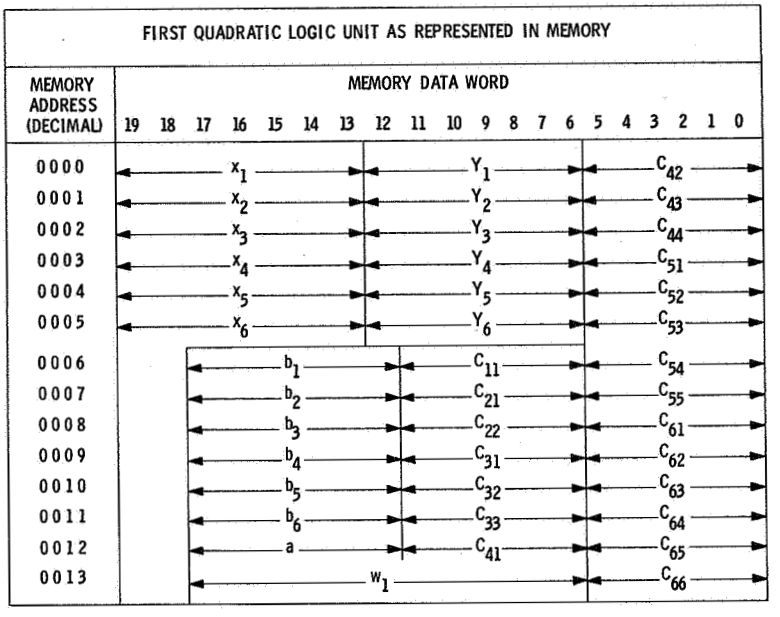
4.5.2 Memory

The memory consists of 4096 storage locations in which information can be stored and from which information can be extracted. Each storage location is 20 bits long. The information is of two types. Numbers representing the weights, thresholds and coordinate points are stored in locations 0 through 3583. Locations 3584 through 3820 (257 words) contain the program, and storage for the results of the program (22 locations for response sums). In addition, this section of memory allows for the storage of 12 accessed data points (2 quadratic units). The last 275 locations are free and could be used for such things as a magnetic tape writing subroutine or outputting response sums to a transmitter. The contents of a word in memory locations 0 through 3583 remain unchanged; that is, in reading the contents

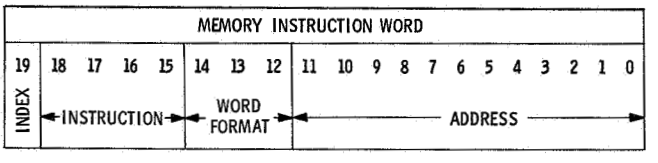
of these locations the information is automatically restored. Figure 76 illustrates the approach used in packing the 256 quadratic units and corresponding response weight into locations 0 through 3583 and also shows the instruction word format. Notice that storage of the data required to generate one quadratic unit and augment the associated response unit requires 14 memory locations. The memory instruction word is divided into two parts — an operation code and an address. The operation code contains the index bit, the instruction and the word format. The instruction portion represents the operation to be obeyed, i.e., addition, multiplication, etc. A one in the index bit indicates that the contents of the index register are to be added to the address before accessing memory (Figure 76). The word format specifies the number and position of the bits in the memory word on which operations are to be performed. The various instructions, clock pulses per instruction and function of the instructions are listed in Figure 77.

The memory unit selected is produced by Electronic Memories Inc. and designated by the manufacturer as the SEMS 5. The SEMS 5 (Severe Environment Memory System) is a small, fast, lower power, light weight, coincident current core memory. As the memory title implies, it is designated to withstand severe shock, vibration, humidity and temperature. The major specifications of the SEMS 5, as used, appear in Table XVIV.

The time used to process and classify a strip (11 subsections) is 2.45 seconds, using the program described in Section 4.5.7, and the relatively fast SEMS 5 2μsec cycle time as the system clock rate. The time between strips was previously estimated as 24.2 seconds; therefore, the memory is operating only 10.1% of the time and is in standby the remaining time. The relatively high speed of the memory provides a method for reducing power dissipation which can not be achieved using other slower magnetic memory forms, such as a drum, disc or magnetic tape unit.



ALL 256 QUADRATIC LOGIC UNITS AND CORRESPONDING RESPONSE WEIGHT ARE CONTAINED BY LOCATIONS 0 THRU 3583.



PROGRAM, RESPONSE SUMS (22 LOCATIONS), TZ_1 THRU TZ_6, AND Z_1 THRU Z_6 ARE CONTAINED BY LOCATIONS 3584 THRU 3820.

MEMORY WORD FORMATS	
BINARY CODE	DATA IN THIS FORMAT
0 0 0	X_1, Y_1 THRU X_6, Y_6 ($MO_6 \rightarrow MO_{19}$)
0 0 1	C_{11} THRU C_{41} ($MO_6 \rightarrow MO_{11}$)
0 1 0	C_{42} THRU C_{66} ($MO_0 \rightarrow MO_5$)
0 1 1	b_1 THRU b_6 ($MO_{12} \rightarrow MO_{17}$)
1 0 0	a ($MO_{12} \rightarrow MO_{17}$)
1 0 1	w ($MO_6 \rightarrow MO_{17}$)
1 1 0	ALL FULL-WORD DATA ($MO_0 \rightarrow MO_{19}$)
1 1 1	Z_1 THRU Z_6, TZ_1 TZ_6, LSS ($MO_0 \rightarrow MO_3$)

c34/17

Figure 76. Memory Data Packing and Memory Instruction Word

OCTAL CODE	MNEMONIC	INSTRUCTION	COMMENTS	CYCLE TIMES
00	LDA	LOAD A	MEMORY → A REGISTER	2
01	STA	STORE A	A REGISTER → MEMORY	2
02	SAN	SKIP IF A NEGATIVE	IC → IC + 1	2
03	BRU	BRANCH UNCONDITIONALLY	MEMORY ADDRESS → IC	2
04	ATI	A TO INDEX	A REGISTER → INDEX REGISTER	2
05	ITA	INDEX TO A	INDEX REGISTER → A REGISTER	2
06	LDDA	LOAD D/A REGISTER	MEMORY → D/A	2
07	LSS	LOAD SUBSECTION REGISTER	MEMORY → SUBSECTION REGISTER	2
11	ADD	ADD	MEMORY → M, M + A → A	3
12	MULT	MULTIPLY	A → M, 0 → A, MEMORY → Z, (Z)(M) → A	7
14	STAD	STORE A/D	CONVERTER → MEMORY	2
15	WRITE		MEMORY → TAPE UNIT	
13	WAIT			

c3418

Figure 77. Instruction List

TABLE XVIV

SPECIFICATION OF THE SEMS 5 MEMORY UNIT

Speed	2 μsec cycle time .6 μsec access time
Temperature	-55 to +85°C
Shock & Vibration	meets applicable portions of MIL-E-5400
Capacity	4096 word, 20 bits each
Weight	6 pounds
Volume	132 cubic inches
Dimensions	4.5" x 3.68" x 8.00"
Read or Write Initiate Pulse	.2 $\pm$ .1 μsec
Voltage & Current Requirement	(@ 20 bits/word)

Voltage ($\pm$ 2%)	Standby Current (amps)	Operating Current (amps)
+15	.06	.80
- 5	.01	3.80
+ 5	.80	.80

Standby Power Required	4.95 watts
Operating Power	25.0 watts maximum
Operating Power (10.1% duty cycle)	8.00 watts maximum

4.5.2.1 Memory Address Logic

The memory may be addressed from the address portion of an instruction word (IR_0 through IR_{11}), from the instruction or program counter (IC_0 through IC_{11}), or by means of the index lines (IN_0 through IN_{11}). Instruction counter outputs are input only during t_7, the time the memory output will contain an instruction. If indexing is to be used, the index bit, IR_{19} of the instruction word, will be true. The memory address lines are called MA_0 through MA_{11}. The memory address logic is

$$MA_0 = (IR_0\overline{IR}_{19} + IN_0IR_{19})\overline{t}_7 + IC_0t_7$$

$$\vdots$$

$$MA_{11} = (IR_{11}\overline{IR}_{19} + IN_{11}IR_{19})\overline{t}_7 + IC_{11}t_7$$

4.5.2.2 Memory Write Logic

Information is written into memory either from the A/D converter (Z_1, Z_2, Z_4, Z_8) output on the occurrence of the instruction store the A/D converter output (STAD) or from the A register (A_0 through A_{19}) of the arithmetic unit under control of the instruction store the A register (STA). The input lines which govern the information to be written into a selected memory location are MI_0 through MI_{19}. Accordingly the logic for these lines is

$$MI_0 = (A_0)\,(STA) + (Z_1)\,(STAD)$$
$$MI_1 = (A_1)\,(STA) + (Z_2)\,(STAD)$$
$$MI_2 = (A_2)\,(STA) + (Z_4)\,(STAD)$$
$$MI_3 = (A_3)\,(STA) + (Z_8)\,(STAD)$$
$$MI_4 = (A_4)\,(STA)$$
$$\vdots$$
$$MI_{19} = (A_{19})\,(STA)$$

4.5.2.3 Read/Write Control

It is required to read data from the memory at three different times. If a multiply is to be executed, data must be read out of the memory during t_1. When an add is to be executed, data must be read from the memory during t_5. For the instructions load the A register (LDA), load the D/A registers (LDDA), load the subsection register (LSS) and write out the response sums (WRITE), data is read out of memory during t_6. Except during the wait phase, a new instruction is always read out during t_7. Accordingly the read logic

$$READ = (MULT)t_1 + (ADD)t_5 + (LDA + LDDA + LSS + WRITE)t_6 + t_7 \overline{WAIT}$$

This logic is used to trigger a one-shot with a .2μsec duration. The one-shot output pulse is delayed .2μsec after the read line comes "up." The output of this one-shot is connected directly to the read initiate line of the memory. It is important to note that memory output lines will represent the accessed data or instruction within .8μsec from the occurrence of a normal clock pulse. This allows ample time for adding the new memory address in the instruction register to the contents of the index register or copying and instruction or data into the appropriate registers on the following normal clock pulse. Note that 1.2μsec occur after the memory output lines are "up" before occurrence of the next clock pulse. In a sense the memory is asynchronous with the rest of the system.

Only two situations occur where it is desired to write-store the A register (STA) or store the converter output (STAD).

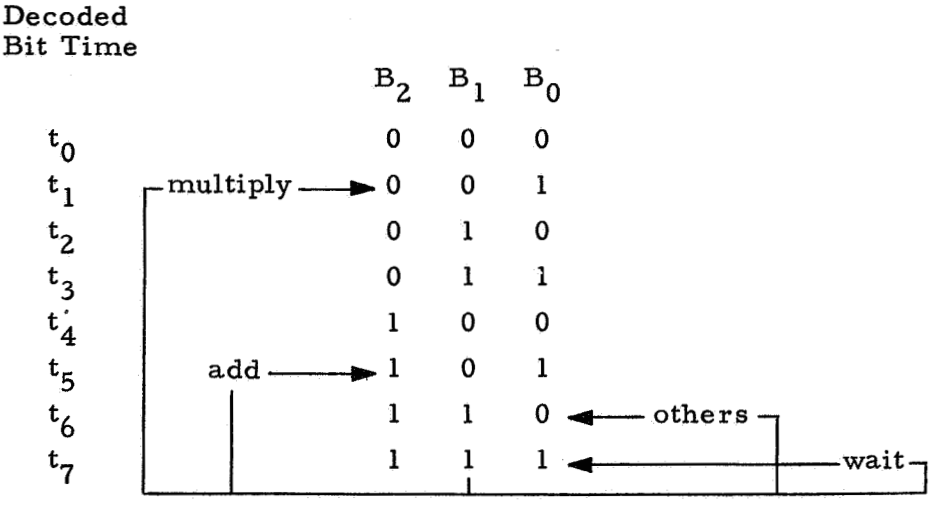
$$WRITE = (STA + STAD)t_6$$

4.5.3 Control Unit (Figure 75)

The control unit provides the timing for the system via the bit counter and contains the instruction counter, the instruction register, the word format register, the index register, index adder and the bit counter.

4.5.3.1 Bit Counter and Decoder

The bit counter counts in a normal binary fashion except that it is preset by the instruction to be executed. The duration to which the bit counter is preset always corresponds to the time required to execute the current instruction and fetch the next instruction. A diagram of the counter operation is shown below.



Before proceeding with the bit counter logic it is necessary to define H. H is a line that causes the bit counter to hang-up in t_7. This line will come true only on the occurrence of a wait instruction and if the new strip of data has not come into view ($\overline{START} = 1$) or if a wait instruction occurs and the response sums have not as yet been read ($\overline{RSR} = 1$). Hence

$$H = (WAIT)(\overline{START} + \overline{RSR})$$

If H occurs it causes the bit counter to freeze or hang-up in t_7. Presetting of the bit counter is always done as a result of t_7 occurring. In presetting, the counter is forced to t_6 unless an add or multiply is to occur as indicated by $(\overline{MO}_{18} + MO_{17})t_7$. The resultant set/reset logic of the bit counter (B_0 least significant bit) is,

$$1^{B_0} = \overline{B}_0\overline{H}\,\bar{t}_7 + t_7(\overbrace{MO_{18}\overline{MO}_{17}\overline{MO}_{16}MO_{15}}^{\text{add}} + \overbrace{MO_{18}\overline{MO}_{17}MO_{16}\overline{MO}_{15}}^{\text{mult.}}) + H$$

$$0^{B_0} = B_0\overline{H}\,\bar{t}_7 + t_7(\overline{MO}_{18} + MO_{17})$$

$$1^{B_1} = B_0\overline{B}_1\overline{H}\,\bar{t}_7 + t_7(\overline{MO}_{18} + MO_{17}) + H$$

$$0^{B_1} = B_0B_1\overline{H}\,\bar{t}_7 + t_7(MO_{18}\overline{MO}_{17}\overline{MO}_{16}MO_{15} + MO_{18}\overline{MO}_{17}MO_{16}\overline{MO}_{15})$$

$$1^{B_2} = B_0B_1\overline{B}_2\overline{H}\,\bar{t}_7 + t_7(\overline{MO}_{18} + MO_{17} + MO_{18}\overline{MO}_{17}\overline{MO}_{16}MO_{15}) + H$$

$$0^{B_2} = B_0B_1B_2\overline{H}\,\bar{t}_7 + t_7(MO_{18}\overline{MO}_{17}MO_{16}\overline{MO}_{15})$$

This logic has not been reduced to its minimum form. In decoding it is apparent that, since count zero never occurs, it may be used as an excluded count. All other counts are decoded. Accordingly

$$t_1 = \overline{B}_1\overline{B}_2 \qquad t_4 = \overline{B}_0\overline{B}_1$$

$$t_2 = \overline{B}_0\overline{B}_2 \qquad t_5 = B_0\overline{B}_1B_2$$

$$t_3 = B_0B_1\overline{B}_2 \qquad t_6 = \overline{B}_0B_1B_2$$

$$t_7 = B_0B_1B_2$$

4.5.3.2 Instruction Counter

The instruction counter is used to address the next instruction. Normally this counter advances in straight binary, one count as a result of the occurrence of t_7. However in the instruction, skip if the A register is negative (SAN), it is required to advance two counts. In addition, on a branch instruction, the instruction counter is caused to copy the current memory address of the instruction register. The instruction counter contains 12 flip-flops (IC_0 through IC_{11}). The least significant bit of this counter is IC_0. The set/reset logic for this counter is

$$1^{IC_0} = \overline{IC}_0[t_6(SAN)A_{19} + \overline{BRU}\,t_7\overline{H}] + IR_0 BRU\,t_7$$

$$0^{IC_0} = IC_0[t_6(SAN)A_{19} + \overline{BRU}\,t_7\overline{H}] + \overline{IR}_0 BRU\,t_7$$

$$1^{IC_1} = IC_0\overline{IC}_1[t_6(SAN)A_{19} + \overline{BRU}\,t_7\overline{H}] + IR_1 BRU\,t_7$$

$$0^{IC_1} = IC_0 IC_1[t_6(SAN)A_{19} + \overline{BRU}\,t_7\overline{H}] + \overline{IR}_1 BRU\,t_7$$

$$\vdots$$

$$1^{IC_{11}} = (IC_0 IC_1 \ldots \overline{IC}_{11})[t_6(SAN)A_{19} + \overline{BRU}\,t_7\overline{H}] + IR_{11} BRU\,t_7$$

$$0^{IC_{11}} = (IC_0 IC_1 \ldots IC_{11})[t_6(SAN)A_{19} + \overline{BRU}\,t_7\overline{H}] + \overline{IR}_{11} BRU\,t_7$$

4.5.3.3 Index Register and Index Adder

The index register and adder provide a method for modifying the address specified by an instruction prior to accessing memory. This feature facilitates cycling through the same set of instructions to evaluate successive logic units.

The contents of the A register are copied into the index register when the instruction, transfer A to index register, occurs (ATI). The bits of the index register are labeled IND_0 through IND_{11}, where IND_0 is the least significant position. The logic for copying is straight forward but is written down for completeness.

$$1^{IND_0} = A_0(ATI)t_7 \qquad 0^{IND_0} = \overline{A}_0(ATI)t_7$$

$$1^{IND_1} = A_1(ATI)t_7 \qquad 0^{IND_0} = \overline{A}_1(ATI)t_7$$

$$\vdots \qquad\qquad\qquad \vdots$$

$$1^{IND_{11}} = A_{11}(ATI)t_7 \qquad 0^{IND_{11}} = \overline{A}_{11}(ATI)t_7$$

A parallel adder is used to add the index register to the address contained by the instruction register (IR_0 through IR_{11}) Addition is complete in less than 1.2μsec and the modified address is available for use in addressing the memory on the first read or write pulse occurring after t_7. The outputs of the adder are designated IN_0 through IN_{11}. The carries, internal to the adder, are designated CI_1 through CI_{11}. The conventional adder logic is,

$$IN_0 = (IR_0)(\overline{IND}_0) + (\overline{IR}_0 IND_0)$$

$$CI_1 = IR_0 IND_0$$

$$IN_1 = IR_1 IND_1 CI_1 + IR_1 \overline{IND}_1 \overline{CI}_1 + \overline{IR}_1 \overline{IND}_1 CI_1 + \overline{IR}_1 IND_1 \overline{CI}_1$$

$$CI_2 = IR_1 IND_1 + IR_1 CI_1 + IND_1 CI_1$$

$$\vdots$$

$$CI_{11} = IR_{10} IND_{10} + IR_{10} CI_{10} + IND_{10} CI_{10}$$

$$IN_{11} = IR_{11} IND_{11} CI_{11} + IR_{11} \overline{IND}_{11} \overline{CI}_{11} + \overline{IR}_{11} \overline{IND}_{11} CI_{11} + \overline{IR}_{11} IND_{11} \overline{CI}_{11}$$

4.5.3.4 Instruction Register and Decoder

The instruction register is used to retain the instruction to be executed, since normally in executing an instruction the memory output changes from an instruction word to a data word, erasing the instruction to be operated on. The instruction from memory is simply copied at the middle of t_7 into the instruction register by using the out-of-phase clock. All 13 instructions are decoded. The instruction register flip-flops are labeled IR_0 through IR_{11} and IR_{15} through IR_{19}. The copy logic for this register is simply

$$
\begin{aligned}
{}_{1}{}^{IR_0} &= MO_0 t_7 & {}_{0}{}^{IR_0} &= \overline{MO}_0 t_7 \\
&\vdots & &\vdots \\
{}_{1}{}^{IR_{11}} &= MO_{11} t_7 & {}_{0}{}^{IR_{11}} &= \overline{MO}_{11} t_7 \\
{}_{1}{}^{IR_{15}} &= MO_{15} t_7 & {}_{0}{}^{IR_{15}} &= \overline{MO}_{15} t_7 \\
&\vdots & &\vdots \\
{}_{1}{}^{IR_{19}} &= MO_{19} t_7 & {}_{0}{}^{IR_{19}} &= \overline{MO}_{19} t_7
\end{aligned}
$$

Since the excluded counts are not used to reduce the instruction decoding logic, three additional instructions may be added to expand the computing capability with no change in the decoding logic.

$$
\begin{aligned}
LDA &= \overline{IR}_{15}\overline{IR}_{16}\overline{IR}_{17}\overline{IR}_{18} & LSS &= IR_{15}IR_{16}IR_{17}\overline{IR}_{18} \\
STA &= IR_{15}\overline{IR}_{16}\overline{IR}_{17}\overline{IR}_{18} & ADD &= IR_{15}\overline{IR}_{16}\overline{IR}_{17}IR_{18} \\
SAN &= \overline{IR}_{15}IR_{16}\overline{IR}_{17}\overline{IR}_{18} & MULT &= \overline{IR}_{15}IR_{16}\overline{IR}_{17}IR_{18} \\
BRU &= IR_{15}IR_{16}\overline{IR}_{17}\overline{IR}_{18} & WAIT &= IR_{15}IR_{16}\overline{IR}_{17}IR_{18} \\
ATI &= \overline{IR}_{15}\overline{IR}_{16}IR_{17}\overline{IR}_{18} & STAD &= \overline{IR}_{15}\overline{IR}_{16}IR_{17}IR_{18} \\
ITA &= IR_{15}\overline{IR}_{16}IR_{17}\overline{IR}_{18} & WRITE &= IR_{15}\overline{IR}_{16}IR_{17}IR_{18} \\
LDDA &= \overline{IR}_{15}IR_{16}IR_{17}\overline{IR}_{18}
\end{aligned}
$$

4.5.3.5 Word Format Register and Decoder

The word format register copies the memory bits MO_{12} and MO_{14} at the end of t_7. The copy logic is

$$1^{WF_0} = MO_{12}t_7 \qquad 0^{WF_0} = \overline{MO}_{12}t_7$$

$$1^{WF_1} = MO_{13}t_7 \qquad 0^{WF_1} = \overline{MO}_{13}t_7$$

$$1^{WF_2} = MO_{14}t_7 \qquad 0^{WF_1} = \overline{MO}_{14}t_7$$

Only six of the possible eight word formats are decoded. This is true since lines going to the D/A's (X, Y and subsection) are hardwired to the appropriate bit positions, hence the required portion of the word from memory is automatically selected by the instruction. The format decoding logic is,

$$F_1 = \overline{WF}_0\overline{WF}_1\overline{WF}_2 \qquad F_4 = \overline{WF}_0\overline{WF}_1WF_2$$

$$F_2 = \overline{WF}_0WF_1\overline{WF}_2 \qquad F_5 = WF_0\overline{WF}_1WF_2$$

$$F_3 = WF_0WF_1\overline{WF}_2 \qquad F_6 = \overline{WF}_0WF_1WF_2$$

4.5.4 Arithmetic Unit

The arithmetic unit has provisions for multiplication and addition. All arithmetic operations are performed in fixed point. Negative numbers are represented in two's complement form. The arithmetic unit consists of three registers labeled M, A, and Z, logic circuitry to perform addition and complementing the M, A and Z register set/reset logic for loading, transferring, shifting and retaining the resultant sums, partial products and products.

Since arithmetic operations are performed in fixed point, the numbers held by the A or M register are of limited size. To insure that these registers are of adequate size, consider the largest value which can be obtained in generating a quadratic or linear response sum. For the quadratic unit, the quadratic weights C_{ij} are five bits plus sign. This choice is based on error considerations. The linear weights b_i are selected as nine bits plus sign; however, only the most significant five bits after rounding plus sign are used in forming the b_iZ_i products. The value of a_o may be as large as 13 bits plus sign; however, only the most significant five bits, after

rounding, plus sign will be used. Notice that since c_{ij}'s, b_i's and a_o are all stored in memory as five bits plus sign loading of the M and A registers varies. Correct loading is accomplished by means of the word format. Returning to the size of the M and A register, it is clear that the largest possible sum of all the products of a quadratic unit, including the sign bit, is

$$\pm [(21)(2^5)(2^4)(2^4) + 6(2^9)(2^4) + 2^{13}] = \pm (28)2^{13} = \pm 1.75 \times 2^{17}$$

To accommodate this size number the M and A register are made equal to 19 bits plus sign. Using 20-bit M and A registers is overly conservative, since the weights are never simultaneously a maximum nor all of the same sign. In experiments with two quadratic logic units, it has been found that a particular Z vector (set of six Z_i's), the sum of all terms yields numbers ranging from -12,426 to 9,684. This corresponds to 14 bits plus sign, however the order in which the sum is generated could produce even larger interim values.

The response unit weights are chosen as 11 bits plus sign. Assuming half of the quadratic unit output's (128) supply inputs to a response unit, the largest possible response sum is $\pm 2^7 \cdot 2^{11} = \pm 2^{18}$. The largest numbers which can be accommodated by the M or A register is $+2^{19} - 1$ and -2^{19} so the register capacity is more than adequate.

4.5.4.1 Two's Complement Arithmetic

Negative numbers are held in memory in binary two's complement form. The two's complement of a number is formed by adding one to the one's complement (logical inverse) of the number. This convention allows the sign of a number to be used as an integral part of the number and obviates the need for keeping track of a detached sign with computer logic. Another advantage of two's complement arithmetic is that only one representation of zero exists. The largest possible number which can be represented with the M and A register, equal to 20 bit positions, is $+(2^{19} - 1)$ and -2^{19}, as previously stated.

To add positive or negative numbers, with negative number in 2's complement form, it is only necessary to add the A and M register outputs, bit by bit, including the sign bit and to neglect any

carry occurring from the most significant bit position (sign bit) if a carry occurs. A true overflow in addition can only result if the signs of the sign bits (M_{19} and A_{19}) are alike, but the indicated sum, S_{19}, is of opposite sign. In the event the signs M_{19} and A_{19} are opposite, it is impossible for a true overflow to occur. Overflow in the arithmetic unit is not checked since the size of the M and A register prevent overflow from occurring for the problem being served.

To perform multiplication, the contents of the A register are transfered to the M register and the A register is zero set, while the multiplier (Z data point) is entered into the Z register. Multiplication is always carried out in five bit times, excluding the time to access the multiplier and multiplicand and fetch the instruction.

Multiplication is done by examining the pairs of the multiplier digits, that is, Z_1Z_0, Z_2Z_1, Z_3Z_2, Z_4Z_3, Z_5Z_4 to determine the action to be taken with regard to the multiplicand in the M register. The process may be reduced to three simple rules.

1. If a multiplier digit is the same as the next lower-order multiplier digit, add nothing and shift the M register left.
2. If a multiplier digit is false and the next lower order is true, add the contents of M to A and then shift M left.
3. If a multiplier digit is one and the next lower order is false, add the two's complement of M to A and then shift M left.

The sign digit of the Z register is operated upon exactly as though it were the highest order digit of the number.

The process yields the correct product, independent of the sign of M or Z. In the system the Z sign bit will always be positive but the M sign bit may be either positive or negative. In shifting the M register the least significant bit M_o is zero set as a result of the multiply instruction and bit time t_2, to insure that zeros propagate to the left from the least significant bit position.

The following two examples are shown in Figure 78 to show how the multiplication algorithm works. Notice in comparing Z_1Z_0 that it is essential to force $Z_0 = 0$ to make the first decision as

A = 0.0000000000000000000
M = 0.0000000000000001101
Z = 0.1011 ---

(+11)(+13)

Bit Time	Action Taken	A Register Contents
t_2	Add 2's Complement of M to A & Shift M	$(1)(2^{20}-13)$
t_3	Shift M	$(1)(2^{20}-13)$
t_4	Add M to A & Shift M	$(4)(13)+1(20^{20}-13)$
t_5	Add 2's Complement of M to A & Shift M	$8(20^{20}-13)+4(13)+1(2^{20}-13)$
t_6	Add M to A	$(16)(13)+8(2^{20}-13)+4(13)+1(2^{20}-13) \rightarrow \underline{11x13}$

A = 0.0000000000000000000
M = 1.1111111111111110010
Z = 0.1011 ---

(+11)(-13)

Bit Time	Action	A Register Contents
t_2	Add 2's Complement of M to A & Shift M	$(1)(13)$
t_3	Shift M	$(1)(13)$
t_4	Add M to A & Shift M	$4(2^{20}-13)+1(13)$
t_5	Add 2's Complement of M to A & Shift M	$8(13)+4(2^{20}-13)+1(13)$
t_6	Add M to A	$16(2^{20}-13)+8(13)+4(2^{20}-13)+1(13) \rightarrow \underline{2^{20}-11 \cdot 13}$

Figure 78. Multiplication

c3419

to the action to be taken. To compare successive bit pairs in the proper sequence the Z register is shifted right. The M register is shifted left.

This section is included to clarify the method used for multiplication since several techniques exist.

4.5.4.2 Adder and True/Complement Logic and Complement Control Logic

The two's complement of the contents of the M register may effectively be added to the contents of the A register by inserting a 1 in the adder's zero position carry, C_0, and adding the one's complement of the contents of the M register (logical inverse of M). Preceding any addition or multiplication it must be determined whether the normal true or logical inverse sides of the M register are to act as inputs to the adder. When adding, the true sides of the M register are always required. In multiplication either true or the false side may be used or all modified M lines (M') eminating from the true/complement logic are forced false (adding zero). The complement control logic is

$$NC_0 = Add + \overline{Z}_1 Z_0 \text{ (MULT)}$$

$$C_0 = Z_1 \overline{Z}_0 \text{ (MULT)}$$

where NC_0 provides the true M lines to the adder. C_0 provides the one's complement of the M lines to the adder and forces a zero carry to the least significant bit of the adder.

The true/complement logic is tabulated below. The least significant output from the M register is M_0 whereas the resultant output to the adder is M'_0.

$$M'_0 = M_0(NC_0) + \overline{M}_0 C_0$$

$$M'_1 = M_1(NC_0) + \overline{M}_1 C_0$$

$$\vdots$$

$$M'_{19} = M_{19}(NC_0) + \overline{M}_{19} C_0$$

The full adder logic as well known is

$$C_0 = C_0$$

$$S_0 = A_0M_0'C_0 + \overline{A}_0\overline{M}_0'C_0 + \overline{A}_0M_0'\overline{C}_0 + A_0\overline{M}_0'\overline{C}_0$$

$$C_1 = A_0M_0' + A_0C_0 + M_0'C_0$$

$$S_1 = A_1M_1'C_1 + \overline{A}_1\overline{M}_1'C_1 + \overline{A}_1M_1'\overline{C}_1 + A_1\overline{M}_1'\overline{C}_1$$

$$C_2 = A_1M_1' + A_1C_1 + M_1'C_1$$

$$\vdots$$

$$S_{18} = A_{18}M_{18}'C_{18} + \overline{A}_{18}\overline{M}_{18}'C_{18} + \overline{A}_{18}M_{18}'\overline{C}_{18} + A_{18}\overline{M}_{18}'\overline{C}_{18}$$

$$C_{19} = A_{18}M_{18}' + A_{18}C_{18} + M_{18}'C_{18}$$

$$S_{19} = A_{19}M_{19}'C_{19} + \overline{A}_{19}\overline{M}_{19}'C_{19} + \overline{A}_{19}M_{19}'\overline{C}_{19} + A_{19}\overline{M}_{19}'\overline{C}_{19}$$

4.5.4.3 Z Register Logic

The Z register contains five flip-flops, Z_4 through Z_0. This register is used exclusively for multiplying. Z_4 through Z_1 are loaded during a multiply instruction from the memory output line MO_0 through MO_3. As a result of t_1 occurring, the Z_0 flip-flop is cleared to zero. In addition the register has the capability of shifting right during multiplication. The set/reset logic for this register is

$$1^{Z_0} = \text{MULT}(Z_1\bar{t}_1) \qquad 0^{Z_0} = (\text{MULT})(\overline{Z}_1\bar{t}_1 + t_1)$$

$$1^{Z_1} = \text{MULT}(Z_2\bar{t}_1 + MO_0t_1) \qquad 0^{Z_1} = \text{MULT}(\overline{Z}_2\bar{t}_1 + \overline{MO}_0t_1)$$

$$1^{Z_2} = \text{MULT}(Z_3\bar{t}_1 + MO_1t_1) \qquad 0^{Z_2} = \text{MULT}(\overline{Z}_3\bar{t}_1 + \overline{MO}_1t_1)$$

$$1^{Z_3} = \text{MULT}(Z_4\bar{t}_1) + MO_2t_1) \qquad 0^{Z_3} = \text{MULT}(\overline{Z}_4\bar{t}_1 + \overline{MO}_2t_1)$$

$$1^{Z_4} = \text{MULT}(Z_5\bar{t}_1 + MO_3t_1) \qquad 0^{Z_4} = \text{MULT}(\overline{Z}_5\bar{t}_1 + \overline{MO}_3t_1)$$

$$Z_5 = \text{TRUE SUPPLY VOLTAGE} \qquad \overline{Z}_5 = \text{FALSE SUPPLY VOLTAGE}$$

4.5.4.4 A Register Logic

The A register (A_0 through A_{19}) copies the sum lines (S_0 through S_{19}) as a result of an add instruction and t_6 occurring or for the multiply instruction except during t_1 or t_7. In a multiply instruction as a result of t_1 occurring, the A register is cleared. The A register is also loaded from memory as a result of the instruction, load A (LDA), and the occurrence of t_6. The manner in which the memory is loaded depends upon the memory word format. If the memory word format is F_1 or F_2 the data being loaded contains a quadratic weight C_{ij}. If the weight is negative its sign bit must be spread from A_5 through A_{19}. For a full word format, F_6, the memory output (MO_0 through MO_{19}) is simply copied directly into A. In addition the A register is also loaded from the index register (IND_0 through IND_{11}) as a result of the instruction ITA and t_6. ITA and t_6 also cause A_{12} through A_{19} to be zero set.

To yield the previously described results the A register logic is shown on the following page:

$$P = \text{MULT}\, t_1 \qquad Q = \text{LDA}\, t_6 F_6 \qquad R = \text{ADD}\, t_6 + \text{MULT}\, \bar{t}_1 \bar{t}_7$$

$$T = (\text{LDA})\, t_6 (MO_5 F_2 + MO_{11} F_1) \qquad U = (\text{LDA})\, t_6 (\overline{MO}_5 F_2 + \overline{MO}_{11} F_1)$$

$$V = (\text{LDA}\, t_6 \qquad W = (\text{ITA})\, t_6 \qquad X = (\text{LDA})\, t_6 (F_2 + F_6)$$

$$Y = \text{LDA}\, t_6 F_1$$

$$1^{A_0} = S_0 R + MO_0 X + MO_6 Y + IND_0 W$$

$$0^{A_0} = \bar{S}_0 R + \overline{MO}_0 X + \overline{MO}_6 Y + \overline{IND}_0 W + P$$

$$1^{A_5} = S_5 R + MO_5 X + MO_{11} Y + IND_5 W$$

$$0^{A_5} = \bar{S}_5 R + \overline{MO}_5 X + \overline{MO}_{11} Y + \overline{IND}_5 W + P$$

$$1^{A_6} = S_6 R + MO_6 Q + T + IND_6 W$$

$$0^{A_6} = S_6 R + \overline{MO}_6 Q + U + IND_6 W + P$$

$$1^{A_{11}} = S_{11} R + MO_{11} Q + T + IND_{11} W$$

$$0^{A_{11}} = \bar{S}_{11} R + \overline{MO}_{11} Q + U + \overline{IND}_{11} W + P$$

$$1^{A_{12}} = S_{12} R + MO_{12} Q + T$$

$$0^{A_{12}} = \bar{S}_{12} R + \overline{MO}_{12} Q + U + W + P$$

$$1^{A_{19}} = S_{19} R + MO_{19} Q + T$$

$$0^{A_{19}} = \bar{S}_{19} R + \overline{MO}_{19} Q + U + W + P$$

4.5.4.5 M Register Logic

The A register is transferred to the M register as a result of a multiply instruction and t_1 occurring. The M register is required to shift left during a multiply instruction except during t_1 or t_7. With the multiply instruction and as a result of t_2 occurring, M_0 is zero set; this insures that zeros are propagated to the left from the least significant bit position during a multiply. In addition the M register copies the memory output as a result of the add instruction and t_5 occurring. The manner in which the data is copied depends upon the word format. Data flows into the M register on formats F_3, F_4, F_5 and F_6. The linear weights of the quadratic unit, threshold of the quadratic unit and weight of the response unit are formats F_3, F_4, and F_5, respectively. The magnitude of the linear quadratic weight goes into positions M_4 through M_8 while M_9 through M_{19} copy the sign of the weight and M_0 through M_4 are zero set. The quadratic unit threshold goes into positions M_8 through M_{12} while M_{13} through M_{19} copy the sign of the threshold and M_0 through M_9 are zero set. The response weight is entered into M_0 through M_{11} with M_{11} through M_{19} copying the sign bit of this weight. Full word formats, F_6, are transferred on a bit by bit basis from the memory output to the M register.

The M register logic is shown on the following page:

$\alpha = MO_{17}F_3D \qquad \beta = \overline{MO}_{17}F_3D \qquad \gamma = MO_{17}F_4D \qquad \delta = \overline{MO}_{17}F_4D \qquad \mu = MO_{17}F_5D \qquad \nu = \overline{MO}_{17}F_5D$

$D = (ADD)t_5 \qquad E = (MULT)t_1 \qquad K = (MULT)\bar{t}_1\bar{t}_7$

$$
\begin{aligned}
{}_1M_0 &= A_0E + MO_0F_6D + MO_6F_5D \\
{}_1M_1 &= A_1E + M_0K + MO_1F_6D + MO_7F_5D \\
{}_1M_2 &= A_2E + M_1K + MO_2F_6D + MO_8F_5D \\
{}_1M_3 &= A_3E + M_2K + MO_3F_6D + MO_9F_5D \\
{}_1M_4 &= A_4E + M_3K + MO_4F_6D + MO_{12}F_3D + MO_{10}F_5D \\
{}_1M_5 &= A_5E + M_4K + MO_5F_6D + MO_{13}F_3D + MO_{11}F_5D \\
{}_1M_6 &= A_6E + M_5K + MO_6F_6D + MO_{14}F_3D + MO_{12}F_5D \\
{}_1M_7 &= A_7E + M_6K + MO_7F_6D + MO_{15}F_3D + MO_{13}F_5D \\
{}_1M_8 &= A_8E + M_7K + MO_8F_6D + MO_{16}F_3D + MO_{12}F_4D + MO_{14}F_5D \\
{}_1M_9 &= A_9E + M_8K + MO_9F_6D + \alpha + MO_{13}F_4D + MO_{15}F_5D \\
{}_1M_{10} &= A_{10}E + M_9K + MO_{10}F_6D + \alpha + MO_{14}F_4D + MO_{16}F_5D \\
{}_1M_{11} &= A_{11}E + M_{10}K + MO_{11}F_6D + \alpha + MO_{15}F_4D + \mu \\
{}_1M_{12} &= A_{12}E + M_{11}K + MO_{12}F_6D + \alpha + MO_{16}F_4D + \mu \\
{}_1M_{13} &= A_{13}E + M_{12}K + MO_{13}F_6 + \alpha + \gamma + \mu \\
&\vdots \\
{}_1M_{19} &= A_{19}E + M_{18}K + MO_{19}F_6D + \alpha + \gamma + \mu
\end{aligned}
$$

$$
\begin{aligned}
{}_0M_0 &= \overline{A}_0E + (MULT)t_2 + \overline{MO}_0F_6D + F_3D + F_4D + \overline{MO}_6F_5D \\
{}_0M_1 &= \overline{A}_1E + \overline{M}_0K + \overline{MO}_1F_6D + F_3D + F_4D + \overline{MO}_7F_5D \\
{}_0M_2 &= \overline{A}_2E + \overline{M}_1K + \overline{MO}_2F_6D + F_3D + F_4D + \overline{MO}_8F_5D \\
{}_0M_3 &= \overline{A}_3E + \overline{M}_2K + \overline{MO}_3F_6D + F_3D + F_4D + \overline{MO}_9F_5D \\
{}_0M_4 &= \overline{A}_4E + \overline{M}_3K + \overline{MO}_4F_6D + \overline{MO}_{12}F_3D + F_4D + \overline{MO}_{10}F_5D \\
{}_0M_5 &= \overline{A}_5E + \overline{M}_4K + \overline{MO}_5F_6D + \overline{MO}_{13}F_3D + F_4D + \overline{MO}_{11}F_5D \\
{}_0M_6 &= \overline{A}_6E + \overline{M}_5K + \overline{MO}_6F_6D + \overline{MO}_{14}F_3D + F_4D + \overline{MO}_{12}F_5D \\
{}_0M_7 &= \overline{A}_7E + \overline{M}_6K + \overline{MO}_7F_6D + \overline{MO}_{15}F_3D + F_4D + \overline{MO}_{13}F_5D \\
{}_0M_8 &= \overline{A}_8E + \overline{M}_7K + \overline{MO}_8F_6D + \overline{MO}_{16}F_3D + \overline{MO}_{12}F_4D + \overline{MO}_{14}F_5D \\
{}_0M_9 &= \overline{A}_9E + \overline{M}_8K + \overline{MO}_9F_6D + \beta + \overline{MO}_{13}F_4D + \overline{MO}_{15}F_5D \\
{}_0M_{10} &= \overline{A}_{10}E + \overline{M}_9K + \overline{MO}_{10}F_6D + \beta + \overline{MO}_{14}F_4D + \overline{MO}_{16}F_5D \\
{}_0M_{11} &= \overline{A}_{11}E + \overline{M}_{10}K + \overline{MO}_{11}F_6D + \beta + \overline{MO}_{15}F_4D + \nu \\
{}_0M_{12} &= \overline{A}_{12}E + \overline{M}_{11}K + \overline{MO}_{12}F_6D + \beta + \overline{MO}_{16}F_4D + \nu \\
{}_0M_{13} &= A_{13}E + M_{12}K + MO_{13}F_6D + \beta + \delta + \nu \\
&\vdots \\
{}_0M_{19} &= A_{19}E + M_{18}K + MO_{19}F_6D + \beta + \delta + \nu
\end{aligned}
$$

4.5.5 Power Supplies

The power supplies will convert the 28 v D.C. source of power, normally on-board a satellite, to multiple D.C. levels. The power inverter and voltage regulators will supply ±15, ±5 and -2400 volts to the system. The inverter and regulators are estimated to be 70% efficient and hence dissipate about 8.8 watts since the power requirements for all other subsystems is estimated to be 20.6 watts. The required supply should weigh less than 2.5 pounds and occupy less than 275 cubic inches.

4.5.6 Summary of Sequential/Digital Hardware

Item	Estimated		
	Volume (cubic inches)	Weight (pounds)	Power (watts)
Input Unit	180	7.2	5.2
Memory	195	8.7	8.5
Control Unit	350	3.2	3.0
Arithmetic Unit	450	4.1	4.0
Power Supplies	275	2.5	8.8
Total	1450	25.7	29.5

4.5.7 Program

The system program is described by the flow chart of Figure 79. Note that the program provides for data to be accessed during the interval when computing a quadratic logic unit is taking place. The complete program using the available 13 instructions has been written.

4.6 Recommendations

As a result of the hardware feasibility study it has been shown that a recognition system on board a satellite is feasible, providing either the sequential/hybrid or sequential/digital implementation is employed.

The power consumption of both sequential implementations is less than that used by the APT system (40.1 watts). The weight of the sequential systems is comparable with the weight of the APT system (25.5 pounds).

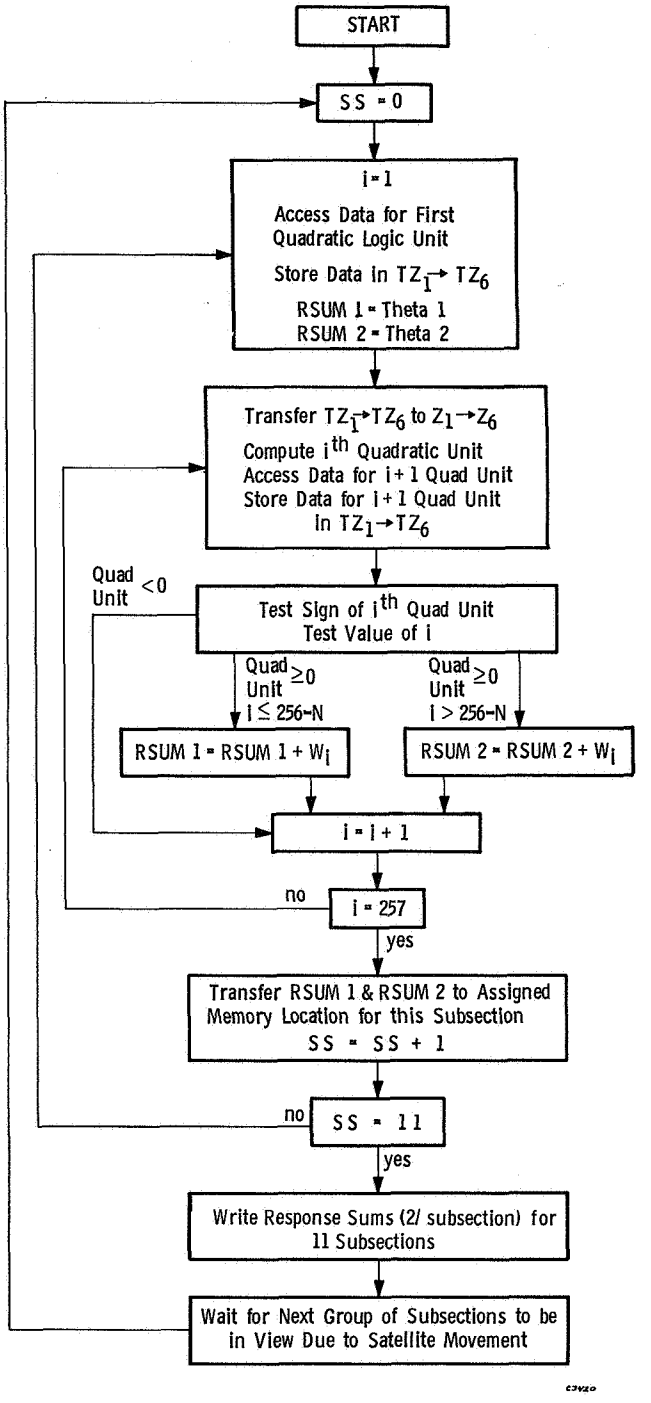


Figure 79. Flow Diagram of Stored Program Using Sequential/ Digital Decision Network Implementation

None of the systems studied use moving parts and for this reason should be extremely reliable.

One major reason for practicality of the sequential implementations stems from the recent developments in digital integrated circuits. Since the preliminary design has been considered using standard printed circuit boards and connectors, it is possible to reduce the estimated system weight and size by employing multilayer printed circuit boards.

The sequential/hybrid system size and weight could be improved significantly by using a 38-bit core stack with 2048 words. With this approach data accessing is done in coincidence with loading quadratic weights and thresholds. The sequential/hybrid approach provides a simpler control unit than the sequential/digital system. Computing circuitry could be significantly improved if the quadratic property filters could be replaced by parallel hyperplane property filters. The general simplicity of the sequential/hybrid system renders it extremely appealing where only statistical property filters are to be implemented.

Even though other memory techniques may be less expensive, the core memory offers a particularly strong advantage in that the weights, thresholds, data coordinates and stored program may be read into the memory directly from Astropower's SDS 930 computer. The original system design and subsequent design improvements may be implemented within seconds. This is in sharp contrast to the diode matrix which requires time consuming wiring and construction for each design implementation.

The sequential/digital approach makes use of tried and time tested technology. Further, it provides a degree of flexibility that other systems lack. This may be used to advantage to implement both known and statistical property filters with a minimal amount of additional hardware (i.e., memory expansion).

5.0 NEW TECHNOLOGY

(1) Section 3.3.7 of this report discusses a procedure which permits the utilization of the two techniques for generating property filters, known property extraction and statistical property extraction, to be used in concert in designing a recognition system. The results reported in that section demonstrate that the augmentation technique is capable of increasing the performance level of these systems, particularly in those cases where the known properties alone do not achieve a high level of correct classification. The results presented are the first, to the authors' knowledge, demonstrating this capability. As such they are reported, under separate cover, in compliance with NASA's requirements.

(2) The hardware discussions of Section 4.0 present several alternate approaches to the design of a recognition system for the on-board processing of video data. Though each design incorporates well known and documented design techniques, the mechanization into a system capable of performing the classification of cloud pattern data and compatible with satellite system requirements has not been reported in the past.

In each of the above cases documentation is being supplied to the New Technology Representative at Douglas for further action.

REFERENCES

1. Abramson, N., Learning in Pattern Recognition, Tech. Note No. 1 (ASTIA No. AD 267 798), International Telephone & Telegraph Corp., ITT Federal Laboratories, Palo Alto, Calif., Oct. 1961, 63p.

2. Abramson, N. M., Dickinson, W. E., and Wood, F. B., The Application of Decision Theory to Voice Recognition Machines, with an Illustrated Example (a revision of IBM Research Report RJ 158 of March 5, 1959), IBM Advanced Systems Development Division, San Jose, Calif., Dec. 15, 1959, 32p.

3. Adaptive Techniques As Applied to Textual Data Retrieval, Vols. I & II, Tech. Documentary Report No. RADC-TDR-64-206 (ASTIA No. AD 605 260), Douglas Aircraft Co., Astropower Laboratory, Newport Beach, Calif., May 1964, 219p.

4. Alt, F. L., "Digital Pattern Recognition by Moments," J. Assn. for Computing Machinery, 9:2, April 1962, pp. 240-258.

5. Anderson, T. W. and Bahadur, R. R., "Classification into Two Multivariate Normal Distributions with Different Covariance Matrices," Annals of Math. Stat., 33:2, June 1962, pp. 420-431.

6. Andrew, A. M., "The Conditional Probability Computer," in Mechanisation of Thought Processes, Vol. II (Proc. of the National Physical Laboratory Symposium No. 10), Her Majesty's Stationery Office, London, 1959, pp. 945-946.

7. Anonymous, "Research Highlights: Machine Learning and Pattern Recognition," Automatic Control, August 1960, pp. 16-17.

8. Babcock, T. R. and Richmond, G. E., Application of Perceptrons to Photointerpretation, Report No. VE-1446-G-4, (ASTIA No. AD 605 402), Cornell Aeronautical Laboratory, Inc., Buffalo, N. Y., July 1964, 75p.

9. Bailey, C. E. G., "Introductory Lecture on Character Recognition" (Specialist Discussion Meetings on New Digital-Computer Techniques-Session 1, Feb. 16-17, 1959), Proc. IEE, Part B, Sept. 1959, pp. 444-445.

10. Barus, C., "A Scheme for Recognizing Patterns from an Unspecified Class," in Optical Character Recognition (G. L. Fischer, Jr., D. K. Pollock, B. Radack, and M. E. Stevens, eds.), Spartan Books, Washington, D. C., 1962, pp. 227-247.

11. Barus, C., Pattern Learning by Subset Selection: An Improved Method, Final Progress Report on NSF Grant C-5945, Dept. of Electrical Engineering, Swarthmore College, Dec. 28, 1963, 17p.

12. Barus, C., Pattern Recognition by Statistical Overlap, Dept. of Electrical Engineering, Swarthmore College, August 11, 1960, 5p.

13. Bauldreay, J. and Milbradt, E., "Solving Registration Problems in Optical Character Recognition," Electronics, 35:1, Jan. 5, 1962, pp. 77-81.

14. Bellman, R., Dynamic Programming, Intelligent Machines, and Self-Organizing Systems, Memo. No. RM-3173-PR (ASTIA No. AD 276 533), The RAND Corp., Santa Monica, Calif., June 1962, 15p.

15. Bledsoe, W. W., "Further Results on the N-tuple Pattern Recognition Method," IRE Trans. on Electronic Computers, EC-10:1, March 1961, p. 96.

16. Bledsoe, W. W. and Browning, I., "Pattern Recognition and Reading by Machine," 1959 Proc. Eastern Joint Computer Conference, 9p.

17. Bledsoe, W. W., Bomba, J. S., Browning, I., Evey, R. J., Kirsch, R. A., Mattson, R. L., Minsky, M., Neisser, U., and Selfridge, O. G., "Discussion of Problems in Pattern Recognition," 1959 Proc. Eastern Joint Computer Conference, pp. 233-237.

18. Bomba, J. S., "Alpha-Numeric Character Recognition Using Local Operations," 1959 Proc. Eastern Joint Computer Conference, pp. 218-224.

19. Brain, A. E. and Forsen, G. E., Graphical Data Processing Research Study and Experimental Investigation, Quarterly Progress Report 7 (ASTIA No. AD 275 836), Stanford Research Institute, Menlo Park, Calif., Dec. 1 to Feb. 28, 1962, 34p.

20. Brain, A. E., Duda, R. O., Hall, D. J., and Munson, J. H., Graphical Data Processing Research Study and Experimental Investigation, Quarterly Progress Report 5 (ASTIA No. AD 453 094), Stanford Research Institute, Menlo Park, Calif., July 1 to Sept. 30, 1964, 52p.

21. Braverman, E. M., "Experiments in Training a Machine to Distinguish Visual Shapes," Avtomatika i Telemekhanika, XXIII:3, Moscow, 1962, pp. 349-364 (JPRS No. 13694, May 9, 1962).

22. Bryan, J. S., "Experiments in Adaptive Pattern Recognition," IEEE Trans. on Military Electronics, MIL-7:2 & 3, April-July 1963, pp. 174-179.

23. Calderwood, J. H. and Porter, A., "Pattern Recognition in the Synthesis of Complex Switching Systems," J. Electronics and Control, 4, May 1958, pp. 466-480.

24. Caldwell, W. F., "Recognition of Sounds by Cochlear Patterns," IEEE Trans. on Military Electronics, MIL-7:2 & 3, April-July 1963, pp. 179-185.

25. Cameron, S. H., Self Organizing Networks, Annual Report for Feb. 15, 1961 - Feb. 14, 1962, Armour Research Foundation, Illinois Institute of Technology, Chicago, Ill., Feb. 14, 1962, 15p.

26. Carne, E. B., A Study of Generalized Machine Learning, Tech. Documentary Report ASID-TDR-62-166 (ASTIA No. AD 277 493), Melpar, Inc., Falls Church, Va., April 1962, 238p.

27. Carne, E. B., Connelly, E. M., Halpern, P. H., and Logan, B. A., "A Self-Organizing Binary Logical Element," in Biological Prototypes and Synthetic Systems, Vol. I (E. E. Bernard and M. R. Kare, eds.), Plenum Press, New York, 1962, pp. 311-330.

28. Carroll, K. D., Character Recognition Devices for Electronic Computers: An Annotated Bibliography, Special Research Bibliography No. SRB-60-11, Lockheed Aircraft Corp., Missiles and Space Division, Sunnyvale, Calif., Nov. 1960, 96p.

29. Chatten, J. B. and Teacher, C. F., "Character Recognition Techniques for Address Reading," in Optical Character Recognition (G. L. Fischer, Jr., D. K. Pollock, B. Radack, and M. E. Stevens, eds.), Spartan Books, Washington, D. C., 1962, pp. 51-59.

30. Cheatham, T. P., Jr. and Kohlenberg, A., "Optical Filters: Their Equivalence to and Difference from Electrical Networks," 1954 IRE Nat'l. Convention Record, pp. 6-12.

31. Chow, C. K., "A Recognition Method Using Neighbor Dependence," IRE Trans. on Electronic Computers, EC-11:5, Oct. 1962, pp. 683-690.

32. Chow, C. K., "Comments on Optimum Character Recognition Systems," IRE Trans. on Electronic Computers, EC-8:2, June 1959, p. 230.

33. Chow, C. K., "Optimum Character Recognition System Using Decision Function," 1957 IRE WESCON Convention Record, Part 4, pp. 121-129.

34. Clark, W. A. and Farley, B. G., "Generalization of Pattern Recognition in a Self-Organizing System," 1955 Proc. Western Joint Computer Conference, pp. 86-91.

35. Cooper, P. W., Classification by Statistical Methods (Pattern Recognition), Tech. Note No. 61/2 (ASTIA No. AD 278 689), Melpar Inc. Watertown, Mass., April 1961, 53p.

36. Cover, T. M., Geometrical and Statistical Properties of Linear Threshold Devices, Tech. Report No. 6107-1, Stanford Electronics Laboratories, System Theory Laboratory, Stanford University, Stanford, Calif., May 1964, 70p.

37. Daly, J., Joseph, R. D., and Kelly, P. M., "Self-Organizing Logic Systems," Astronautica Acta, VIII:Fasc. 6, 1962, pp. 376-400.

38. Daly, J. A., Joseph, R. D., and Ramsey, D. M., "An Iterative Design Technique for Pattern Classification Logic," 1963 Proc. WESCON Convention, Paper 1.3, 6p.

39. Daly, J. A., Joseph, R. D., and Ramsey, D. M., "Perceptrons As Models of Neural Processes," in Computers in Biomedical Research, Vol. I (R. W. Stacy and B. D. Waxman, eds.), Academic Press, New York, 1965, pp. 525-545.

40. Das Gupta, S., "Optimum Classification Rules for Classification into Two Multivariate Normal Populations," Annals of Math. Stat., 36:4, August 1965, pp. 1174-1184.

41. Dertouzos, M. L., "An Approach to Single-Threshold Element Synthesis," IEEE Trans. on Electronic Computers, EC-13:5, Oct. 1964, pp. 519-528.

42. Dusheck, G. J., Hilinski, T. C., and Putzrath, F. L., "A Flexible Neural Logic Network," IEEE Trans. on Military Electronics, MIL-7:2 & 3, April-July 1963, pp. 208-213.

43. Earnest, L. D., "Machine Recognition of Cursive Writing," Proc. IFIP Congress 62 (C. M. Popplewell, ed.), North-Holland Publishing Co., Amsterdam, 1963, pp. 462-466.

44. Estavan, D., Pattern Recognition Learning, and Automated Teaching, Report No. SP-70, System Development Corp., Santa Monica, Calif., April 18, 1959, 8p.

45. Farley, B. G., "Self-Organizing Models for Learned Perception," in Self-Organizing Systems (M. C. Yovits and S. Cameron, eds.), Pergamon Press, New York, 1960, pp. 7-30.

46. Feigenbaum, E. A. and Simon, H. A., "Generalization of an Elementary Perceiving and Memorizing Machine," Proc. IFIP Congress 62 (C. M. Popplewell, ed.), North-Holland Publishing Co., Amsterdam, 1963, pp. 401-406.

47. Feigenbaum, E. A. and Simon, H. A., "Performance of a Reading Task by an Elementary Perceiving and Memorizing Program," Behavioral Science, 8:1, Jan. 1963 pp. 72-76.

48. Fischler, M., "Hyperplane Techniques in Pattern Recognition," Proc. IEEE, 51:3, March 1963, pp. 497-498.

49. Flores, I., "An Optimum Character-Recognition System Using Decision Functions," IRE Trans. on Electronic Computers, EC-7:2, June 1958, p. 180.

50. Frankel, S., "Information-Theoretic Aspects of Character Reading," Proc. Internat'l Conference on Information Processing, UNESCO, 1960, pp. 248-251.

51. Freeman, H., "On the Encoding ot Arbitrary Geometric Configurations," IRE Trans. on Electronic Computers, EC-10:2, June 1961, pp. 260-268.

52. Freeman, H. and Garder, L., "Apictorial Jigsaw Puzzles: The Computer Solution of a Problem in Pattern Recognition," IEEE Trans. on Electronic Computers, EC-13:2, April 1964, pp. 118-127.

53. Frick, F. C., Pattern Recognition, Massachusetts Institute of Technology, Lincoln Laboratory, Lexington, Mass., 6p.

54. Gabelman, I. J., "Properties and Transformations of Single Threshold Element Functions," IEEE Trans. on Electronic Computers, EC-13:6, Dec. 1964, pp. 680-684.

55. Gamba, A., "Optimum Performance of Learning Machines," Proc. IRE, 49:1, Jan. 1961, pp. 349-350.

56. Gerlach, R. K., "Wide-Tolerance Optical Character Recognition for Existing Printing Mechanism, in Optical Character Recognition (G. L. Fischer, Jr., D. K. Pollock, B. Radack, and M. E. Stevens, eds.), Spartan Books, Washington, D. C., 1962, pp. 93-114.

57. Gill, A., "Minimum-Scan Pattern Recognition," IRE Trans. on Information Theory, IT-5:2, June 1959, pp. 52-58.

58. Giuliano, V. E., Jones, P. E., Kimball, G. E., Meyer, R. F., and Stein, B. A., "Automatic Pattern Recognition by a Gestalt Method," Information and Control, 4:4, Dec. 1961, pp. 332-345.

59. Greanias, E. C., "Some Important Factors in the Practical Utilization of Optical Character Readers," in Optical Character Recognition, (G. L. Fischer, Jr., D. K. Pollock, B. Radack, and M. E. Stevens, eds.), Spartan Books, Washington, D. C., 1962, pp. 129-146.

60. Greenberg, H. J. and Konheim, A. G., "Linear and Nonlinear Methods in Pattern Classification," IBM Journal of Research and Development, 8:3, July 1964, pp. 299-307.

61. Green, P. H., "On the Representation of Information by Neural Net Models," in Self-Organizing Systems 1962 (M. C. Yovits, G. T. Jacobi, and G. D. Goldstein, eds.), Spartan Books, Washington, D. C., 1962, pp. 551-563.

62. Griffin, E., "An Optical Character Recognition System Using a Vidicon Scanner," in Optical Character Recognition (G. L. Fischer, Jr., D. K. Pollock, B. Radack, and M. E. Stevens, eds.), Spartan Books, Washington, D. C., 1962, pp. 73-83.

63. Grimsdale, R. L. and Bullingham, J. M., "Character Recognition by Digital Computer Using a Special Flying-Spot Scanner," Computer J., 4 July 1961, pp. 129-135.

64. Guinn, D. F., "Large Artificial Nerve Net (LANNET)," 1963 Proc. Bionics Symposium, Air Force Systems Command, Wright-Patterson Air Force Base, Ohio, 22p.

65. Hannan, W. J., "The RCA Multi-font Reading Machine," in Optical Character Recognition (G. L. Fischer, Jr., D. K. Pollock, B. Radack, and M. E. Stevens, eds.), Spartan Books, Washington, D. C., 1962, pp. 3-14.

66. Harmon, L. D., The Use of Computers in the Simulation of Pattern Recognition (preprint of a paper presented at a Symposium on Pattern Recognition, Univ. of Michigan, Ann Arbor, Oct. 21-23, 1957), Bell Telephone Laboratories, Inc., Murray Hill, N. J., 17p.

67. Hawkins, J. K., "Self-Organizing Systems – A Review and Commentary," Proc. IRE, 49:1, Jan. 1961, pp. 31-48.

68. Hawkins, J. K. and Munsey, C. J., "A Two-Dimensional Iterative Network Computing Technique and Mechanizations," in Workshop on Computer Organization (A. A. Barnum and M. A. Knapp, eds.), Spartan Books, Inc., Washington, D. C., 1963, pp. 93-125.

69. Highleyman, W. H., "An Analog Method for Character Recognition," IRE Trans. on Electronic Computers, EC-10:3, Sept. 1961, pp. 502-512.

70. Highleyman, W. H., "Further Comments on the N-tuple Pattern Recognition Method," IRE Trans. on Electronic Computers, EC-10:1, March 1961, p. 97.

71. Highleyman, W. H., "Linear Decision Functions, with Application to Pattern Recognition," Proc. IRE, 50:6, June 1962, pp. 1501-1515.

72. Highleyman, W. K. and Kamentsky, L. A., "A Generalized Scanner for Pattern and Character Recognition Studies," 1959 Proc. Western Joint Computer Conference, pp. 291-294.

73. Highleyman, W. H. and Kamentsky, L. A., "Comments on a Character Recognition Method of Bledsoe and Browning," IRE Trans. on Electronic Computers, EC-9:2, June 1960, p. 163.

74. Highleyman, W. H. and Kamentsky, L. A., "Pattern Recognition (Perception) Machines," Behavioral Science, 4:3, July 1959, p. 248.

75. Hopcroft, J. E. and Mattson, R. L., "Synthesis of Minimal Threshold Logic Networks," IEEE Trans. on Electronic Computers, EC-14:4, August 1965, pp. 552-560.

76. Joseph, R. D., Contributions to Perceptron Theory, Ph. D. Thesis, Cornell University, Ithaca, N. Y., Sept. 1961, 124p.

77. Joseph, R. D., "On Predicting Perceptron Performance," 1960 IRE Internat'l Convention Record, 8:2, pp. 71-77.

78. Joseph, R. D., Kelly, P. M., and Viglione, S. S., "An Optical Decision Filter," Proc. IEEE, 51:8, August 1963, pp. 1098-1118.

79. Joseph, R. D., Viglione, S. S., and Wolf, H. F., "Cloud Pattern Recognition," 1964 Proc. 19th Nat'l ACM Conference, Paper D2.3, 17p.

80. Kain, R. Y., "Autocorrelation Pattern Recognition," Proc. IRE, 49:6, June 1961, pp. 1085-1086.

81. Kamentsky, L. A., "Pattern and Character Recognition Systems -- Picture Processing by Nets of Neuron Like Elements," 1959 Proc. Western Joint Computer Conference, pp. 304-309.

82. Kanal, L., "Evaluation of a Class of Pattern-Recognition Networks," in Biological Prototypes and Synthetic Systems, Vol. I (E. E. Bernard and M. R. Kare, eds.), Plenum Press, New York, 1962, pp. 261-269.

83. Kay, R. H., "Differential Optical Scanning," IBM Technical Disclosure Bulletin, 4:9, Feb. 1962, p. 79.

84. Keller, H. B., "Finite Automata, Pattern Recognition and Perceptrons," J. Assn. for Computing Machinery, 8 Jan. 1961, pp. 1-20.

85. Kelly, J. L. and Selfridge, O. G., "Sophistication in Computers: A Disagreement," IRE Trans. on Information Theory, IT-8:2, Feb. 1962, pp. 78-80.

86. Kirsch, R. A., Cahn, L., Ray, C., and Urban, G. H., "Experiments in Processing Pictorial Information with a Digital Computer," 1957 Proc. Eastern Joint Computer Conference, pp. 221-230.

87. Laemmel, A., Linear Statistical Classification, Report No. 54 G-0021 (ASTIA No. AD 237 125), Massachusetts Institute of Technology, Lincoln Laboratory, Lexington, Mass., May 16, 1960, 9p.

88. Lee, R. J., Gilstrap, L. O., Jr., Snyder, R. F., and Pedelty, M. J., Theory of Probability State Variable Systems, Vols. I-VI, Report No. ASD-TDR-63-664, Adaptronics, Inc., Alexandria, Va., Dec. 1963, 1025p.

89. Leimer, J. J., "Design Factors in the Development of an Optical Character Recognition Machine," IRE Trans. on Information Theory, IT-8:2, Feb. 1962, pp. 167-171.

90. Lewis, P. M. II., "A Note on Realization of Decision Networks Using Summation Elements," Information and Control, 4:3, Sept. 1961, pp. 282-290.

91. Lewis, P. M., "The Characteristic Selection Problem in Recognition Systems," IRE Trans. on Information Theory, IT-8:2, Feb. 1962, pp. 171-178.

92. Liu, C. N., "A Programmed Algorithm for Designing Multifront Character Recognition Logics," IEEE Trans. On Electronic Computers, EC-13:5, Oct. 1964, pp. 586-593.

93. Lohninger, W. J., Photometric Character Recognition Devices, U. S. Patent No. 2,927, 216, March 1, 1960, 13p.

94. Marill, T. and Green, D. M., "Statistical Recognition Functions and the Design of Pattern Recognizers," IRE Trans. on Electronic Computers, EC-9:4, Dec. 1960, pp. 472-477.

95. Maron, M. E., Artificial Intelligence and Brain Mechanisms, Memo. No. RM-3522-PR (ASTIA No. AD 299 478), The RAND Corp., Santa Monica, Calif., March 1963, 35p.

96. Maron, M. E., Mechanisms Underlying Predictive Behavior for an Intelligent Machine, Memo. No. RM-3011-PR (ASTIA No. AD 272 894), The RAND Corp., Santa Monica, Calif., Feb. 1962, 29p.

97. Marrill, T., Hartley, A. K., Evans, T. G., Bloom, B. H., Park, D. M. R., Hart, T. P., and Darley, D. L., "Cyclops-1: A Second-Generation Recognition System," Proc. Fall Joint Computer Conference, 1963 (AFIPS), pp. 27-33.

98. Marzocco, F. N., "Computer Recognition of Handwritten First Names," IEEE Trans. on Electronic Computers, EC-14:2, April 1965, pp. 210-217.

99. Marzocco, F. N. and Bartram, P. R., "Statistical Learning Models for Behavior of an Artificial Organism," in Biological Prototypes and Synthetic Systems, Vol. I (E. E. Bernard and M. R. Kare, eds.), Plenum Press, New York, 1962, pp. 88-96.

100. Mattson, R., A Self-Organizing Binary System, Report No. LMSD-288029 (ASTIA No. AD 227 943), Lockheed Aircraft Corp., Missiles and Space Division, Sunnyvale, Calif., Sept. 1959, 25p.

101. Mattson, R. L., "A Self-Organizing Binary System," 1959 Proc. Eastern Joint Computer Conference, pp. 212-217.

102. Mattson, R. L., "An Adaptive Classifier," in High Speed Computer System Research, Quarterly Progress Report 6 (ASTIA No. AD 231 166), Massachusetts Institute of Technology, Computer Components and Systems Group, April 30, 1959, pp. 29-33.

103. Mattson, R. L., An Approach to Pattern Recognition Using Linear Threshold Devices, Tech. Report: Mathematics No. LMSD-702680 (ASTIA No. AD 246 244), Lockheed Aircraft Corp., Missiles and Space Division, Synnyvale, Calif., Sept. 1960, 31p.

104. Mattson, R. L., The Analysis and Synthesis of Adaptive Systems Which Use Networks of Threshold Elements, Tech. Report No. 1553-1, Stanford Electronics Laboratories, Solid-State Electronics Laboratory, Stanford University, Stanford, Calif., Dec. 1962, 95p.

105. Mattson, R. L. and Firschein, O., Feature Word Construction for Use with Pattern Recognition Algorithms: An Experimental Study, Tech. Report: Mathematics No. 6-90-62-58 (ASTIA No. AD 287 898), Lockheed Missiles & Space Co., Sunnyvale, Calif., August 1962.

106. Mattson, R. L., Firschein, O., and Fischler, M., Methods of Increasing Redundancy in a Class of Pattern Recognition Synthesis Algorithms: An Experimental Study, Tech. Report: Mathematics No. 6-90-62-83 (ASTIA No. AD 295 696), Lockheed Missile & Space Co., Sunnyvale, Calif., Sept. 1962, 47p.

107. McCarthy, J. and Minsky, M. L., "Artificial Intelligence-Research Objectives," in Quarterly Progress Report 52, Massachusetts Institute of Technology, Research Laboratory of Electronics, Cambridge, Mass., Jan. 15, 1959, p. 129.

108. Minsky, M. L., "Machine Learning, Character Recognition, and Perception," in Frontier Research on Digital Computers (J. W. Carr III and M. D. Spearman, eds.), Univ. of North Carolina Press, Chapel Hill, N. C., 1959, pp. 1(X) - 25(X).

109. Mosteller, F. and Wallace, D., "Notes on an Authorship Problem," (in the Proceedings of a Harvard Symposium on Digital Computers and Their Applications), Annals of the Computation Laboratory of Harvard University, XXXI, 1962, pp. 163-197.

110. Muroga, S., "Lower Bounds of the Number of Threshold Functions and a Maximum Weight," IEEE Trans. on Electronic Computers, EC-14:2, April 1965, pp. 136-148.

111. Murray, A. E., Perceptron Applicability to Photointerpretation-Phase I Interim Report, Report No. VE-1446-6-1, Cornell Aeronautical Laboratory, Inc., Buffalo, N. Y., Nov. 1, 1960, 45p.

112. Nadler, M., "An Analog-Digital Character Recognition System," IEEE Trans. on Electronic Computers, EC-12:6, Dec. 1963, pp. 814-821.

113. Neisser, U. and Weene, P., "A Note on Human Recognition of Hand-Printed Characters," Information and Control, 3:2, June 1960, pp. 191-196.

114. Nelson, E. E., Daly, J. A., and Joseph, R. D., "Time-Varying Threshold Logic," in Biophysics and Cybernetic Systems (M. Maxfield, A. Callahan, and L. J. Fogel, eds.), Spartan Books, Inc., Washington, D. C., 1965, pp. 101-113.

115. Nilsson, N. J., Learning Machines: Foundations of Trainable Pattern Classifying Systems, McGraw-Hill Systems Science Series, New York, 1965, 137p.

116. Papert, S., "Some Mathematical Models of Learning," in Information Theory: Fourth London Symposium (C. Cherry, ed.), Butterworths, Washington, D. C., 1961, pp. 353-363.

117. Pedelty, M. J., A Review of the Field of Artificial Intelligence and Its Possible Applications to NASA Objectives, Final Report on NASA Contract NASr-229, The American University, School of Government and Public Administration, Washington, D. C., Feb. 1965, 16p.

118. Pierce, W. H., "Adaptive Decision Elements to Improve the Reliability of Redundant Systems," 1962 IRE International Convention Record, 10:4, pp. 124-131.

119. Platt, J. R., "How We See Straight Lines," Scientific American, 202:6, June 1960, pp. 121-122, 124-126, 128-129.

120. Prather, R. C. and Uhr, L. M., Discovery and Learning Techniques for Pattern Recognition, Report No. SP-1561/000/01 (ASTIA No. AD 610 725), System Development Corp., Santa Monica, Calif., Nov. 18, 1964, 16p.

121 Rabinow, J., "Developments in Character Recognition Machines at Rabinow Engineering Company," in Optical Character Recognition (G. L. Fischer, Jr., D. K. Pollock, B. Radack, and M. E. Stevens, eds.), Spartan Books, Washington, D. C., 1962, pp. 27-50.

122. Reineds, J., Synthesis of a Character Recognition Machine, Research Report RC-451, IBM Corp. Research Center, Yorktown, N. Y., May 12, 1961, 50p.

123. Research in Systems Theory, Devices, and Physical Phenomena for Microsystem Electronics, Final Tech. Documentary Report No. AL-TDR-64-81, Stanford Electronics Laboratories, Solid-State Electronics Laboratory, Stanford University, Stanford, Calif., June 1964, 276p.

124. Roberts, L. G., Pattern Recognition with an Adaptive Network, Report 51G-0013 (ASTIA No. Ad 236 325), Massachusetts Institute of Technology, Lincoln Laboratory, Lexington, Mass., April 26, 1960, 10p.

125. Rosenblatt, F., Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms, Spartan Books, Washington, D. C., 1962, 616p.

126. Rosenfeld, A., "Automatic Recognition Techniques Applicable to High-Information Pictorial Inputs," 1962 IRE Internat'l Convention Record, 10:4, pp. 114-123.

127. Rushforth, C. K., Detection of a Pattern in Unknown Position, Scientific Report No. 2 (ASTIA No. AD 610 712), Utah State University, Electro-Dynamics Laboratories, Logan, Utah, Nov. 1, 1964, 22p.

128. Sebestyen, G. S., Classification Decisions in Pattern Recognition, Tech. Report 381, Massachusetts Institute of Technology, Research Laboratory of Electronics, Cambridge, Mass., April 25, 1960, 79p.

129. Sebestyen, G. S., Decision-Making Processes in Pattern Recognition, The Macmillan Co., New York, 1962, 162p.

130. Sebestyen, G. S., "Pattern Recognition by an Adaptive Process of Sample Set Construction," Proc. IRE (Professional Group on Information Theory), IT-8:5, Sept. 1962, pp. S82-S91.

131. Sebestyen, G. S., "Recognition of Membership in Classes," IRE Trans. on Information Theory, IT-7:1, Jan. 1961, pp. 44-50.

132. Sebestyen, G. and Edie, J. Pattern Recognition Research, Final Report AFCRL-64-821 (ASTIA No. AD 608 692), Litton Systems, Inc., Information Sciences Laboratory, Waltham, Mass., June 14, 1964, 123p.

133. Sheng, C. L.,"A Method for Testing and Realization of Threshold Functions," IEEE Trans. on Electronic Computers, EC-13:3, June 1964, pp. 232-239.

134. Shepard, R. N., Application of IPL-V to the Simulation of Perceptual Learning (preprint of a paper presented at the American Psychological Assn. conference on "Information-Processing Languages for Digital Computers: A Technique for Model Building," Cincinnati, Ohio, Sept. 8, 1959), Bell Telephone Laboratories, Murray Hill, N. J., 5p.

135. Shepard, R. N., "Computers and Thought - A Review," (E. Feigenbaum and J. Feldman, eds., McGraw-Hill, New York, 1963), Behavioral Science, 9:1, Jan. 1964, pp. 57-65.

136. Singer, J. R., "A Self Organizing Recognition System," 1961 Proc. Western Joint Computer Conference, pp. 545-554.

137. Stone, W. P., Alpha-Numeric Character Reader, RADC Report No. TN 56-194 (ASTIA No. AD 103 237), Rome Air Development Center, Rome, N. Y., June 1956, 13p.

138. Uffelman, M. R., "CONFLEX I - A Conditioned Reflex System," 1962 IRE Internat'l Convention Record, 10:4, pp. 132-139.

139. Uhr, L., "A Possibly Misleading Conclusion As to the Inferiority of One Method for Pattern Recognition to a Second Method to Which It Is Guaranteed to Be Superior," IRE Trans. on Electronic Computers, EC-10:1, March 1961, pp. 96-97.

140. Uhr, L. and Vossler, C., "Suggestions for Self-Adapting Computer Models of Brain Functions," Behavioral Science, 6:1, Jan. 1961, pp. 91-97.

141. Uhr, L. and Vossler, C., "The Search to Recognize," in Optical Character Recognition (G. L. Fischer, Jr., D. K. Pollock, B. Radack, and M. E. Stevens, eds.), Spartan Books, Washington, D. C., 1962, pp. 319-329.

142. Uhr, L., Vossler, C., and Uleman, J., "Pattern Recognition over Distortions, by Human Subjects and by a Computer Simulation of a Model for Human Form Perception," J. Exper. Psychol., 63:3, 1962, pp. 227-234.

143. Unger, S. H., "A Computer Oriented Toward Spatial Problems," Proc. IRE, 45:10, Oct. 1958, pp. 1744-1750.

144. Unger, S., "Pattern Detection and Recognition," Proc. IRE, 47:10, Oct. 1959, pp. 1737-1752.

145. Uttley, A. M., "Conditional Probability Computing in a Nervous System," in Mechanisation of Thought Processes, Vol. I (Proc. of the National Physical Laboratory Symposium No. 10), Her Majesty's Stationery Office, London, 1959, pp. 121-147.

146. Uttley, A. M., "Imitation of Pattern Recognition and Trial-and-Error Learning in a Conditional Probability Computer," Reviews of Modern Physics, 31:2, April 1959, pp. 546-548.

147. Uttley, A. M., "Temporal and Spatial Patterns in a Conditional Probability Machine," in Automata Studies (C. E. Shannon and J. McCarthy, eds.), Princeton University Press, Princeton, N. J., 1956, pp. 277-285.

148. Vossler, C. and Uhr, L., A Computer Simulation of Pattern Perception and Concept Formation, Report No. SP-459 (ASTIA No. AD 276 378), System Development Corp., Santa Monica, Calif., August 25, 1961, 31p.

149. Widrow, B., Adaptive Sampled-Data Systems, Tech. Report No. 2104-1 (ASTIA No. AD 243 265), Stanford Electronics Laboratories, Stanford University, Stanford, Calif., July 15, 1960, 41p.

150. Widrow, B. and Hoff, M. E., "Adaptive Switching Circuits," 1960 IRE WESCON Convention Record, Part 4, pp. 96-104.

151. Widrow, B. and Hoff, M. E., Adaptive Switching Circuits, Tech Report No. 1553-1 (ASTIA No. AD 241 531), Stanford Electronics Laboratories, Solid-State Electronics Laboratory, Stanford University, Stanford, Calif., June 30, 1960, 34p.

152. Widrow, B., Pierce, W. H., and Angell, J. B., Birth, Life, and Death in Microelectronic Systems, Tech. Report No. 1552-2/1851-1 (ASTIA No. AD 259 538), Stanford Electronics Laboratories, Solid-State Electronics Laboratory, Stanford University, Stanford, Calif., May 30, 1961, 33p.

153. Williams, J. H., Jr., "A Discriminant Method for Automatically Classifying Documents," Proc. Fall Joint Computer Conference, 1963 (AFIPS), pp. 161-166.

154. Winder, R. O., Threshold Logic in Artificial Intelligence, Scientific Report No. 6, Radio Corp. of America, RCA Laboratories, Princeton, N. J., Nov. 15, 1962, 26p.

155. Lunar Atlas, Vol. I, First Edition, University of Chicago, February 1960.

156. Whitaker, E. A., Kuiper, G. P., Hartmann, W. K., and Spradley, L. H., Rectified Lunar Atlas, Supplement Number 2 to the USAF Lunar Atlas, University of Arizona, 1963.

157. Widrow B., "Generalization and Information Storage In Networks of Adaline Neurons," Self Organizing Systems 1962, Spartan Books, 1962.

158. Parzen, E., "On Estimation of a Probability Density Function and Mode," Annals of Math.Stat., 33, 1067-1076, 1962.

159. Murthy, V. K., Nonparametric Estimation of Multivariate Densities with Applications, Douglas Aircraft Company, Santa Monica, Calif., Paper No. 3490, 1965.

160. Design of a Cloud Pattern Recognition System, Final Report No. SM-46215-F, Douglas Aircraft Co., Newport Beach, Calif., 1965.

161. Tippet, et al (eds.), Optical and Electro-Optical Processing, MIT Press, 1965.

162. Hunter, C. M., et al, Automatic Picture Transmission System, NASA Document N63-18619.

163. Research on the Utilization of Recognition Techniques to Identify Classify Objects in Video Data, Report No. SM-48464-TPR-3, Douglas Aircraft Co., Newport Beach, August 1966.

164. American Institute of Physics Handbook, Second Edition, Mc Graw Hill, p. 2-141.

165. Reference Data for Radio Engineers, Fourth Edition, IT & T Corp., p. 401.

166. Eberhardt, E. H., "Signal-to-Noise Ratio in Image Dissectors," ITT Industrial Laboratories, Research Memo No. 386.

167. Bahadur, R. R., "On Classification Base on Responses to n Dichotomous Items," USAF SAM series in Statistics, Randolf AFB, Texas, 1959.

"The aeronautical and space activities of the United States shall be conducted so as to contribute . . . to the expansion of human knowledge of phenomena in the atmosphere and space. The Administration shall provide for the widest practicable and appropriate dissemination of information concerning its activities and the results thereof."

—NATIONAL AERONAUTICS AND SPACE ACT OF 1958

NASA SCIENTIFIC AND TECHNICAL PUBLICATIONS

TECHNICAL REPORTS: Scientific and technical information considered important, complete, and a lasting contribution to existing knowledge.

TECHNICAL NOTES: Information less broad in scope but nevertheless of importance as a contribution to existing knowledge.

TECHNICAL MEMORANDUMS: Information receiving limited distribution because of preliminary data, security classification, or other reasons.

CONTRACTOR REPORTS: Scientific and technical information generated under a NASA contract or grant and considered an important contribution to existing knowledge.

TECHNICAL TRANSLATIONS: Information published in a foreign language considered to merit NASA distribution in English.

SPECIAL PUBLICATIONS: Information derived from or of value to NASA activities. Publications include conference proceedings, monographs, data compilations, handbooks, sourcebooks, and special bibliographies.

TECHNOLOGY UTILIZATION PUBLICATIONS: Information on technology used by NASA that may be of particular interest in commercial and other non-aerospace applications. Publications include Tech Briefs, Technology Utilization Reports and Notes, and Technology Surveys.

Details on the availability of these publications may be obtained from:

SCIENTIFIC AND TECHNICAL INFORMATION DIVISION

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Washington, D.C. 20546