Final Report

Design Study of a Cloud Pattern Recognition System

(15 June 1964 - 15 September 1965)

Contract No.: NAS5-3866

Prepared By

Missile and Space Systems Division
Astropower Laboratory
Electronics Department
Douglas Aircraft Company, I
Newport Beach, Californi

for

Goddard Space Flight Center
Greenbelt, Maryland

Final Report

Design Study of a Cloud Pattern Recognition System

(15 June 1964 - 15 September 1965)

Contract No.: NAS 5-3866

Prepared By

Missile and Space Systems Division
Astropower Laboratory
Electronics Department
Douglas Aircraft Company, Inc.
Newport Beach, California

Approved By

S. S. Vighone
Manager, Electronics Department

for

Goddard Space Flight Center
Greenbelt, Maryland

ABSTRACT

An extensive investigation of a particular design technique for pattern recognition, and particularly its application to TIROS photographs, has been conducted. Many aspects of the technique which were not specific to this application were investigated in a series of experiments on alphabetic characters. Other experiments were performed on data from actual TIROS frames in an attempt to separate frames containing vortices from those which do not.

In the studies on alphabetic characters, the design technique was able to provide networks giving perfect separation of the sample patterns with a very small number of property filters. The generalization performance of these networks is quite good in comparisons with classification systems produced using several other design techniques. A method for building invariance to linear changes in the gray scale into the property filters is developed.

Optical studies to determine the resolution necessary for untrained observers to classify vortex patterns in TIROS frames were conducted. In a large number of cases no agreement was obtained by the panel, even at the highest resolution level used (240 lines). In most cases final classification occurred either at the highest resolution level or at the lowest (70 lines), the cases occurring with about equal frequencies.

Networks were designed to separate a sample of 1000 high-resolution TIROS frames. Again, the technique provides perfect separation of the sample patterns with a very limited set of property filters — in this case, about 240. Generalization performance, tested on an additional 200 frames, is not good, but consistently better than chance. An error rate as low as 36.5% was achieved, but a more representative rate would be 40%. Further studies indicate that this result is probably due to non-representativeness of the sample patterns.

Means for improving these results by normalization of the patterns and by some decoupling of the design technique are suggested.

FOREWORD

This report was prepared by the Astropower Laboratory, Missile and Space Systems Division, Douglas Aircraft Co., under Contract NAS 5-3866. The contract was administered by the Information Technology Group, Goddard Space Flight Center, National Aeronautical and Space Administration, Greenbelt, Maryland, with Mr. J. R. Silverman as Project Engineer.

The work performed on this study program began on 15 June 64. This document is the final report on the contract and was preceded by monthly status reports and three quarterly progress reports. The work was accomplished by the personnel of the Electronics Laboratory of Astropower. The principal contributors were:

Dr. R. D. Joseph – Project Leader
A. G. Mucci
A. G. Ostensoe
R. G. Runge
M. Uemura
S. S. Viglione – Program Manager

The computing facilities at the Douglas Space Center in Huntington Beach, California were utilized for the simulation studies. Acknowledgments are due to Mssrs. T. Mueller, D. Mercer, and H. Belcher of the computing center for their assistance and support, particularly during the last few weeks of the program; to Messrs. V. Dvorak and H. Q. Van Dyke, Weather Bureau Satellite Center, Pt. Mugu, for their time, effort and patience in classifying the TIROS cloud photographs and to Mr. H. F. Wolf, formerly of the Electronics Lab, for his contributions to the earlier formative stages of this program.

TABLE OF CONTENTS

		Page
1.	INTRODUCTION	1
	1.1 Background Discussion	1
	1.2 Scope of the Present Program	2
2.	EVALUATION OF RESULTS	5
	2.1 Summary and Conclusions	5
	2.2 Recommendations	13
3.	GENERAL DISCUSSION OF PATTERN RECOGNITION	15
	3.1 Pattern Recognition Mechanisms	15
	3.2 Signal Conditioning	17
	3.3 Known Property Extraction	21
	3.4 Statistical Property Extraction	22
	3.5 Pattern Identification	25
4.	DESIGN CONSIDERATIONS	31
	4.1 General Discussion	31
	4.2 Iterative Design	32
	4.3 Discriminant Analysis	36
	4.3.1 Oriented Hyperplane	41
	4.3.2 Perpendicular Bisector	42
	4.3.3 Quadratic Surface	43
	4.4 Gray Scale Invariance	44
	4.5 Approximation to the Quadratic Surface	47
	4.6 Convergence Proof for Iterative Design	50
	4.7 Hardware Considerations	54
	4.7.1 Discussion	54
	4.7.2 Property Filter Mechanization	55
	4.7.3 System Considerations	76
	4.8 Summary of Results	84
5.	STUDIES ON ALPHABETIC CHARACTERS	90
	5.1 Topics	90

			Page
5.2	The Sample Problem		92
	5.2.1	The Design Techniques	92
	5.2.2	Sample Patterns	95
	5.2.3	The Sample Task	106
	5.2.4	Correlation Analysis of the Sample Patterns	107
5.3	Computer Programs		108
5.4	Results		116
	5.4.1	Random Variations	116
	5.4.2	Parameter Variations	119
	5.4.3	Gray Invariance	129
	5.4.4	Distributional Assumptions	138
	5.4.5	Technique Complexity	143
	5.4.6	Random Disturbances	146
	5.4.7	Standard Perceptrons	151
5.5	Summary of Results		154
6.	STUDIES ON CLOUD PATTERNS		161
6.1	Topics		161
6.2	The Sample Problem		162
	6.2.1	The Design Technique	162
	6.2.2	The Sample Cloud Patterns	165
	6.2.3	Modification of the Sample Frames	167
6.3	Computer Programs		171
6.4	Results		177
	6.4.1	Rate of Design	177
	6.4.2	Rate of Activity	179
	6.4.3	Separability of the Patterns	179
	6.4.4	Generality of Design	189
	6.4.5	Continuation of Design	201
	6.4.6	Input Field Restriction	201
6.5	Resolution Studies		203
	6.5.1	Photographic Studies	203
	6.5.2	Computer Studies	210
6.6	Summary of Results		215
REFERENCES			219

LIST OF ILLUSTRATIONS

Figure		Page
1	Pattern Recognition Mechanism	16
2	Decision Network Structure	33
3	Discriminant Functions	39
4	Hardware vs Discriminant Function Complexity	56
5	Operational Amplifier	58
6	Differential Voltage Comparator	59
7	Linear Threshold Logic Unit (I.C.)	60
8	Linear Threshold Logic Units	62
9	Parallel Hyperplane (I.C.)	63
10	Parallel Hyperplane (laboratory model)	65
11	Function Generated by Thyrite with Series Resistor	68
12	Normalized Parallel Hyperplane	69
13	Switching Surface for Normalized Parallel Hyperplane	71
14	Quadratic Surface Mechanization	74
15	General Purpose Recognition System	78
16	Tape Format	79
17	Arithmetic Unit	83
18	Deck 1 Sample Patterns	96
19	Deck 1 Sample Patterns	97
20	Deck 1 Sample Patterns	98
21	Deck 1 Sample Patterns	99
22	Deck 1 Sample Patterns	100

Figure		Page
23	Deck 2 Verification Patterns	101
24	Deck 2 Verification Patterns	102
25	Deck 2 Verification Patterns	103
26	Deck 2 Verification Patterns	104
27	Deck 2 Verification Patterns	105
28	Flow Chart for Main Program	111
29	Flow Chart for Generalization Program (VERPR)	114
30	Effects of Randomness - System Loss	117
31	Effects of Randomness - System Loss	118
32	Errors as a Function of System Loss	120
33	Effect of Iterative Cycles — Perpendicular Bisector Technique	121
34	Effects of Selection Ratio — System Loss	124
35	Effects of Number of Input Connections — System Loss	126
36	Gray Invariance Comparisons — System Loss	130
37	Gray Invariance Comparisons — Learning Rate	131
38	Gray Invariance Comparison — Rate of Decrease in System Loss	132
39	Gray Invariance Comparisons — Generalization Errors	133
40	Normalized Comparison of Gray Invariance — System Loss	136
41	Degrees of Freedom Comparisons — System Loss	137
42	Effects of Gray Scale Patterns — Deck I, 7x3	139
43	Effects of Gray Scale Patterns — Deck II, 7x3	140

Figure		Page
44	Effects of Gray Scale Patterns — Deck I, VAR	141
45	Effects of Gray Scale Patterns — Deck II, VAR	142
46	Effects of Distributional Assumptions — System Loss	144
47	Effects of Technique Complexity — System Loss	145
48	Technique Complexity Comparison — Learning Rate	147
49	Technique Complexity Comparison — System Loss	148
50	Technique Complexity Comparison — Rate of Decrease in System Loss	149
51	Technique Complexity Comparisons — Generalization Errors	150
52	Effects of Random Perturbations — Parallel Hyperplane Technique	152
53	Learning Procedures Applied to Deck I	155
54	Generalization with Deck II	156
55	Program Flow Chart	174
56	Rate of Design (Beta)	178
57	Rate of Activity (COSPHI)	180
58	Machine Design #1	182
59	Machine Design #2	183
60	Machine Design #3	184
61	Machine Design #4 — Reduced Resolution Patterns (90 x 90)	185
62	Machine Design #5 — Reduced Resolution Patterns (60 x 60)	186
63	Machine Design #6 — Logic Unit Input Field Limited (90 x 90)	187

Figure		Page
64	Machine Design #7 — Logic Unit Input Field Limited (45 x 45)	188
65	A Special Design Technique	191
66	Network Performance — Machine #1 Plus #2 Plus #3	192
67	Network Performance — Machine #2 Plus #3 Plus #1	193
68	Network Performance — Machine #3 Plus #1 Plus #2	194
69	Network Performance — Machine #4	195
70	Network Performance — Machine #5	196
71	Network Performance — Machine #6	197
72	Network Performance — Machine #7	198
73	Pattern 150 at Resolutions of 122, 180, 220, and 320	209
74	Pattern 48 at Resolutions of 122, 150, 180 and 320	211
75	Pattern 80 at Resolutions of 133, 165, 200 and 329	212
76	Pattern 30 at Resolutions of 95, 111, 150 and 220	213

LIST OF TABLES

Table		Page
I	Technique Comparison	38
II	Series Resistor Values	67
III	Operation of Normalized Parallel Hyperplane Circuit	72
IV	Instruction List	85
V	Technique Comparison	94
VI	Generalization Errors Using Highest Individual Correlation Classification Scheme	109
VII	Effects of Iterations on System Loss Perpendicular Bisector Technique; Designed on Deck 2; DELOSU vs AHINRT	122
VIII	Effects of Cone Angle on System Loss Gray Invariant Parallel Hyperplanes Technique; Incomplete Networks; DELOSU vs AHINRT	128
IX	Training Sample	168
X	Generalization Sample	170
XI	Classification Results in Resolution Study	208

1. INTRODUCTION

1.1 Background Discussion

Present processing techniques convert the returned data from the TIROS satellite, and associated communication and tracking system, into an area map with general descriptors of weather conditions in the traversed region. Manual analysis of the high resolution photographs by trained meteorologists then provides inputs, concerning atmospheric stability, wind flow, storm centers, fronts, etc., to be superimposed on the computer-generated maps. The final weather maps, the results of a merger of computer and manual analysis, are then transmitted to weather centrals providing timely, extensive weather data for user consumption.

This highly automated system is necessitated by the extensive amount of weather data presently being accumulated on a routine basis by the TIROS satellite system. With world wide coverage on a timely basis as the goal of the meteorological satellites, additional satellites will be placed in orbit, further increasing the amount of processing required. As suggested above, much of the data can be processed by computer. The task that currently defies automatic handling is the recognition of the cloud formations and interpretation of the cloud cover photographs. If a device were developed that could recognize cloud features such as storm centers, cloud streets, cloud fronts, etc., and interpret this data, it is conceivable that future weather forecasting systems could be further automated and the burden that has been placed on the meteorologist alleviated.

Under contract NASw609,[1] Astropower Laboratory of Douglas Aircraft Company investigated the feasibility of performing the recognition task with a self-organizing, parallel logic system categorized as a "forced learning perceptron." Using optical correlation of actual cloud cover negatives and an estimation procedure mechanized on the IBM 7094, the size of the parallel layer required to achieve effective recognition of vortex pattern was to be determined. Program results showed

that a "forced learning perceptron" was capable of correctly identifying only 65 to 70 percent of the patterns examined. This indicated that poor class separation existed between the vortex and nonvortex patterns, due in part to the inaccuracies in the optical processing and correlation procedures involved, and in part to the inherent overlap of the patterns themselves.

To achieve more effective recognition, the design procedure must account for the pattern similarities. Studies devoted to advanced self-organizing procedures led to the development of the discriminant-analysis iterative design algorithm. This design routine could be implemented in a digital computer simulation and the digitized cloud patterns examined directly, thus overcoming the inaccuracies of the optical correlation procedure.

1.2 Scope of the Present Program

The purpose of this project was to extend the study of the feasibility of automatic TIROS photograph analysis. A specific objective was to arrive at the design specification for a feasibility model of an automatic vortex recognition system.

The study had four main phases — (1) the development of logical design techniques applicable to the analysis of TIROS photographs, (2) experimental investigations of selected design parameters on a simplified problem, (3) the simulated design of a vortex recognition system using actual TIROS photographs, and (4) the consideration of suitable hardware for implementing the system.

In the study of the design approach, a variation of the discriminant analysis-iterative design technique was developed. A description of this technique is given in Section 4. Several discriminant analyses, in which alternative pattern differences are assumed are described (Section 4.3). Only one of these, one in which differences in covariance matrices of the pattern distributions are exploited, is suitable for the cloud pattern analysis. This discriminant

analysis yields a quadratic discriminant surface and requires complex hardware for mechanization of the resulting first layer logic units. To simplify the hardware implementation of the system, an approximation to the quadratic unit was developed. The approximate unit is derived from a principal axis solution, and substitutes a pair of parallel hyperplanes for the quadratic switching surface of the more complex unit (Section 4.5). Methods have been developed to make the system invariant to changes in the brightness and contrast of the input patterns (Sections 4.4 and 4.5). This invariance is considerably more effective than a simple normalization of the input pattern, as it is achieved by making each logic unit invariant to such changes. The iterative design process itself is a means for assigning output weights to the logic units, and for emphasizing the difficult patterns. In common with the popular error correction methods, iterative design will find a solution whenever it is possible to assign these output weights to give perfect performance on the sample patterns (Section 4.6). Unlike error correction, iterative design provides an "optimum" set of weights when no solution exists, and maximizes the switching surface to pattern distances when one does exist. Possible methods for hardware mechanization of the resulting system are discussed in Section 4.7.

A portion of the experimental program has been performed on a simplified problem using low resolution, hand-printed alphabetic characters. These studies were used to investigate selected aspects of the design technique, rather than its application. The simplified problem permitted a very much more extensive investigation than would be possible on the cloud patterns. The average computer simulation time (IBM 7094) to design a recognition network for cloud patterns was five hours — for the alphabetic characters, five minutes. Two sets of alphabetic patterns were obtained, each consisting of twenty examples of twelve letters. One of these sets was used to design recognition networks, the other to test them. The topics studied include the cost of providing contrast and brightness invariance, the cost of the parallel hyperplane approximation,

the effects of the distributional assumptions, the variability of machine design, the effects of the number of input connections per logic unit, the number of iterative cycles, the number of logic units designed for each one selected for the network, the effect of perturbations of the weights on the recognition network performance, and a comparison with some standard perceptron techniques. The results of these studies are presented in Section 5.4, and summarized in Sections 2.1.and 5.5. . The sample problem is discussed in Section 5.2, and the computer programs utilized are described in Section 5.3.

Certain topics, namely those which are particular to TIROS photograph analysis, or the application of the design technique to cloud pattern analysis, are not suitable for the sample problem investigation. Studies were therefore performed on actual TIROS cloud photographs. In one study, the resolution of negatives taken from the TV display were modified by a screen process. Prints of these modified resolution frames were examined by a large number of people to determine the resolution requirements of untrained personnel for vortex recognition (Section 6.5.1). The remaining studies were performed on patterns stored as digitized video signals. The study of primary interest was whether or not a vortex recognition network could be designed that would give good generalization results, i.e., a network capable of recognizing vortices in cloud photographs not included in the training or learning sample. Pattern sets of 1000 patterns and 200 patterns were utilized, the former for network design and the latter for generalization testing. Other topics studied were the effect of changing the resolution of the design set of patterns, the effect of changing the rate of network design, the effect of merging several network designs, the effect of limiting the input fields of the logic units, and the effect of varying a parameter which controls the proportion of patterns which turn the logic units on. The results of these studies are given in Sections 6.4 and 6.5, and are summarized in Sections 2.1 and 6.6. A description of the sample cloud patterns and the design technique are given in Section 6.2, and the computer programs described in Section 6.3. Actual network designs resulting from the simulation programs are discussed in Section 6.4.

2. EVALUATION OF RESULTS

2.1 Summary and Conclusions

An extensive study of a particular design technique for pattern recognition, and particularly its application to TIROS photographs, has been conducted. Many aspects of the technique which were not specific to this application were investigated in a series of experiments on alphabetic characters. Other experiments were performed on data from actual TIROS frames in an attempt to separate frames containing vortices from those which do not.

Automatic design techniques for pattern recognition networks may be given varying degrees of coupling to a sample of patterns. At one extreme, the technique may be tightly coupled in that the only design criterion is the correct classification of the design sample of patterns. The drawback to this extreme is that even if the sample of patterns is very representative of the actual distributions, the finiteness of the sample can lead to networks giving poor generalization results. A non-representative sample almost insures poor generalization. The advantages are (1) that there is no need to make distributional assumptions; and (2) that since the goal is a high level of performance on all patterns, and since it is unlikely that the generalization performance will exceed the performance on the design sample, perfect or near perfect performance on the design sample is a good starting point. At the other extreme are the techniques with minimal coupling to the design sample. A design is accomplished by making strong assumptions concerning the actual distributions of patterns, using the sample to estimate a limited number of parameters of these distributions. The disadvantages are that if the distributional assumptions are incorrect, then poor performance is obtained. Even if the assumptions are nearly correct, the technique usually results in networks which classify a majority of the patterns correctly, but leave a substantial minority incorrect. Increasing the sample size does not help usually, for although the less common patterns are better

represented, only a few average values are extracted from the sample. The advantage is that the number of sample patterns required by the technique is minimized, since the technique is not nearly as sensitive to noise and false clues in the sample patterns.

The present technique was selected to provide a compromise between these extremes. Initially, the design of the property filters is very loosely coupled in that populations of them are designed statistically under very strong distributional assumptions. The selection of property filters and the design of a linear discriminant for the decision element are tightly coupled to the design sample. This is done in terms of a loss function which reflects how well each sample pattern is classified. The losses are used as weightings in the statistical averages used to design property filters so that as the design process proceeds, the design of these filters becomes more tightly coupled to the sample patterns. The intent was to derive properties in a loosely coupled fashion to avoid the sensitivity to sample representativeness of the tightly coupled systems. The purpose of increasing the extent of this coupling was to avoid the common failing of loosely coupled systems — that of solving the same portion of the problem over and over, while ignoring the harder parts of the problem.

The effects of this increase in coupling are evident in the experimental programs, in which the generalization performance nearly always does not improve, and often deteriorates as the last few errors in the design phase are eliminated. Overall, the design technique appears to be tightly coupled, having never indicated an inability to provide perfect performance on the sample patterns with a relatively small network.

Techniques were devised to provide property filters which were invariant to linear changes in the gray scale. Experiments on the alphabetic characters indicated that the increase in network complexity required for perfect performance on the sample patterns was nominal, although for the type of property filter selected, the gray invariance was costly in the hardware implementation.

Consideration of the TIROS frame analysis indicates that the property filters must be designed by a covariance analysis — that is, it is combinations of intensity values rather than individual intensities which are important. The alphabetic characters seem more susceptible to analysis of the distribution means, due to centering of the patterns. In the experiments on alphabetic characters both mean analysis and covariance analysis were used. On the TIROS frames only covariance analysis was used.

Normal populations which differ in their covariance matrices give rise to quadratic input property filters. These are very expensive in analog hardware, and require very large amounts of memory in digital implementations. A parallel hyperplane approximation was developed. Experiments on the alphabetic characters indicate that with this approximation the increase in the number of property filters to separate the sample patterns is not great, considerably reducing the overall system complexity. Implementation of the gray invariant version of this approximation in digital equipment is simple, but the analog form suffers from the same problems as the quadratic unit, on a more limited scale. The gray invariant approximation was used in the TIROS frame experiments.

In the experiments on the alphabetic characters, eight methods for generating property filters were used. Four of these were designed to exploit differences in the means of the distributions, and four to exploit differences in the covariance matrices. Half of the techniques in each case produce property filters invariant to changes in the gray scale.

There was sufficient information in the sample patterns to permit each of these techniques to provide complete networks. The number of property filters required to separate the sample patterns generally ranged between eight and 15, depending on the particular technique.

The generalization performance of these networks was reasonable. For the techniques based on differences in the distribution means, there is little difference in performance, each technique averaging

about an 8% error rate. The best individual network had about a 6% error rate. The quadratic units gave a 10% error rate. The parallel hyperplane technique gave a 13% error rate and its gray invariant version yielded 15% errors. The best generalization performance generally occurred when the design was about 70% completed. For the gray invariant parallel hyperplane, best performance occurred at 40% completion, and again at completion. Although the performance degrades only about one per cent after the best level is reached, better levels might have been achieved if the coupling did not become as great as it does toward the end of the network design.

The utility of the gray invariance was demonstrated by modifying the gray scale of the sample patterns. A linear change has no effect on the performance of the gray invariant networks while having profound effects on the noninvariant ones. Even for some nonlinear changes in the scale, the effect on the noninvariant networks is much greater than on the invariant ones.

These performances were achieved on one binary decision. The design and generalization sets of sample patterns consisted of 20 examples, each containing 12 letters. These were sloppy handprinted characters. To put the results in perspective, several other design techniques were tried. The best of these, in which all of the sample patterns are stored in memory, and the unknown patterns correlated against all of these to find the highest correlation, gave a 5% error rate. More than 17,000 memory locations are used to store the sample patterns. In other tests perceptrons with 50 randomly selected property filters, or 50 property filters designed by analysis of the distribution means, and forced learning and Bayes weights for decision function assignment gave 25 and 20% generalization errors for forced learning and Bayes weights, respectively. Using 14 property filters taken from a network designed by the current technique (one which analyzes distribution means), the forced learning approach gave 9% generalization errors, just one per cent more than the network from which the units were taken. The value of the selectivity and the coupling in the present technique is apparent.

Despite the shortcomings shown in the generalization data, and the hardware difficulties in implementation, the gray invariant version of the parallel hyperplane technique was used in the TIROS frame investigation. Unless a considerable amount of normalization can be achieved, methods based on differences in the distribution means will not be effective in photointerpretation. This leaves the quadratic units and the parallel hyperplane units. The parallel hyperplane units were selected, as they require only 22% as much memory for unit specification. The value of the gray invariance when the gray scale does change was deemed to be sufficient compensation for the slightly higher error rate on the black and white generalization patterns.

One thousand patterns were used as design patterns in the TIROS investigation. Five hundred of these contained vortices; 500 did not. An additional 200 patterns used as a generalization sample were also equally divided. This classification was accomplished by meteorologists with experience on TIROS photographs. Classification by Douglas personnel was less consistent. The 1200 frames were obtained by considering four rotations of each of 300 pictures. These pictures are 180-by-180, covering 9/16 the area of a TIROS frame. The editing was accomplished to remove as much of the horizon as possible. No attempt was made to balance the locations of the fiducial marks in the resultant patterns. Only 223 actual TIROS frames were used, the additional patterns being obtained by choosing more than one edit location on a frame in a number of cases. Great difficulties were encountered in obtaining even this many usable digitized patterns.

Optical studies were performed on some TIROS pictures to determine the minimum resolution requirements compatible with good recognition performance by people. Sixty different frames were utilized. Prints of these frames were made at various resolutions, using a screen process to control the resolutions. These were examined by 15 to 20 people, in order of increasing resolution, to determine the coarsest resolution level at which their classification became consistent. A frame

was nearly always called nonvortex when the resolution was too low. Thus, opinions almost never changed when the final decision was nonvortex. A remarkable number of frames resulted in split decisions, nearly half of the people calling them vortex and half nonvortex, even at the highest resolution level. For the vortex patterns the decision point seems to split evenly between the highest and lowest resolution levels (240 lines and 70 lines).

Seven networks were designed to separate the sample patterns. The first three of these were designed with 180-by-180 resolutions, with each property filter receiving ten connections selected randomly from the entire input field. A different starting random number provides distinct networks. The first two networks required 250 property filters, the third required 220 to separate the sample patterns. Thus the networks averaged 2400 input connections, or 8% of the available picture points. Since the resolution study did not indicate that the smaller vortices could be identified at 70 lines (giving 8% as many elements as 240 lines), one network alone will not be particularly general. If the design sample is sufficiently representative, but the design process stopped when the network was sufficient to classify the design sample patterns but not the generalization sample, then improved generalization should result if the networks are combined. Such combinations do not result in significant improvements in the generalization performance.

Each of the three networks can produce generalization error rates less than 40% at some stage in its design. Machine No. 1 is best in this respect, giving a minimum of 36.5% errors at one point, and maintaining 40% or less through most of the design. For machines 1 and 2 the error rate has a minimum at a point about half way through the design. Quite consistently, the error rate on the design patterns is 3-6%. In all cases the error rate deteriorates toward the end of the design. Combining the networks overcomes this tendency for the error rate to rise. The conclusion from these data is that the increase in coupling in the design of the property filters is undesirable.

The poor generalization performance is due to the nonrepresentativeness of the sample patterns. That 500 patterns, unnormalized for size, position, horizon location, or fiducial mark location, do not adequately represent the vortex or nonvortex classes on a 32,400 point input field is not surprising. The network which achieved perfect separation on nonrepresentative samples may have accomplished some combinations of two things. The network may be a good design for a smaller class of vortex and nonvortex patterns of which the design sample is representative, or it may be a design which capitalizes on the nonrepresentativeness of the extraneous features in the samples, such as the location of the horizon or fiducial marks. The extent of the combination achieved cannot be determined without a very detailed study of the design and generalization patterns. Some indication is given in the resolution results to be discussed next. That at least some honest properties were found is evidenced by the fact that generalization performance is consistently better than chance.

Networks 4 and 5 were designed on lower resolution patterns. These maintained the 180-by-180 format, but the gray level changed only every second or third point, respectively. Generalization for these networks was tested against full resolution patterns. The networks required 240 and 250 property filters, respectively. Each network has a best generalization error rate of 41%, near the beginning of the design for Network 4 and near the end for Network 5. The mid-design minimum is 87 errors in each case. The mid-design minimums for the first three networks were 75, 82, and 79 errors respectively. This significant deterioration compared with Networks 1, 2, and 3 is again indicative that some honest properties have been derived. The rapid improvement in generalization performance toward the end of the design is almost unique in the seven networks, and no explanation has been found for this event.

Networks 6 and 7 were designed with limitations on the input fields of the property filters. For Network 6 the connections for the filters

were drawn from 90-by-90 subfields positioned randomly in the main field. For Network 7 45-by-45 subfields were used. These networks required 280 and 290 property filters for complete design. The rate of design is not noticeably slower than for the other networks until the error rate on the design patterns is less than 2%. These networks do not show the deterioration of generalization performance toward the end of the design that the first three networks did. Instead the performance remains nearly constant over the final one-fourth of the design. The minimum generalization error rates are 41-1/2% and 43-1/2%, respectively, each minimum being achieved twice. Network 6 achieves its minimum at the three-quarter design point and at the end of the design; Network 7 achieves them at the beginning and at the one-third point. Therefore, there does not appear to be any advantage, and indeed there is some disadvantage, in seeking more localized properties.

The following conclusions are drawn:

(1) The design technique is capable of producing networks to separate the design sample of patterns. When the sample is representative, as with the alphabetic characters, good generalization results are achieved. In this case, the primary value of the process is in the set of property filters derived rather than the discriminant function. When the sample is nonrepresentative, as with the TIROS frames, only limited generalization success is achieved.

(2) The coupling between the design technique and the sample patterns is too tight. The increase in coupling in the property filter design should be removed or reduced, the parameter which controls the rate of design should be adjusted to give a lower rate, and perhaps the criteria for acceptance of property filters should be weakened.

(3) The size of the design sample required depends upon the minimum number of picture points needed for recognition and on the amount of normalization and "cleaning-up" of the patterns. For essentially unnormalized vortex patterns on a 32,400 point input field, 1000 sample patterns are inadequate to give a properly stratified (representative) sample.

(4) If the designs achieved are to be practical, considerably more attention must be given to normalization of the patterns. A size normalization would permit the use of a resolution level suitable to all patterns rather than just the smallest ones. The resultant decrease in the number of picture points required would make the sample more representative by editing or averaging out noise, and by limiting the number of effective translations of the patterns. Local and general normalizations of the contrast in the frame could eliminate the need for gray invariant property filters, resulting in a much simpler hardware implementation of the parallel hyperplane technique, and somewhat better performance. Position normalization (i. e., centering) could make one of the techniques based on distribution means practical, or at least would simplify the recognition task.

2.2 Recommendations

The recommendations reflect the conclusions of the preceding section. It is felt that more encouraging results could be obtained if the design sample could be made more representative. This can be accomplished by normalizing, editing, and resolution reduction. Further improvements can be achieved if the degree of coupling between the network design and the design patterns is reduced. Accordingly, it is recommended that:

(1) The file of sample patterns be examined and normalized. This should include severe cropping, centering, removal of fiducial marks, and some degree of contrast and brightness control. Edge enhancement might be considered. Resolution should be standardized to a minimum level for the resultant patterns, and probably should not be greater than 60-by-60.

(2) The file of patterns should be expanded by considering additional small translations of the centered patterns. Additional TIROS frames might also be used.

(3) The design of the property filters should be decoupled from the loss function. The rate of design should be slowed by adjustment

of the controlling parameter. If sufficient normalization of contrast and brightness is applied to the patterns, a noninvariant property filter technique would be appropriate. The effect of selecting the second best block of units rather than the best should be tested to see if generalization performance improves.

(4) The possibility of using idealized patterns (i. e., stylized, noise-free, simplified black and white patterns) for the design sample should be investigated. A gray invariant technique should be used.

(5) The possibility of "salting" the property filter set by supplying coherent sets of input connections (rather than random sets) to some of the candidates should be investigated. A set of points clustered around a spiral might, for example, produce a logic unit useful in vortex recognition.

(6) The above suggestions can be implemented using the existing computer program. With additional programming, different design techniques may be compared. The use of cluster analysis for the design of the property filters or for the design of multiple linear discriminants or both is a promising method. With the multiple discriminants, several of these discriminants are assigned to each pattern class. The decision function selects the highest discriminant. Thus the technique is similar to the correlation method which worked so well on the alphabetic characters, but uses the property filter structure and clustering to reduce the memory requirements.

It is felt that the above modifications would sufficiently improve the results obtained such that the possibility of classifying the vortices by type could be considered. This should be done, however, only after testing the above suggestions on the simpler problem of identifying vortices.

3. GENERAL DISCUSSION OF PATTERN RECOGNITION

3.1 Pattern Recognition Mechanisms

Pattern recognition is accomplished by comparing information derived from an input signal with similar data derived from known sample patterns (called paradigms). Based on these comparisons, a decision is made as to the nature of the input pattern.

The key to pattern recognition is invariance. It is desirable that the classification assigned to an object or pattern of interest in the field of view be independent, to one extent or another, of the position of that object in the field, the aspect at which it is viewed, the background against which it is seen, partial blocking of the object, individuality of design of the object, and changes in illumination. It is not too difficult to provide any one of these invariances. To provide all of the desired invariances with a practical amount of hardware, however, requires that the decision mechanism extract the essence of the objects to be identified.

The identification of objects may occasionally be accomplished by correlating the directly acquired data with stored paradigms. This procedure has no built in mechanisms for invariance. To obtain invariance, one must provide a multiplicity of sample patterns, or paradigms. As the problem of complexity increases, the memory requirements become enormous.

Modern techniques provide four phases in the design of pattern recognition networks, although individual methods may omit one or more of these phases. These phases are illustrated in the block diagram, Figure 1.

The purpose of signal conditioning, or "preprocessing," as it is called, is to provide a convenient input format, to provide invariance, to provide in many cases a reduction in the amount of input data, and to emphasize aspects of the input signal which are deemed important.

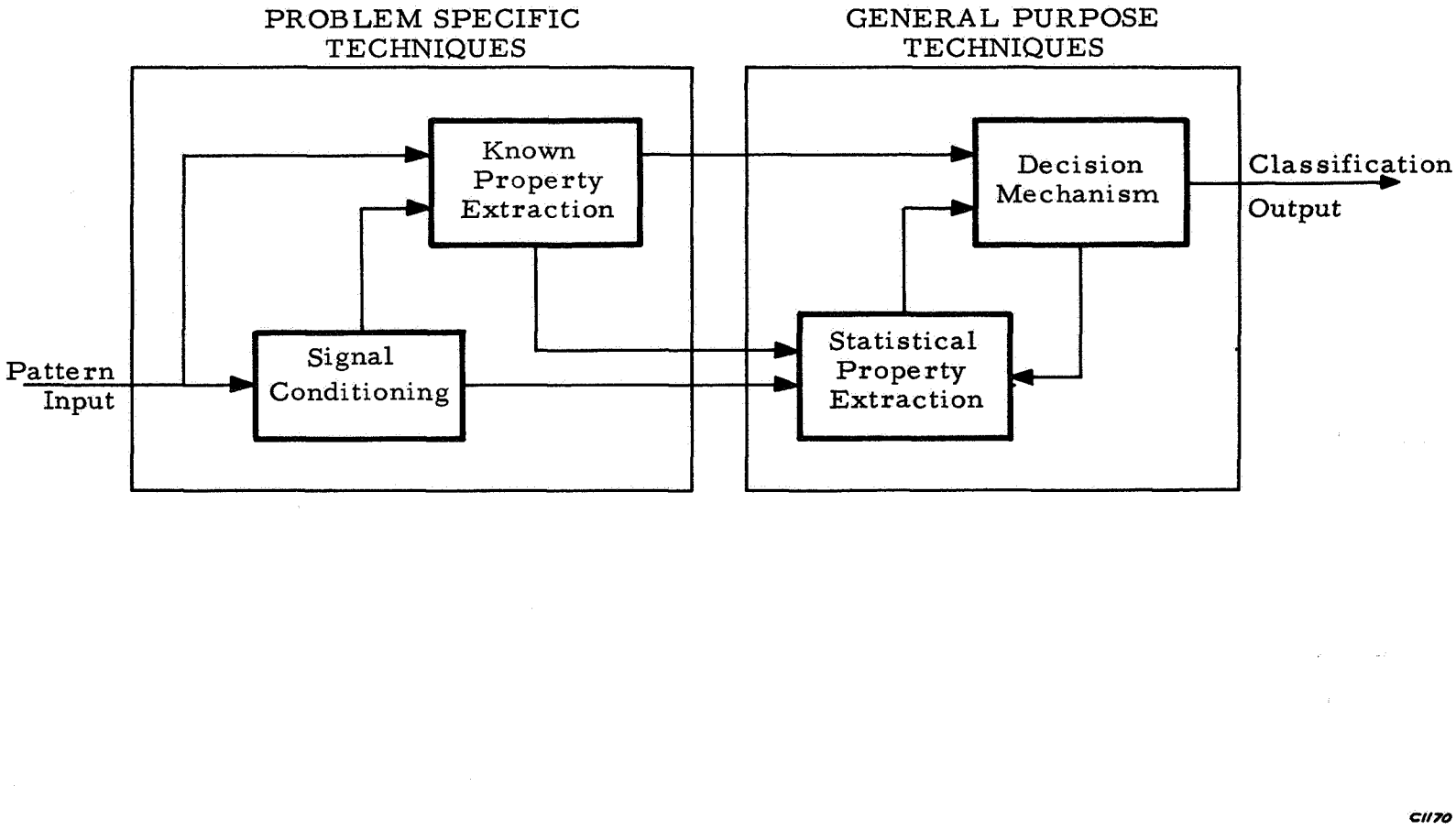


Figure 1. Pattern Recognition Mechanism

Preprocessing techniques include scanning, edge enhancement, Fourier transformation, autocorrelation and so on.

An almost universal approach to pattern recognition is to extract properties or features from the original signal, and to perform the recognition on the property profile of the input signal. This serves several functions. First, by reducing the input pattern to its essential features, the memory required for storing the paradigms is reduced to realistic levels. Secondly, by reducing the input pattern to independent features, a considerable amount of invariance to exact form is obtained. Finally, by extracting more than the minimum number of properties required, a degree of invariance to noise and background may be achieved, as the decision process then need not require a very high degree of matching between the input pattern and the paradigms.

Property extraction is accomplished in two phases. First, the designer constructs property detectors for those properties that he knows or suspects are important. These may prove to be inadequate, or may provide a format not suitable for the decision mechanism (many decision mechanisms provide only for linear discrimination). Statistical property extraction, in which a sample of classified patterns is analyzed, is used to augment the known property list and to re-format the property profile.

Most decision mechanisms are based on correlation. The distances of the property profile from stored paradigms is computed, and are used to provide an appropriate classification for the pattern. There are a variety of methods for obtaining the paradigms, and for combining the distances into a decision.

The following sections describe the four phases of the pattern recognition mechanism in greater detail.

3.2. Signal Conditioning

The basic data format of the investigation under question is determined by the characteristics of video systems. That is,

the data is assumed to be serial, analog data with resolution and bit rate limited by the television process. Various transformations of this video data are considered.

It is known that positional and rotational transformations of images of real objects seldom affect the proper categorization of the object. Consequently techniques that normalize the data to present a constant image, independent of such translations of the original data, lower the number of separate pattern classes that must be considered. Similarly the level of illumination, the contrast of the image, and, within limits, the relative size of the patterns of interest should not effect their categorization.

The proper classification of a pattern is often affected by the objects around it. Consequently, it is often desirable to suppress the background and fix attention on one feature of a video pattern to facilitate its classification.

Finally, it is often the case that the features defining visual patterns concern the sharp edges and the detail within the pattern. Such information is often highlighted with an edge enhanced data presentation.

Perhaps the most universal technique used to insure translation invariance is to scan a picture with a sample aperture. With this approach, only a small portion of the input signal is treated at one time. Decisions are made only on the portion of the input signal in the aperture. The entire signal is scanned with the aperture, producing a sequence of decisions. Individual decisions in this sequence are not influenced by other decisions in this sequence, and only rarely is any further machine processing done on the sequence of decisions. For this reason, the sample aperture is taken to be large enough to cover the entirety of patterns of interest in the scene. At the same time, it is desirable to have the pattern of interest as large as possible, with respect to the aperture, so that the figure-ground problem is simplified.

Scanning with an aperture can provide a considerable amount of invariance to translations. Varying the image to aperture size ratio can also satisfy part of size invariance and background suppression functions of the signal conditioning.

Various techniques for centering the image of a pattern of interest have also been investigated. Such techniques, however, are difficult to apply if the data contains images of many objects, of if the background cannot be entirely suppressed.

Optical preprocessing to produce the autocorrelation function has also been suggested to provide translation invariance and has been used extensively for such tasks as character recognition. It has been used with some success[2] on TIROS frames for classifying several cloud types.

Scanning in rotation can meet some of the rotation invariance requirements. This can be accomplished by rotating the sensor or image. In one technique[3] all of the scans obtained by rotation are considered simultaneously, however if a significant amount of information is to be extracted from each scan, individual processing seems to be more practical.

Overall brightness and contrast controls may also be introduced at this point. Brightness may be regulated by using the camera iris, or by controlling a bias added to the video signal. Contrast may be regulated by controlling the gain applied to the video signal. Since these controls regulate the overall brightness and contrast, other controls for the local areas (see Section 4.4) used in property extraction may also be desired.

There are processes that produce edge enhancement when this seems desirable. One such technique is to filter the picture, f(x, y), by applying the Laplacian operator, $\nabla^2 = \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{dx^2}$.

Since in regions where picture intensity is constant $\nabla^2 f$ is zero, the resulting picture consists of the fine detail present in the original. That is, the boudaries of highly contrasting regions are displayed against a dark background. This effect may be enhanced by a thresholding operation.

Laplacian filtering may be approximated by giving a small region of the field of view a large positive weight while weighting its nearest neighboring regions negatively to balance the positive weight. The filtered brightness at the small element under consideration is a weighted average of brightness at the element under consideration and its neighbors.

A similar technique to Laplacian filtering is gradient filtering, which involves the averaging of the square of the first derivative of the output of the sensor. Again, the result may be enhanced by a thresholding operation.

Contouring can also be accomplished by combining several scans. If the centers of the two fields of view are close to each other (but not coincident), by subtracting the two video signals some of the edges will be picked up and the regions of uniform intensity suppressed. Those edges which are parallel to the line joining the center of the two fields of view will be lost. Repeating this process by moving one of the fields so that the relative motion is along a line perpendicular to the line joining the two centers in the first instance, and again subtracting the outputs of the two scans, the remaining edges are picked up. The sum of the two difference signals yields a result similar to that achieved with an outlining or contouring operation.

Many of the property extraction and identification techniques in use at the current time use parallel logic and do not adapt to a serial data presentation. Consequently parallel pictorial data presentations are often useful. Typically such devices utilize photosensor matrices in conjunction with optical systems or in conjunction with a television receiver.

Similarly many techniques use a computer store for pictorial data. Such systems may directly store digitized television data, or may use (programmed) flying spot scanners to acquire data from photographic or optical images.

Finally, the input signal may be a time-frequency-amplitude domain rather than in a time-amplitude form typical of video data. Fourier transforms, vocoders, filter banks and prolate spheroidal transforms all are typical of the techniques used to achieve such representations. It may be noted, that while such presentations are almost universally used in audio data, they are relatively rare with video pattern recognition devices.

3.3. Known Property Extraction

Property extraction is the second phase of signal processing in a pattern recognition system. Techniques for defining properties which carry significant information may be divided into human design techniques and automatic design techniques.

Unfortunately there are no general techniques for the design of property extractions which utilize the designer's a priori knowledge of the recognition problem. Each case depends upon the specific problem and the knowledge and ingenuity of the designer. The Litton system[4] uses special scans to derive local texture properties (e.g. long-line content, high frequency content) which provide a property profile. Michigan University, Cornell Aeronautical Laboratory and many others use spatial filtering to extract a very limited number of special properties. The most common mechanization of extracting a property is by correlating the input signal against a prototype pattern by means of a linear input threshold unit.

The importance of such design, however, cannot be overestimated. By utilizing the considerable experience in pattern recognition of the designer, enormous savings in the amount of statistical

analysis performed in the automatic design phase can be effected. However, because the designer's knowledge and ingenuity is being tapped in some depth, automatic procedures for accomplishing this do not exist.

3.4. Statistical Property Extraction

The last step prior to the design of the decision mechanism, is the extraction of additional properties to facilitate the decision making process. These properties may be extracted from the data directly, from other properties, or from a combination of these. Their purpose is to augment the property profile descriptions with additional properties to facilitate pattern class separation by the limited structures which can be derived in the identification portion of a pattern recognition program. The techniques for extracting these additional properties may be classified by two criteria — the means for choosing the subset of data from which a property is extracted, and the means by which the function of this data representing the property is chosen.

Occasionally, as in SRI's Minos systems and Gamba's Pappa, properties are extracted by correlating a mask or prototype against the entire input pattern. More often, however, the input data are sampled and only a small fraction is correlated against a prototype. There are a variety of methods for selecting the particular subset of data used for a given property detector, as well as a variety of approaches to the specification of the prototype pattern.

In the original perceptron work, properties were generated randomly. Each property detector was a linear input threshold unit with a fixed number of input connections. The source points for the connection were selected randomly, as were the values of the prototype against which the input signal is correlated. The result of this method is completely random property detectors. As might be expected, only a small fraction of the properties so derived prove useful. Systems employing this method depend upon having many more property detectors than actually needed for the task, and an effective identification technique

for properly evaluating and utlizing their information. In one variation of this approach[5] the evaluations of the information of properties determined in the identification unit design were used to select only the most useful of randomly generated properties. These selected property detectors were the only ones incorporated in the final network.

Randomly selected sources for input connections are still widely used; most often, however, geometric constraints are placed on the selection of the sources. A point in the input field is selected as a center, and random connections are drawn from points constrained to be in a local area about this center. In this way, the input connections for a property detector all come from an area in which the patterns are likely to be coherent. A nonrandom system with geometric constraints for choosing input connection has been used by Philco[6]. Eight by eight regions of the input pattern provide source points for the property detectors. These eight by eight regions overlap and cover the input field.

A statistical technique called discriminant analysis[7,8] has been applied to the generation of properties.[9,10,11] In the use of this technique, distributional assumptions on the input patterns are required. Most often, a multivariate normal distribution is assumed for each class, and although this assumption is not often met, the resultant units are still considerably better than randomly derived prototypes. If one assumes that the covariance matrices are equal for the pattern classes, the discriminant analysis results in a linear input threshold unit for property detectors. Unfortunately, this assumption is not too suitable for processing visual input data (the design of a decision network for classifying sample patterns is not much affected, but the generalization capabilities of the resultant network are). Quadratic input threshold units result when this assumption is not made. These are difficult to implement, and a very effective approximation to this unit has been devised (see Section 4.5) using a linear input unit with two thresholds. A discriminant analysis design technique utilizing linear input units with

multiple thresholds, has also been reported.[12] Modifications to the discriminant analysis (see Section 4.4) which result in property detectors that are invariant to changes in brightness and contrast in the input data are also available.

Since one of the functions of the statistically derived property detectors is to reformat the input information so that it will be most readily identified, a technique has been developed at Douglas[9] in which the set of statistical property detectors and the decision structure utilizing their information are developed simultaneously. A set of statistical property detectors is designed by using discriminant analysis on randomly selected input connection sets. A minimum loss technique is used to design decision logic for these property detectors. Most of the property detectors are discarded, based on the evaluation supplied by the minimum loss technique. The loss numbers assigned to the individual sample patterns reflect how well the remaining structure classifies these patterns. These loss numbers are used to weight the contributions of the sample patterns to the statistical information needed to design a second set of property detectors. By so weighting the patterns, one essentially factors out that portion of the problem which has already been solved. Again, most of these additional units are discarded, based on the evaluation of their worth as additions to the existing structure. The process is continued until the problem of classifying the sample patterns is solved. This technique is described more fully in Section 4.

In a variation of this procedure, a minimum loss technique can be used to design the property detectors in exactly the same way as the decision network is designed. Input connections for the logic units are chosen sequentially by the selection process (from a geometrically constrained set) on the basis of which ones provide the most improvement to the existing partial set.

The use of a loss function in the last two approaches and in particular an exponential loss function, does not have a rigorous

foundation. It has been found, however, that the combination is very effective in designing networks to separate the sample patterns. The entropy function of information theory would provide a natural medium for evaluating the utility of input connections for property detectors or of the property detectors for the decision structure, since it is based on the uncertainty of the decision process.

For the systems in which masks are correlated against the entire input image (Minos, Pappa), the masks thus far have been designed by intuition. Discriminant analysis for unequal covariance matrices requires the inversion of matrices the size of the number of resolution points. When the entire image is used, this computation becomes unwieldy. The choice of parallel lines, checkerboards, etc. is, in reality, a form of random selection of properties, as these forms are neither statistically derived from sample patterns, nor derived from the designers knowledge of the specific problem.

3.5 Pattern Identification

Nearly all current pattern recognition decision mechanisms essentially involve correlating the property profile derived from the input pattern against one or more prototype patterns. Correlation schemes differ in the number of prototypes utilized, and the means for specifying these prototype patterns. In the original perceptron work,[13] one prototype per pattern class was used. A decision was achieved by determining which prototype correlated most highly with the property profile of the input pattern.

When n (more than two) pattern classes are of interest, attempts to provide economy have led to the use of $\log_2 n$ independent binary classifications of the input patterns. While some of these attempts met with striking success,* the consensus today is

*The Astropower Decision Filter[5] had two binary output units to resolve four pattern classes.

that a single prototype each is not adequate to represent the rather arbitrary groupings of patterns which result from such a process. Current practice provides at least one prototype for each pattern class.

When more than one prototype per pattern class is to be used, one must specify how many prototypes should be assigned to each class, and how the classification is to be made, based on the multiplicity of correlations obtained. A simple averaging, over all of the prototypes representing a class, of the correlations obtained is the equivalent of providing a single average prototype for each pattern class.

Widrow's Madaline[14] pointed the way to one system. In this system, the individual correlations are thresholded. That pattern class with the largest number of correlations exceeding a threshold is selected. In the special case when two pattern classes are involved, this system may be implemented by substituting the negatives of the prototype of one of the classes, and using a simple majority gate for obtaining a decision. This special case constitutes the Madaline system.

A second method utilizing multiple prototypes which is gaining favor is to assign to an input pattern the classification of that prototype with which it has the highest individual correlation. This is the system which has been adopted by Litton.[4] Stanford Research Institute[15] has also begun experimenting with this method, and reports results superior to those achieved with the Madaline system.

The number of prototypes to be used provide another divergence in approaches. In Litton's technique the property profile of every sample pattern is saved. This approach should provide the highest accuracy, assuming that a sufficient number of representative samples are obtained, and that the memory facilities can be provided. SRI favors a more limited number of prototypes. In an experiment using two prototypes per pattern class, good results were obtained even when the (artificially generated) property profiles had four and more cluster

points per class. It is possible that this last result could be due to the actual juxtaposition of the selected cluster points, and may not be true in general.

An area receiving an enormous amount of attention in pattern recognition research is that of specifying the prototype patterns. The technique first used in the development of perceptron theory has already been mentioned above — the single paradigm for each class was taken to be the average (or sum) of the property profiles of all of the sample patterns in that class. This technique was given the name "Forced Learning." (The technique is optimal if pattern classes in the property profile space have multivariate normal distributions with the same covariance matrices).

If one assumes that the property filters are independent, that is, that there is no redundancy in the property list, then a likelihood ratio test can be used to provide the prototype patterns. For a given pattern class, the i-th coordinate of the prototype vector is given by $\ln \frac{r_i}{1-r_i}$, where r_i is the rate at which the i-th property occurs in patterns of that class. This technique has been given many names, among them "log weights," "Bayes weights," "discriminant analysis," "optimum weights." It has been suggested by Gamba, Joseph, Kanal, Maron and others.* Its proponents recognize that the underlying independence assumption is not valid, but suggest that this results in only a small loss in efficiency.[16]

Perceptron research[18] also produced the famous Error Correction technique for specifying paradigms. The prototypes obtained are weighted averages of the sample property profiles. The manner in which the weightings for the average are derived iteratively

*Mosteller and Wallace[16] suggested the application of this technique to resolving the disputed authorship of some of the Federalist Papers, and recently[17] have claimed success in this application.

distinguishes this method. A given set of prototypes is tested against the sample patterns. Each time an erroneous classification occurs, the sample property profile in question is added to the appropriate prototype. This process is continued until no errors occur, or until the experimenter gives up. The enormous popularity of this method stems from the Error Correction Theorem, which states that if there is any way of choosing a single prototype per class to give perfect performance on the sample patterns, the error correction procedure will produce such a set of prototypes. This technique is currently the most commonly used one in pattern recognition. Indeed, for a while, the largest effort in pattern recognition research consisted of finding new proofs for the Error Correction Theorem.[19]

The search for a method to produce better generalization results led Highleyman[20] to suggest a minimum loss approach.* In this technique each sample pattern is assigned a loss number. This loss number reflects how well or how poorly a pattern is classified. The values of the correlations with the prototype are compared. Properly classified patterns for which the difference in the discriminants (correlation values) is large are given small loss numbers. When the difference in the discriminants is small, a larger loss is assigned, and if the pattern is incorrectly classified, still larger losses are assigned. Highleyman suggested a cumulative normal function for obtaining loss figures from the difference in discriminant values. The prototypes are adjusted to provide a minimum for the sum of all the loss numbers assigned to the sample patterns. The Astropower Laboratory has used this approach extensively, utilizing an exponential function rather than a

*Recently it has been shown that error correction procedures may also be regarded as loss minimization techniques. See Mays, C. H., "The Relationship of Algorithms Used with Adjustable Threshold Elements to Differential Equations," IEEE Trans. on Elect. Computers, Vol. EC 14 #14, Feb. 1965, pp. 62-63.

cumulative normal function. W. Bledsoe* and General Electric* in unpublished efforts have also used a minimum loss approach with exponential loss functions. "Optimum" prototypes cannot be found directly, and an iterative technique, in which the coordinates of the prototype vectors are adjusted one at a time to give minimum loss has been suggested. It has been proven that if prototypes to give perfect performance exists, this iterative minimum loss technique will produce such a set of prototypes; and if perfect performance on the sample patterns is not possible, a set of prototypes yielding minimum loss will be approached.

The techniques for specifying prototypes all originated in systems employing a single prototype per pattern class. The methods for specifying several prototypes per class are not as well understood.

Multiple prototypes are useful when a single prototype cannot adequately represent all of the patterns in a class. This can occur when the distributions of property profile vectors are multimodal, or when the distances between vectors within the same pattern class grow large compared to the distances between vectors in different classes. When several prototypes are used per class, each prototype may be used to represent a portion or a cluster of the distributions for that class.

Widrow provided an error correction scheme in his Madaline. Madaline is a two class machine utilizing a fixed number of prototypes. The distances of an input vector from each prototype are thresholded and a decision reached by a majority gate. If an incorrect decision is made, the property vector in question is added (or subtracted) to that prototype contributing to the wrong decision which is nearest the threshold. The Madaline system does not use an individual prototype to represent a cluster of patterns. The use of the majority gate complicates the action of this system.

SRI suggests an error correction procedure in which the prototypes are treated individually. A fixed number of prototypes are

---

*Personal communication

assigned to each pattern class. A decision is reached by determining which prototype is closest to the input pattern's property profile. If an error is made the property profile is (1) subtracted from the prototype having the highest correlation, and (2) added to the prototype assigned to the proper class which has the highest correlation.

An extreme method for selecting prototypes is the Litton system, in which the property profile of each sample pattern is stored as a prototype. If the number of sample patterns is sufficiently large, one would expect all portions of the distributions to be represented. If one is extremely skillful or extremely lucky in selecting properties, a small number of sample patterns may prove adequate. Otherwise, effective performance is obtained by increasing the number of sample patterns (and hence the memory and computing time requirements).

4. DESIGN CONSIDERATIONS

4.1 General Discussion

The design techniques utilized lead to decision networks having the logical structure shown in Figure 2, a structure common to most current pattern recognition devices. The logic units detect pattern features. Each logic unit accepts as inputs a very small fraction of the data in the pattern, determines whether or not these data possess a given property, and produces a binary output accordingly. The binary outputs of the property detectors, or logic units, are then combined by one or more response units to produce binary decisions about the nature of the entire input pattern. Customarily, a linear discriminant function is used by each response unit.

The techniques to be described provide effective means for designing property filters and for developing the linear discriminant function for a response unit. The techniques differ only in the method for generating property filters. That portion of the design philosophy common to these techniques has been named "Iterative Design."

The recognition logic is designed sequentially. Logic units are designed using discriminant analysis. As logic units are added to the design, the pattern losses indicate which of the sample patterns require the most attention. In acquiring the data required to generate additional logic units, the loss assigned to a pattern is used to weight the contribution of that pattern to the group statistics. This insures that the logic units thus derived are suitable for the part of the recognition problem as yet unsolved.

Logic units are generated by selecting a subspace of the original signal space, and performing a discriminant analysis in that subspace. Subspaces are used since it is usually not practical to perform the discriminant analysis in spaces with high dimensionality, nor is the implementation in analog hardware of the resulting logic unit feasible.

No technique has yet been found for optimally selecting subspaces, so that an element of randomness is used.

Randomly chosen subspaces are not all of equal utility, and often a particular subspace will prove to be of no value at all. For this reason, selection is used. Each time the recognition network is to be expanded by the addition of logic units, more units are generated than are to be added. Only the best of these units are selected. The selection criterion is — which unit would result in the greatest decrease in the system loss.

4.2 Iterative Design

Iterative design is an approach to developing the decision structure of Figure 2 gradually, using information derived from a partially designed structure to highlight difficult problem areas. This is accomplished with the aid of a loss function. Each sample pattern is assigned a loss number that depends upon how strongly it is classified by a partially designed machine. The system loss is taken as the sum of the losses of all sample patterns. Each change made in the system reduces the system loss.

Let:

$$\alpha_j^i = \begin{cases} 1 & \text{if the i-th sample pattern activates the j-th logic} \\ 0 & \text{otherwise} \end{cases}$$

let W_j be the weight assigned to the j-th logic unit, and let θ be the threshold of the response unit.

β_i is used to denote the linear discriminant, or input to the response unit for the i-th pattern.

$$\beta_i = \sum_j \alpha_j^i W_j$$

The network classifies the i-th pattern as a "positive" pattern if the discriminant exceeds the response unit threshold, that is, if

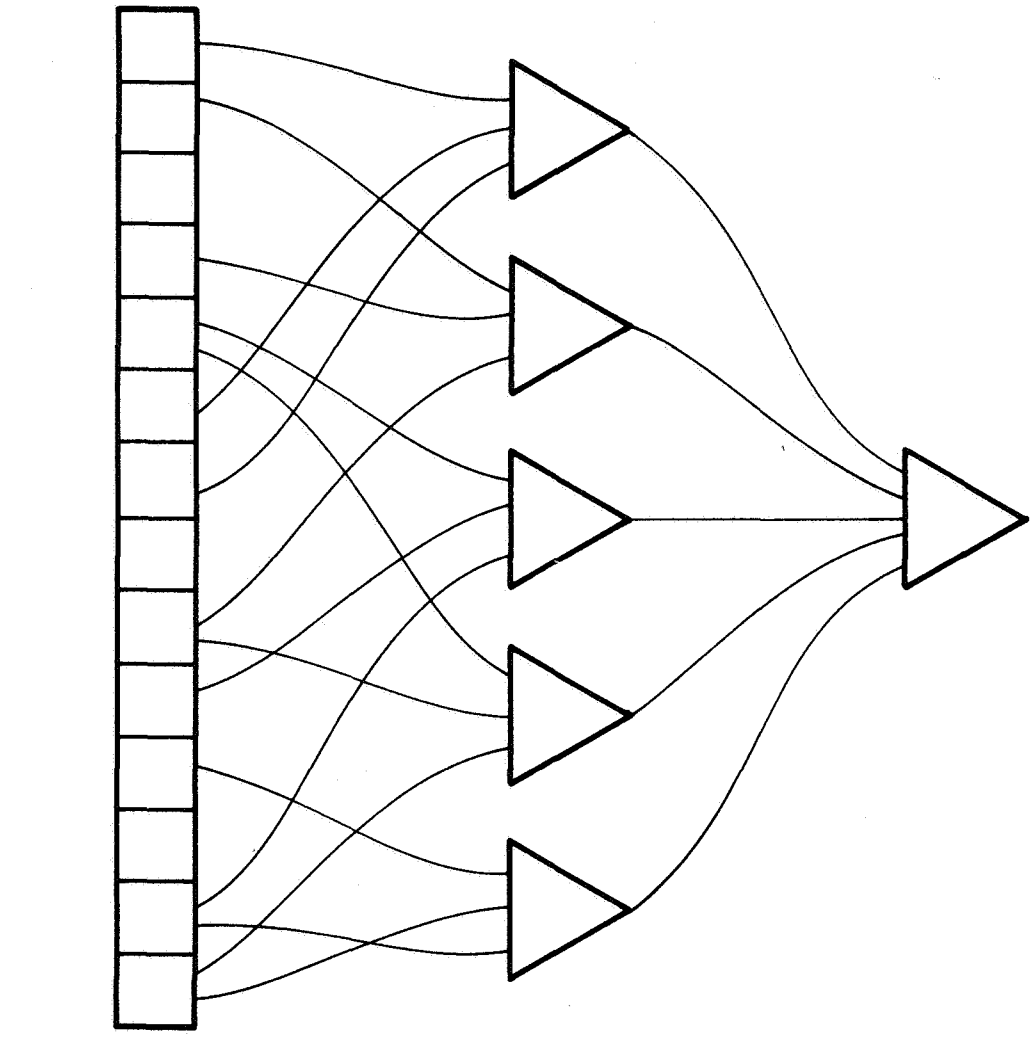


Figure 2. Decision Network Structure

$$\beta_i - \theta > 0$$

and as a negative pattern if the discriminant is less than the threshold. The symbol δ_i is used to indicate the true classification of the i-th sample pattern.

$$\delta_i = \begin{cases} 1 & \text{if the i-th pattern is a "positive" pattern} \\ -1 & \text{if the i-th pattern is a "negative" pattern} \end{cases}$$

The loss function under investigation has an exponential form. The loss assigned to the i-th pattern is

$$L_i = e^{\delta_i(\theta-\beta_i)}$$

This form has many desirable features, but does not permit the analytic optimization of the linear discriminant function. A relaxation technique is substituted.

Suppose that N - 1 logic units have been incorporated in the design, and a discriminant function established. Suppose that a population of logic units that are candidates for inclusion in the network is available. (Section 4. 3 describes techniques for generating such populations.) The iterative design algorithm calls for the addition to the network of that candidate unit which results in the lowest system loss. Because of computational difficulties with the exponential loss function, the present technique modifies this to the addition of the unit which results in the lowest system loss when W_1 through W_{N-1} are held fixed (that is, only W_N and θ may be adjusted). This compromise results in great computational simplicity.

Denote by I_+ the set of indices, i, for which $\delta_i = 1$, and by I_- the set of indices for which $\delta_i = -1$. For a given candidate for the N-th logic unit, let I_N be the set of indices i for which $\alpha_N^i = 1$ and by $\overline{I}_N$ those for which $\alpha_N^i = 0$: Thus the symbol $\sum_{i \epsilon I_+ \cap I_n}$ would mean "the summation over all values of i for which the i-th pattern is 'positive' and activates the N-th unit. "

The system loss, if the candidate unit is added, and the weights $W_1, \ldots, W_{N-1}$ are held fixed, is

$$L = 2\sqrt{\left(\sum_{i\epsilon I_+ \cap I_N} L_i\right)\left(\sum_{i\epsilon I_- \cap I_N} L_i\right)} + 2\sqrt{\left(\sum_{i\epsilon I_+ \cap \bar{I}_N} L_i\right)\left(\sum_{i\epsilon I_- \cap \bar{I}_N} L_i\right)},$$

the new value of the threshold is given by

$$\bar{\theta} = \frac{1}{2} \ln \frac{\sum_{i\epsilon I_- \cap \bar{I}_N} L_i}{\sum_{i\epsilon I_+ \cap \bar{I}_N} L_i},$$

and the weight of the candidate unit by

$$W_N = \bar{\theta} - \frac{1}{2} \ln \frac{\sum_{i\epsilon I_- \cap I_N} L_i}{\sum_{i\epsilon I_+ \cap I_N} L_i}$$

Once the N-th unit for the machine has been selected, the coefficients of the linear discriminant are readjusted, and the pattern losses recomputed. With the exponential loss function, this is also accomplished iteratively — each weight, W_j, being adjusted in turn. Several adjustments for each weight each time a unit is added seem to be adequate.

Despite the computational difficulties it engenders, the exponential loss function offers some desirable features. It permits the approximate calculations described above to be accomplished with great ease. It has an exponential rate of change, insuring that most attention is given to those patterns which are most poorly classified. It does not become constant, insuring that the decision boundary will be placed as far as possible from correctly classified sample patterns.

4.3 Discriminant Analysis

This section describes a number of techniques for generating populations of logic units as candidates for inclusion in the decision network. These techniques use the loss numbers assigned to the sample patterns by the iterative design process to weight the importance of these patterns — the populations of units thus generated tend to emphasize the remaining problem difficulties.

Each of the techniques involves strong distributional assumptions concerning the pattern classes. The resultant logic units can not, therefore, be characterized as having been derived by a distribution-free method. The effects of these assumptions on the network design are softened by the iterative design algorithm, which selects, on the basis of performance, only a small fraction of the units generated.

Discriminant analyses are used to generate a logic unit, given a subspace. To perform the discriminant analysis, assumptions concerning the distributions of patterns in the subspace must be made. These assumptions, in general, are not justified. To some extent these assumptions are important; in other cases, they are not.

In the cases investigated, two pattern classes were considered. The two pattern classes are assumed to have multivariate normal distributions. The statistical data which must be extracted from the patterns is thus limited to the first two moments but this does not seem to be a serious shortcoming for this problem. Experience with discriminant analysis with a normality assumption has shown it to work well even when the distributions are not normal, provided, of course, that the pattern classes may be separated on the first two moments of their distributions. Further, the selection and iterative design procedures evaluate and weight logic units according to their actual performance on the sample patterns rather than the performance predicted for them by the discriminant analysis.

The assumptions which are made concerning the mean vectors and the covariance matrices of the two distributions seem more important. These assumptions control the nature of the differences between the distributions which are to be exploited. They also control the nature of the resulting logic units.

An assumption, which results in easily implemented logic units, is that the covariance matrices are equal. The pattern class distributions differ only in their mean vectors. Under this assumption the optimum decision boundary for a logic unit is a hyperplane — one which bisects the line segment joining the mean points of the two pattern classes. The estimate of the common covariance matrix is used to establish the orientation of this hyperplane. The resulting logic unit is the linear input threshold unit almost universally used in perceptron work. In the following material, this technique is designated "Oriented Hyperplane." Figure 3a illustrates the oriented hyperplane in two dimensions (in this case the hyperplane is a line).

The calculation can be simplified by ignoring the orientation of the hyperplane. The logic unit boundary is the hyperplane which bisects the line segment joining the class means and which is perpendicular to that line segment. This approach eliminated the need for estimating and inverting the covariance matrix. The same logic unit would result if it were assumed that the common covariance matrix was a multiple of the identity matrix. This technique is designated "Perpendicular Bisector" and is illustrated in Figure 3b.

The assumption of the preceding techniques — that the pattern class distributions differ in their mean vectors — does not fit the cloud pattern recognition problem. It can be shown that if patterns are to be recognized in all rotations and translations, then the mean vector for a pattern class will have all components equal. These components will equal the average intensity of all patterns of that class. It would appear that the mean vectors should be assumed equal, and the

common mean vector should have equal components. The pattern classes are differentiated by their covariance matrices.

If it is assumed the covariance matrices are different, then the decision boundary becomes a quadratic surface. The assumption concerning the mean vectors does not alter the general form of the solution, just its computation. The technique which assumes the mean vectors equal, and the common mean vector to have equal components is called "Quadratic Surface." It is illustrated in Figure 3c.

Quadratic units are difficult to implement in hardware, and require large amounts of memory in a digital simulation. An approximation is possible in which the principle axis of differentiation is computed. A pair of parallel hyperplanes perpendicular to this axis is substituted for the quadratic surface. The resulting logic unit is easily implemented. This technique is designated "Parallel Hyperplane" and is illustrated in Figure 3d.

TABLE I

TECHNIQUE COMPARISON

Technique	Difference Exploited	Remarks
Oriented Hyperplane	Mean Vector	Linear Input Threshold Unit
Perpendicular Bisector	Mean Vector	Same - Easier Computation
Quadratic Surface	Covariance Matrix	Quadratic Input Threshold Unit
Parallel Hyperplanes	Covariance Matrix	Linear Input - Double Threshold Unit

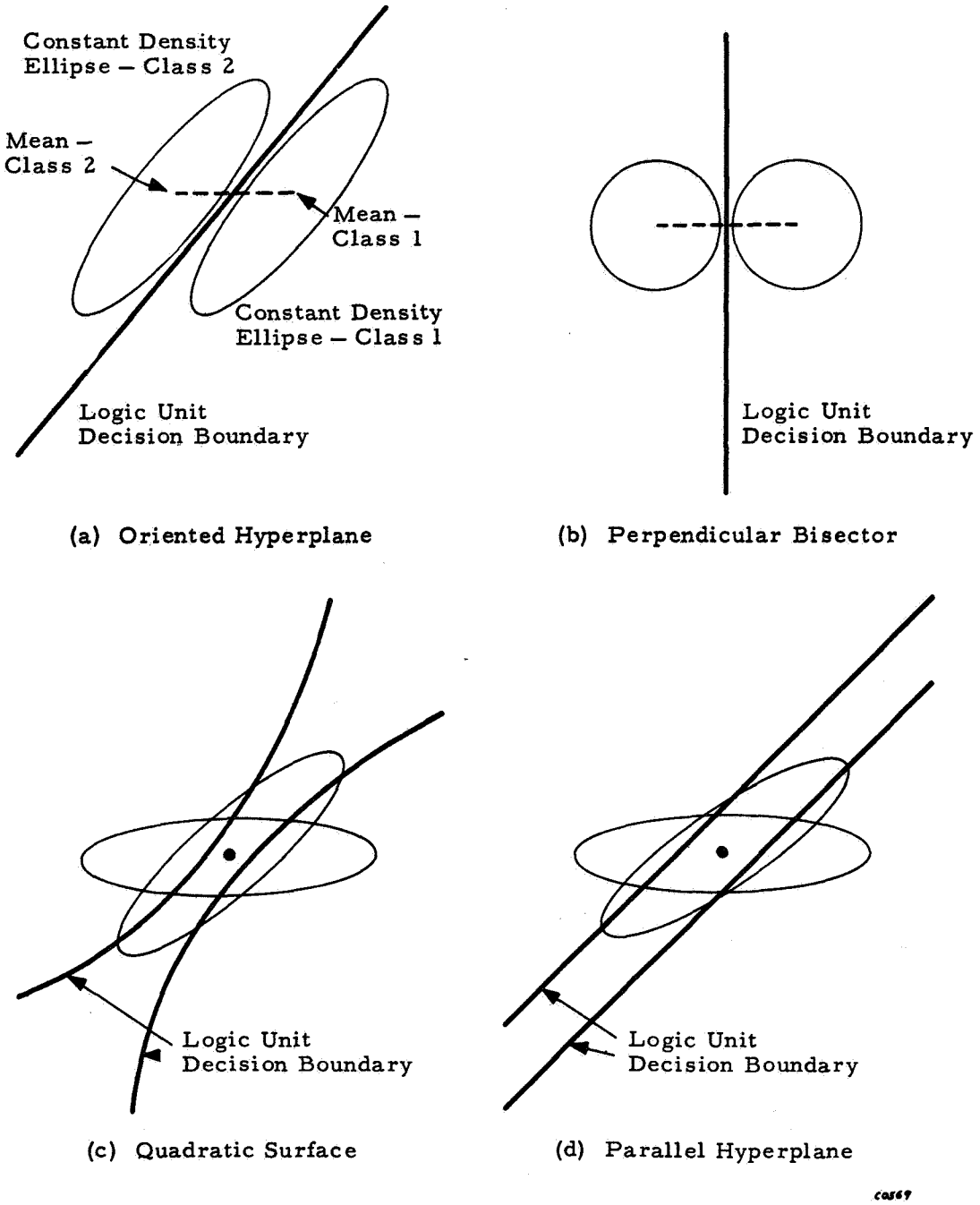


Figure 3. Discriminant Functions

The four techniques share the assumption that, over each pattern class, the measurement vectors have a multivariate normal distribution. They differ in which parameters are left unspecified.

The n-variate normal density function, in matrix form, is

$$\frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(X-\mu)^t \Sigma^{-1}(X-\mu)\right\} \qquad (1)$$

where μ is the (column) mean vector, Σ is the covariance matrix, and $|\Sigma|$ is the determinant of the covariance matrix. We will denote by a subscript "+" those parameters which apply to the distribution of the class of positive patterns, by a subscript "-" those pertaining to the distribution of the negative patterns, and for those parameters which apply to both distributions, no subscript will be used.

The Neyman-Pearson Lemma specifies the likelihood ratio test as the optimum decision rule in the subspace. A pattern with measurement vector X turns the unit on if

$$\frac{\frac{1}{(2\pi)^{n/2}|\Sigma_+|^{1/2}} \exp\left\{-\frac{1}{2}(X-\mu_+)^t \Sigma_+^{-1}(X-\mu_+)\right\}}{\frac{1}{(2\pi)^{n/2}|\Sigma_-|^{1/2}} \exp\left\{-\frac{1}{2}(X-\mu_-)^t \Sigma_-^{-1}(X-\mu_-)\right\}} > \lambda_1 \qquad (2)$$

By taking logarithms this inequality is seen to be equivalent to

$$(X-\mu_+)^t \Sigma_+^{-1}(X-\mu_+) - (X-\mu_-)^t \Sigma_-^{-1}(X-\mu_-) < \lambda_2 = -\ln\lambda_1^2 + \ln\frac{|\Sigma_-|}{|\Sigma_+|} \qquad (3)$$

An alternative form is given by

$$X^t(\Sigma_+^{-1} - \Sigma_-^{-1})X - 2X^t(\Sigma_+^{-1}\mu_+ - \Sigma_-^{-1}\mu_-) < \lambda_3 = \lambda_2 - \mu_+^t \Sigma_+^{-1}\mu_+ + \mu_-^t \Sigma_-^{-1}\mu_- \qquad (4)$$

The constant λ_1 (and hence λ_2 and λ_3) is chosen to reflect differences in the a priori probabilities of the pattern classes, and differences in the costs associated with the two types of errors. The discriminant function,

$$D(X) = X^t(\Sigma_+^{-1} - \Sigma_-^{-1})\, X - 2X^t(\Sigma_+^{-1}\, \mu_+ - \Sigma_-^{-1}\, \mu_-) - \lambda_3 \qquad (5)$$

is specified by estimating μ_+, μ_-, Σ_+, and Σ_- from the sample patterns. It is, of course, a quadratic function of the measurements taken from the pattern to be classified, and thus specifies a quadratic surface as the decision boundary for the logic unit.

The well known result given above is the general discriminant analysis solution for multivariate normal distributions. In the subsequent work, we will not be concerned with this most general form, but with the special cases which occur when certain restrictions are placed on the estimates of the mean vectors and covariance matrices. These restrictions are chosen to serve one or more of four purposes--to simplify the estimation process, to simplify the implementation of the discriminant function, to conform to actual constraints on the patterns, and to provide logic units invariant to changes of brightness or contrast in the input data.

4.3.1 Oriented Hyperplane

The first constraint to be considered is the assumption that the covariance matrices are equal, that is, that

$$\Sigma_+ = \Sigma_- \equiv \Sigma$$

While this assumption simplifies the estimation process since there is only one covariance matrix to estimate and invert, its primary purpose is to simplify the implementation of the discriminant function, which becomes

$$D(X) = -2X^t\, \Sigma^{-1}\, (\mu_+ - \mu_-) - \lambda_3 \qquad (6)$$

This decision is implementable by the linear input threshold device which is so popular in pattern recognition. A weight vector for such a unit is

given by

$$\Sigma^{-1} (\mu_+ - \mu_-) \tag{7}$$

and the threshold by

$$\ln \lambda_1 + \frac{1}{2} (\mu_+ + \mu_-)^t \Sigma^{-1} (\mu_+ - \mu_-) \tag{8}$$

In the empirical studies of Section 5, this technique of generating logic units is called the "Oriented Hyperplane" technique. In that section λ_1 is taken to be "1". The name stems from the fact that the decision boundary of a logic unit is a hyperplane which cuts (bisects for $\lambda_1 = 1$) the line segment joining the means μ_+ and μ_-, but in general has some orientation other than perpendicular with respect to that line segment.

4.3.2 Perpendicular Bisector

The second special case to be considered is a further restriction of the Oriented Hyperplane case given above. The additional assumption is that the common covariance matrix is a multiple of the identity matrix

$$\Sigma = k I$$

This assumption, that the measurements taken from the patterns to be classified are independent with equal variances, has the effect of greatly simplifying the estimation process. An implementation of the decision boundary with a linear input threshold device uses

$$c (\mu_+ - \mu_-) \tag{9}$$

for a weight vector, and has

$$c k \ln \lambda_1 + \frac{c}{2} (\mu_+^t \mu_+ - \mu_-^t \mu_-) \tag{10}$$

for a threshold. The constant "c" can, of course, be chosen arbitrarily. If λ_1 is taken to be "1", as is done in Section 5, then it is not necessary to estimate the parameter "k". If an equal number of positive and negative sample patterns are available, the weight vector can be obtained with adaptive hardware by adding the measurement vectors for positive patterns

to the existing weight vector and subtracting the measurement vector for negative patterns. A difference in the number of sample patterns could be accounted for by a gain adjustment for negative patterns. This particular adaptive procedure is called "Forced Learning" in the perceptron literature.[21]

This technique for generating logic units is called "Perpendicular Bisector." The name is derived from the fact that the decision boundary is a hyperplane perpendicular to the line segment joining the means, and bisecting it when $\lambda_1 = 1$.

4.3.3 Quadratic Surface

In the third special case to be considered, the covariance matrices are again allowed to be arbitrary, but the mean vectors are constrained. When the inputs to the logic units are taken to be the light intensities at various points of a picture, as is the case in the experiments of Section 6, then each component of the mean vectors is an average gray level for some point in the sampling aperture for one class of patterns. Consider a single picture translated and rotated with respect to the aperture. Its contribution to the components of the appropriate mean vector will be the average gray level of the whole picture, since when all translations and rotations occur, each picture point appears at each point in the sampling aperture with equal frequency. Any differences in the mean vectors, then, must be due to differences in the average gray levels of the classes. For the Tiros cloud photographs used in Section 6, the average brightness is controlled primarily by the amount of horizon in the picture, and the sun angle. There seems to be no reason to suspect that there is any difference in average brightness between frames incorporating tropical storms and frames of nonstorm cloud cover. If such a difference is found in the sample patterns, it is probably indicative of the incompleteness of the sample.

The special assumption made is that both pattern classes have the same mean vector, and that this common mean vector

has all of its components equal. Let "U" denote the n-vector, all of whose components are "1". The discriminant function (5) becomes

$$D(X) = (X - cU)^t (\Sigma_+^{-1} - \Sigma_-^{-1}) (X - cU) + \ln \lambda_1^2 + \ln \frac{|\Sigma_+|}{|\Sigma_-|} \qquad (11)$$

Let T be a transformation of maximal rank* such that $T\Sigma_- T^t = I$. Let $S = T\Sigma_+ T^t$. The inequality (11) can be written

$$D(X) = (X - cU)^t T^t (S^{-1} - I) T (X - cU) + \ln \lambda_1^2 + \ln|S| \qquad (12)$$

If the matrices Σ_- and Σ_+ are nonsingular, either form (11) or (12) may be used to define the logic unit. If, however, both matrices are singular, and in particular, if they both have the same null space, then only form (12) will do. This latter situation arises in connection with the modification made for gray invariance in Section 4.4.

In the experiments of Section 5 this technique of generating logic units is called "Quadratic Surface." Although the more general solution (5) also results in a quadratic surface, (5) is not used in the empirical studies of Sections 5 and 6. The quadratic surface unit is very expensive to implement in hardware (see Section 4.7.2) and requires considerably more memory than linear units in simulation programs. An approximation to the Quadratic Surface unit is derived in Section 4.5.

4.4 Gray Scale Invariance

The gray scale of TIROS frames varies from picture to picture, and the darker shades on one picture may actually be lighter than the brighter areas of another frame. Indeed, such variation can occur in different areas within a single frame. It seems desirable that an invariance to such changes in the gray scale be built into the logic of

*Such a transformation always exists, and is readily computed in terms of the eigenvalues and eigenvectors of Σ_-. T will be an m-by-n matrix, with m equal to the rank of Σ_-. The columns of T^t are the eigenvectors of Σ_- corresponding to nonzero eigenvalues, and having length squared equal to the reciprocal of the corresponding eigenvalue.

a decision network, rather than obtained by providing a multiplicity of hardware to operate at different gray scale levels. A single normalization of the gray scale of a frame does not provide the degree of invariance of the approach below.

For the Perpendicular Bisector and Oriented Hyperplane techniques, the logic of each unit can be made invariant to linear transformations of the gray scale by some constraints on the weights for the input connections. It is easily shown that for the sign of

$$\sum_{j=1}^{n} (aX_j+b) W_j - \theta$$

to be invariant for all values of "b" and for all positive "a", necessary and sufficient conditions are that

a) $\sum_j W_j = 0$

b) $\theta = 0$

The problem rests in implementing these restraints with as little disturbance as possible to the optimality of the solution.

Let U be an n-vector all of whose components are "1". The required invariances are achieved by making use of the presumed equivalences, that is, in an n-dimensional subspace, the vectors X and aX+bU are equivalent if "a" is positive.

In generating the weights for the input connections to a logic unit, each sample pattern X, as projected into the n-dimensional subspace, is modified to be

$$X - \left(\frac{X^t U}{n}\right) U$$

so that the sum of its components is zero. (This process will be called reduction.) This ensures that condition (a) is met. The class mean vectors μ_+ and μ_- are then computed from these reduced pattern vectors.

For the Perpendicular Bisector technique, condition (b) is satisfied if the mean vectors have equal length. This can be accomplished

by scaling μ_- to

$$\left(\frac{\mu_+^t \mu_+}{\mu_-^t \mu_-}\right)^{1/2} \mu_-$$

The weight vector for the logic unit is thus given by

$$\mu_+ - \left(\frac{\mu_+^t \mu_+}{\mu_-^t \mu_-}\right)^{1/2} \mu_-$$

It is not necessary to reduce each vector individually. If $\overline{X}_+$ is the average of X over positive patterns and $\overline{X}_-$ the average over negative patterns

$$\mu_+ = \overline{X}_+ - \left(\frac{\overline{X}_+^t U}{n}\right) U$$

$$\mu_- = \overline{X}_- - \left(\frac{\overline{X}_-^t U}{n}\right) U$$

A similar scaling of the reduced mean vectors works for the Oriented Hyperplane technique. The mean vectors are first mapped by a transformation T before scaling. (T, as in 4.3, is a transformation such that $T \not\Sigma T' = I$. The covariance matrix $\not\Sigma$ is computed from the reduced vectors and is thus singular, since each reduced vector has a zero component in the U direction. The matrix T is thus not square and maps X onto a subspace perpendicular to U.) The weight vector is thus given by

$$(\mu_-^t T^t T \mu_-)^{1/2} T^t T \mu_+ - (\mu_+^t T^t T \mu_+)^{1/2} T^t T \mu_-$$

For the quadratic surface technique, the measurement vectors are again reduced to provide invariance against a constant shift in the gray scale. Equation (12) of Section 4.3 must be used as both $\not\Sigma_+$ and $\not\Sigma_-$ have U for a null space. In addition, the reduced vectors for the positive patterns are scaled so that $|S| = 1/\lambda_1^2$. The scaling need not be applied to the individual patterns, as the estimate of S may be

adjusted. The discriminant function obtained is

$$D(X) = X^t T^t (\lambda_1^{2/n} |S|^{1/n} S^{-1} - I)\, T X \qquad (13)$$

Note that the mean vector no longer appears in the discriminant function since

$$TU = 0$$

This discriminant function is implementable with a quadratic input threshold device, with coefficient matrix proportional to

$$T^t (\lambda_1^{2/n} |S|^{1/n} S^{-1} - I)\, T$$

and a zero threshold. This unit is invariant to reversals as well as contrast and brightness, and works equally well for white-on-black and black-on-white patterns.

It is noteworthy that it is not necessary to reduce or scale the measurement vector of a pattern to be classified with any of the three gray invariant versions given in this section. Indeed, the implementation of the invariant versions is easier, particularly for the Quadratic Surface technique, for which the linear terms disappear. The invariance is obtained by a modification of the estimation procedure which results in zero threshold for the units, and zero sums for the input weighting coefficients (in the case of the Quadratic Surface, each row and column of the coefficient matrix has a zero sum).

4.5 Approximation to the Quadratic Surface

The quadratic input units which result when differences in the covariance matrix are to be exploited are very expensive to implement in analog hardware (see Section 4.7.2). Furthermore, the cost of these units is approximately proportional to the number of input connections, as the input mechanism represents most of the cost. This is not true for linear input units, where nominal increases in the number of input connections are virtually free. The studies in Section 5.4.2 indicate that, in the range examined, network capacity depends on the total number of input connections for the most part. Therefore, for

the linear input units, increasing the number of connections represents an economical means of increasing performance.

The Quadratic Surface technique, nevertheless, is more desirable for the experiments on cloud patterns, since the assumptions which result in the linear units exploit distributional differences which should not be present in the Tiros frames (see 4.3). While the Oriented Hyperplane and Perpendicular Bisector techniques should be able to provide networks to classify the sample patterns, since the sample distributions are finite and thus have anomalies, the utility of such networks in generalizing (classifying patterns other than the sample patterns) is questionable.

It is desirable, therefore, to find a linear approximation to the quadratic unit. It is apparently not uncommon in factor analysis to find that a very few principal axes account for nearly all of the total variance — for example, in psychometrics, it is common for the main component to account for about half of the total variance (with the second component accounting for another quarter to a third). If this holds also for cloud patterns, we may simplify the total hardware considerably. As an approximation to the Quadratic Surface technique, we will look for a principal axis of differentiation between the pattern classes, and use only the projection of patterns to define the decision boundary in the subspace.

The transformation T in (12) maps the measurement vectors X into a space in which the negative patterns have a spherical distribution with unit variance in any direction. To find the principal axis of differentiation, we examine the covariance matrix S of the positive patterns in this space. Let η_{min} and η_{max} be the smallest and largest eigenvalues of S, and let E_{min} and E_{max} be the corresponding eigenvectors. If

$$\frac{1}{\eta_{min}} > \eta_{max}$$

then E_{min} lies in the direction of greatest differentiation, and if

$$\frac{1}{\eta_{min}} < \eta_{max}$$

then E_{max} provides the desired direction.

Having found the desired eigenvector, say E_k, only the projections of the transformed measurement vectors on E_k are to be used. These projections have univariate normal distributions for each pattern class. Subject to the constraint that only the projection can be used, the discriminant function (12) becomes

$$D(X) = (X - cU)^t T^t E_k \left(\frac{1}{\eta_k} - 1\right) E_k^t T(X-cU) + \ln\lambda_1^2 + \ln \eta_k$$

$$= \frac{1 - \eta_k}{\eta_k} \left[((X - cU)^t T^t E_k)^2 + \frac{\eta_k}{1 - \eta_k} \ln \lambda_1^2 \ \eta_k \right] \qquad (14)$$

The switching function implied by this discriminant function can be implemented by a linear input logic unit with two thresholds. One possible implementation would have a weight vector of $T^t E_k$ and thresholds of $cU^t T^t E_k \pm \sqrt{\frac{\eta_k}{\eta_k - 1} \ln \lambda_1^2 \ \eta_k}$. The unit would turn on whenever the weighted sum of the inputs fell in the gate between the two thresholds. This "on" region, bounded by two parallel hyperplanes in the logic unit subspace, indicates a positive pattern if $\eta_k < 1$ and a negative pattern if $\eta_k > 1$.

This technique for generating logic units is called the "Parallel Hyperplane" technique in the experiments in Sections 5 and 6. A relatively inexpensive hardware implementation is given in Section 4.7.2.

For this technique, the attempt to provide gray scale invariance produces a different form for the solution. Reducing the sample pattern vectors by a multiple of U again provides invariance against a constant shift in the gray scale (the constant "b" in aX + bU). Again, the matrix T is not square ($T \Sigma T^t = I$). Invariance against contrast changes (indicated by the constant "a") is provided by normalizing the length of X. The quantity on which the logic unit switching is based is then

$$\frac{(X' T' E_k)^2}{X' X}$$

By assumption, the means of the distributions of the reduced vectors is zero. When a threshold for the logic unit is chosen, the boundary of the region in which the critical quantity is less than the threshold is a right circular cone centered on the principal axis. In common with the gray invariant version of the Quadratic Surface technique, this logic unit is invariant to reversals.

Unlike the other three techniques for generating logic units, the gray invariant version of Parallel Hyperplanes is more difficult to implement in hardware than the noninvariant version. A mechanization is given in Section 4.7.2. Despite this increase in difficulty, the resulting device is still less costly than the quadratic unit that it approximates.

4.6 Convergence Proof for Iterative Design

The following mathematical derivation* constitutes a proof that the iterative procedure for adjusting weights (a relaxation process) will, if continued for a sufficiently long time, yield a set of weights resulting in no classification errors on the sample patterns, provided only that such a set of weights exists. Also inherent in this proof is that when the logic unit activity vectors are linearly independent, the iterative process will force the loss function to converge to its smallest possible value, even when no "solution" exists. While the result is clearly true when linear independence does not hold, the proof for this case is not complete.

The proof indicates that is is not necessary to adjust a weight until the partial derivative is zero with respect to that weight — one is permitted to undershoot or overshoot by some amount which depends on the partial derivatives. Simulation results indicate that a fairly high

*The proof is credited to Kesler, C. C. and Mucci, A. G. of the Astropower Laboratory.

degree of overshoot ($5/8 \leq g \leq 3/4$) leads to the fastest convergence.

Let

$$r_i = (d_{i1}, d_{i2}, \ldots, d_{in})$$

denote the ith row vector, and $c_i = \begin{bmatrix} d_{1i} \\ d_{2i} \\ \vdots \\ d_{mi} \end{bmatrix}$ denote the ith column vector, of the m-by-n matrix D, where the d_{jk} are known constants. Consider the set of inequalities $Dw < 0$, where $w = \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix}$ is the weight vector. Let $L = \begin{bmatrix} L_1 \\ \vdots \\ L_m \end{bmatrix} = \begin{bmatrix} e^{r_1 \cdot w} \\ \vdots \\ e^{r_m \cdot w} \end{bmatrix}$ denote the loss vector. Let $(\Sigma L_j)_o$ denote the initial total loss, which is assumed positive. Let g_o be any known positive constant satisfying $g_o^2 < 1$. The w_j's are reinforced one at a time (but D can be specified such that two w_j reinforcements are equivalent to the simultaneous threshold and one weight reinforcement which occurred in the iterative procedure of 4.2). Consider the reinforcement of w_i. Let $y = \partial(\Sigma L_j)/\partial w_i = L \cdot c_i$. Let w_i^o denote the value of w_i, and y_o denote the value of y, just before w_i is reinforced. Δw_i is to be chosen such that

$$\Delta(\Sigma L_j) \leq (g_o^2 - 1) y_o^2 / (2(\Sigma L_j)_o \max_{j,k}(d_{jk})^2). \qquad (15)$$

There are only four possible cases. In each case, $\Delta(\Sigma L_j) = \int d(\Sigma L_j)$

$$= \int_{w_i^o}^{w_i^o + \Delta w_i} y \, dw_i.$$

Case 1: $\sum_{d_{ji} > 0} L_j > 0$ and $\sum_{d_{ji} < 0} L_j = 0$. Let w_i go to minus infinity

slower than all previous w_j's went to plus or minus infinity. Then all L_j's which were zero before w_i was reinforced are still zero after w_i is reinforced.

$$\frac{1}{(\Sigma L_j)_o \max_{j,k}(d_{jk})^2} \int_0^{y_o} -y\, dy \geq \int_{-\infty}^{w_i^o} -y\, dw_i = \Delta(\Sigma L_j).$$

Therefore (15) is true.

Case 2: $\sum_{d_{ji}>0} L_j = 0$ and $\sum_{d_{ji}<0} L_j > 0$. Let w_i go to plus infinity slower than all previous w_j's went to plus or minus infinity. This insures that all L_j's which were zero before w_i was reinforced are still zero after w_i is reinforced.

$$\frac{1}{(\Sigma L_j)_o \max_{j,k}(d_{jk})^2} \int_{y_o}^{0} y\, dy \geq \int_{w_i^o}^{+\infty} y\, dw_i = \Delta(\Sigma L_j).$$

Therefore (15) is true.

Case 3: $y_o = 0$. Define $\Delta w_i = 0$. Then (15) is satisfied.

Case 4: $y_o \neq 0$, $\sum_{d_{ji}>0} L_j > 0$, and $\sum_{d_{ji}<0} L_j > 0$. Assume each d_{jk} is a member of the set (-1, 0, +1). Define Δw_i by $e^{\Delta w_i} = (-\frac{gy_o}{2} + ((\frac{gy_o}{2})^2 + \sum_{d_{ji}=+1} L_j \sum_{d_{ji}=-1} L_j)^{1/2}) / \sum_{d_{ji}=+1} L_j$, where g is any known number satisfying $g^2 \leq g_o^2$. Then just after w_i is reinforced, $y = -gy_o$. Since only w_i varies, for any number c, $\partial y/\partial w_i \big|_{y=c} = \partial y/\partial w_i \big|_{y=-c}$. If

$$y_o<0,\ \Delta(\Sigma L_j) = \int_{y_o}^{-gy_o} y/(\partial y/\partial w_i)dy = \int_{y_o}^{|g|y_o} y/(\partial y/\partial w_i)dy \leq \frac{1}{(\Sigma L_j)_o \max_{j,k}(d_{jk})^2} \int_{y_o}^{|g|y_o} y\, dy;$$

and if

$$y_o>0,\ \Delta(\Sigma L_j) = \int_{-gy_o}^{y_o} -y/(\partial y/\partial w_i)dy = \int_{|g|y_o}^{y_o} -y/(\partial y/\partial w_i)dy \leq \frac{1}{(\Sigma L_j)_o \max_{j,k}(d_{jk})^2}\int_{|g|y_o}^{y_o} -y\,dy. \tag{16}$$

Thus (15) is satisfied. Assume it is not true that each d_{jk} is a member of the set (-1, 0, +1). If Δw_i is chosen such that just after w_i is reinforced, y = 0, then (16) holds for g = 0. Thus (15) is satisfied. Again consider the reinforcement of w_i. For all

$$h,\ |\Delta(L\cdot c_h)| = |\Sigma_j (e^{d_{ji}\Delta w_i}-1)L_j d_{jh}| \leq \Sigma_j |(e^{d_{ji}\Delta w_i}-1)|\ L_j \max_{k,p}|d_{kp}| \leq$$

$$((\max_{k,p}|d_{kp}|)/(\min_{d_{kp}\neq 0}|d_{kp}|))\Sigma_j |(e^{d_{ji}\Delta w_i}-1)|\ L_j|\ d_{ji}| = ((\max_{k,p}|d_{kp}|)/$$

$$(\min_{d_{kp}\neq 0}|d_{kp}|))|\Sigma_j (e^{d_{ji}\Delta w_i}-1)L_j d_{ji}| = ((\max_{k,p}|d_{kp}|)/(\min_{d_{kp}\neq 0}|d_{kp}|))|\Delta(L\cdot c_i)|. \tag{17}$$

It will be assumed that one of the following two restrictions is placed on the sequence of reinforcements. There is a constant b such that within every b consecutive reinforcements, either (1) every w_j has been reinforced or (2) some w_k was reinforced just after determining that

$$(L\cdot c_k)^2 = \max_j (L\cdot c_j)^2. \tag{18}$$

By (15), ΣL_j approaches a limit because ΣL_j is a bounded monotonic sequence. Thus for any $\phi<0$, there is a time t_o such that for $t>t_o$, $\phi < \Delta(\Sigma L_j) \leq 0$. Therefore by (15), for any $\phi>0$, there is a time t_1 such that for $t>t_1$, $0 \leq y_o^2 < \phi$ between reinforcements. Thus by (17) and (18), we approach the state where for all j, $L\cdot c_j = 0$ between reinforcements. Therefore, between reinforcements we approach the state where $L_j = 0$ for the inequalities in the set $Dw<0$ which are not in contradiction. So if a solution of $Dw<0$ exists, L_j approaches zero, and hence w becomes a solution of $Dw<0$ in a finite amount of time.

4.7 Hardware Considerations

4.7.1 Discussion

Analog mechanization of the recognition system of Figure 2 depends to a large extent on the parameters resulting from the simulation program. These include:

- Resolution — the number of resolvable elements required for pattern identification.
- Number of property filters required (first layer logic units).
- Discriminant function to be mechanized by each logic unit — included in this consideration are the number of inputs per unit, the required range of inputs per unit, the required range of input weights, The threshold value(s), and the allowable variation in specification of the function.

A further consideration is the requirement for a continued adaptive capability. This would allow incorporation of new pattern classes or the accommodation of variations in the form of the patterns to be classified due to changes in the input system.

Simulation results indicate the need for several hundreds of logic units to classify the cloud patterns as to the presence or absence of vortices. This would not seem to preclude the development of an analog system except for two of the above mentioned considerations, the complexity of the logical operation performed by each unit and the suspected requirement for a continued adaptive capability.

As a result of the studies performed, analog mechanization of each of the discriminant functions discussed in Section 4.3 has been accomplished. Particular attention was given to the design of a logic unit capable of generating the normalized parallel hyperplane function. The circuits so constructed and tested are discussed in the

following section. Section 4.7.3 discusses the overall system considerations with emphasis on the development of a special purpose digital system capable of mechanizing a variety of property filters with possible continuing adaptive capabilities.

4.7.2 Property Filter Mechanization

Self-organizing design of the first layer logic units is not limited to the linear thresholding elements or linear discriminants. From a strict mechanization standpoint, the advantages of more complex logic units are:

(a) reduction of the number of logic units required to mechanize the system and
(b) reduction of the number of interconnections required.

Consider an extremely simple case (Figure 4) of isolating a single point in two-dimensional space by means of various discriminant functions. The weights are considered as an integral part of the illustrated logic units. Notice as the complexity of the logic unit increased the number of first layer logic units and interconnections decreases, as does the required number of interconnections and weights for the second layer of logic. Unfortunately as the complexity of the function increases the resulting hardware becomes more difficult to mechanize giving rise to increased cost. Therefore a trade-off must be considered. The final decision rests upon the results of the simulation study which evaluates the penalties to be paid in terms of the efficiency of the resulting classification devices for simplicity of design.

For analog mechanization major portions of the individual logic units are readily synthesized with the aid of existing linear integrated circuits. Fortunately the "building block" in the recognition system is a differential dc amplifier of nominal gain. This type of amplifier is currently available in semiconductor integrated circuit form. Simple amplifiers constructed with discrete components require from five to fifteen total parts with the utility of the amplifier

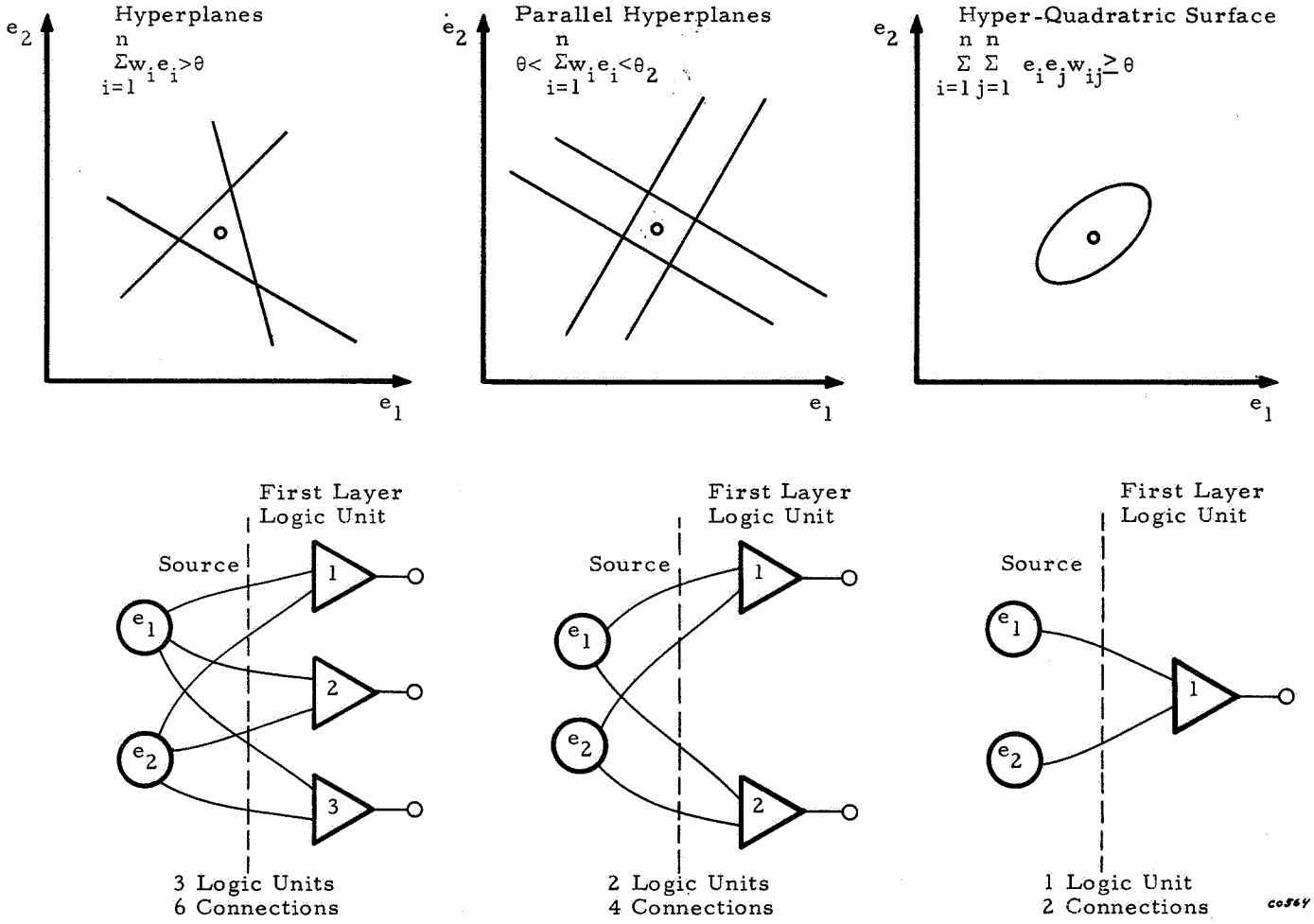


Figure 4. Hardware vs Discriminant Function Complexity

increasing as the parts count increases. The assembly-plus-parts cost of such simple circuits using discrete components will be about ten dollars. Current integrated circuit amplifiers packaged in epoxy cases will sell for prices between two (low grade commercial) and five dollars (high grade commercial) by the beginning of 1966. In addition to the price advantage, the thermal properties and general performance of even low grade integrated circuit amplifiers is far superior to those that can be constructed using discrete components of similar price. Further advantages of the integrated circuit include reduced system size and reduced cost of interconnections.

Examples of two useful I.C. amplifiers are shown in Figures 5 and 6. These circuits will soon be available in a commercial epoxy package (.360" O.D. by .20"). These units can be used in part to implement the various logic units investigated. The following material discusses each of the circuits and the results from the laboratory experiments.

Single Hyperplane or Linear Threshold Unit

The equation to be mechanized is simply:

$$\sum_{i=1}^{n} w_i e_i > \theta \quad \text{(see Figure 7)}$$

$$w_i = \frac{R_p}{R_i} \qquad \text{where } R_p = \frac{1}{\frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_3} + \cdots \frac{1}{R_n}}$$

The allowable differential input voltage is ±5 volts.

$$e_o \cong o \text{ if } \sum_{i=1}^{n} w_i e_i < \theta - (5 \text{ mv})$$

$$e_o \cong +3 \text{ if } \sum_{i=1}^{n} w_i e_i > \theta + (5 \text{ mv})$$

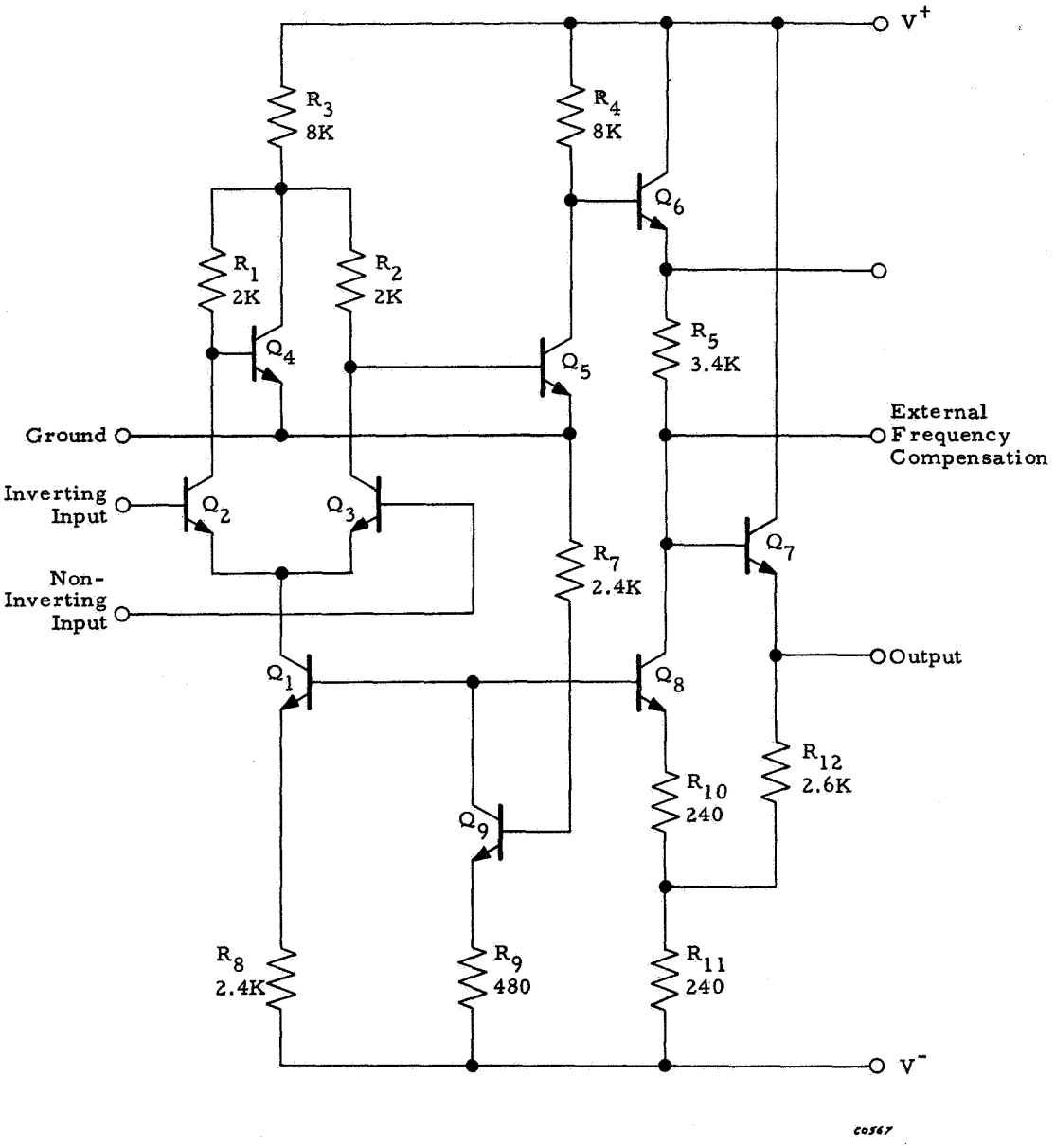


Figure 5. Operational Amplifier

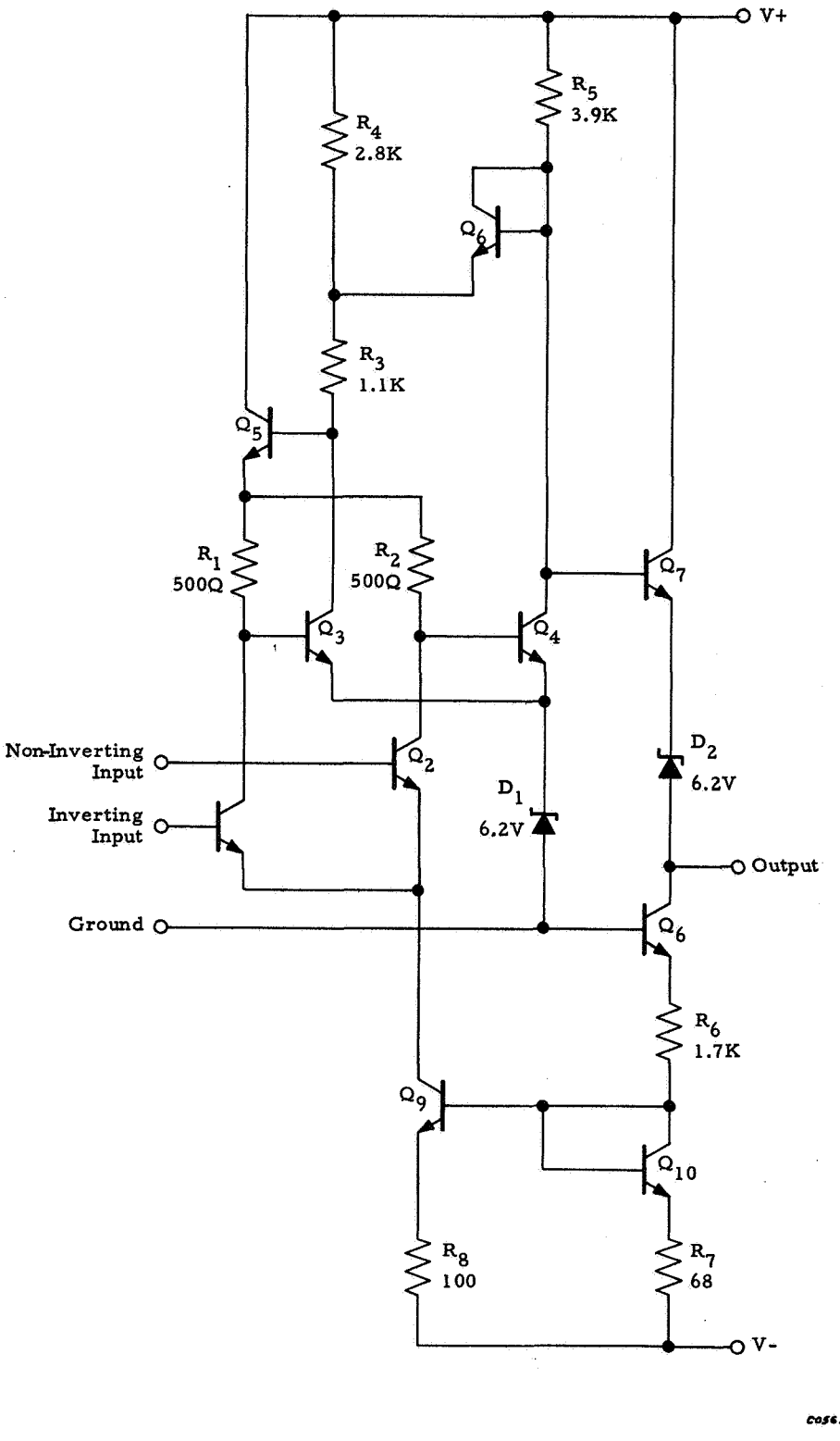


Figure 6. Differential Voltage Comparator

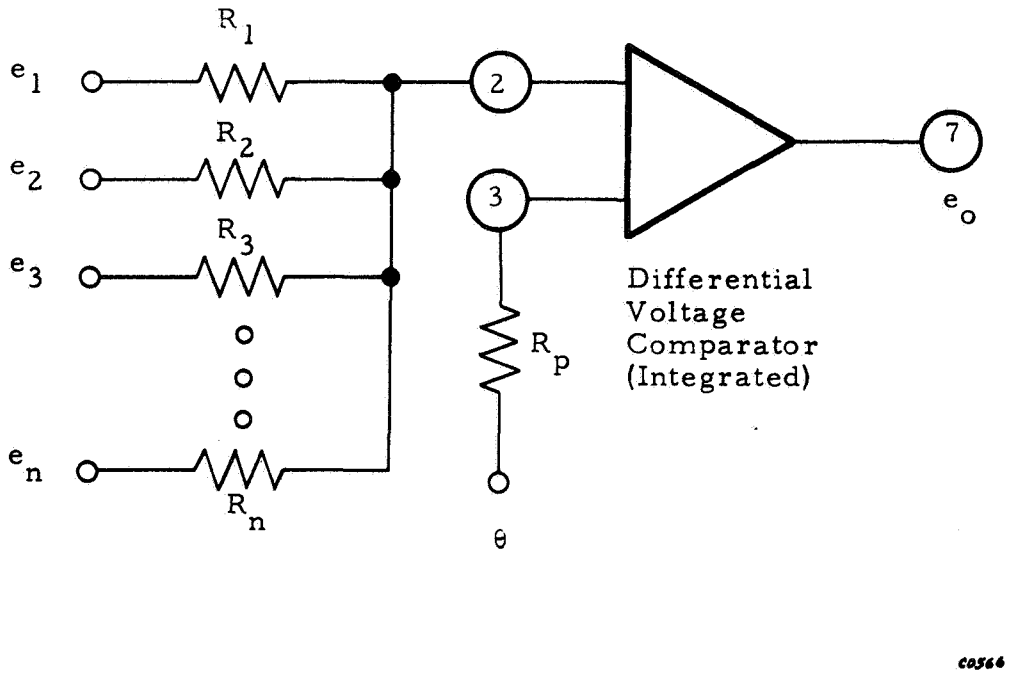


Figure 7. Linear Threshold Logic Unit (I. C.)

Optimum sensitivity and minimum drift are obtained by using small values of R_p; however, such optimization is not a necessary system requirement. A reasonable range for R_p would be from 5 to 50KΩ.

An alternate approach for a single hyperplane using discrete circuit components is shown in Figure 8.

This circuit is useful where temperature variations are not a concern. Generally it is necessary to add an emitter follower to reduce the output impedance and allow mating to the next logic layer. For R_p <50KΩ the sensitivity of the circuit is typically ±20 millivolts.

Parallel Hyperplanes

The function to be mechanized is

$$\theta_{min} < \sum_{i=1}^{n} w_i e_i < \theta_{max}$$

Using integrated circuits the function may be obtained as illustrated in Figure 9 in a rather direct fashion.

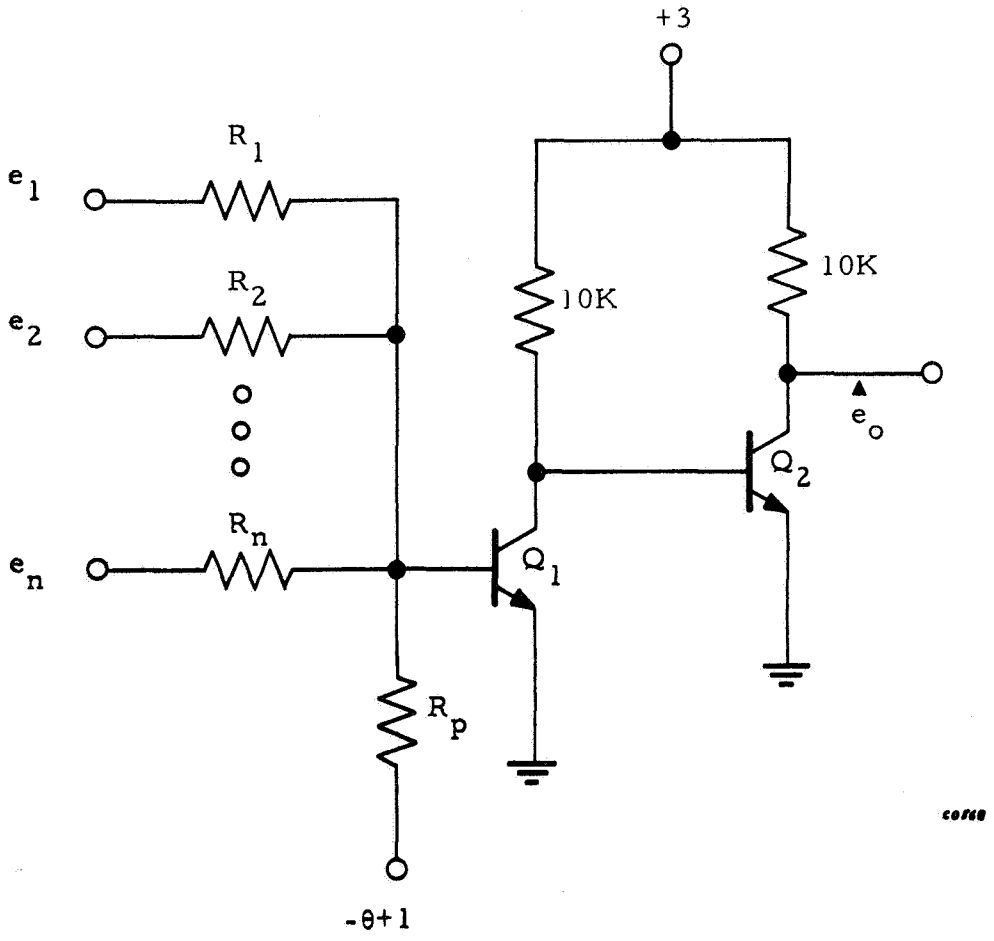
For simplicity of laboratory evaluation a unit was constructed with n = 2. The design weights and threshold for the unit tested were

$$w_1 = 1/2;\ w_2 = -1/2;\ \theta_{max} = +1;\ \theta_{min} = -1/2$$

The ideal and actual boundaries at which a transition occurs are then given as follows:

Ideal	e_2	=	1.000 e_1	- 2.000
Actual	e_2	=	1.008 e_1	- 2.031
Ideal	e_2	=	1.000 e_1	+ 1.000
Actual	e_2	=	0.989 e_1	+ 1.019

The expression for the actual curves used the method of least squares to obtain the slope and intercept from the data



Implementation with Discrete Components

Figure 8. Linear Threshold Logic Units

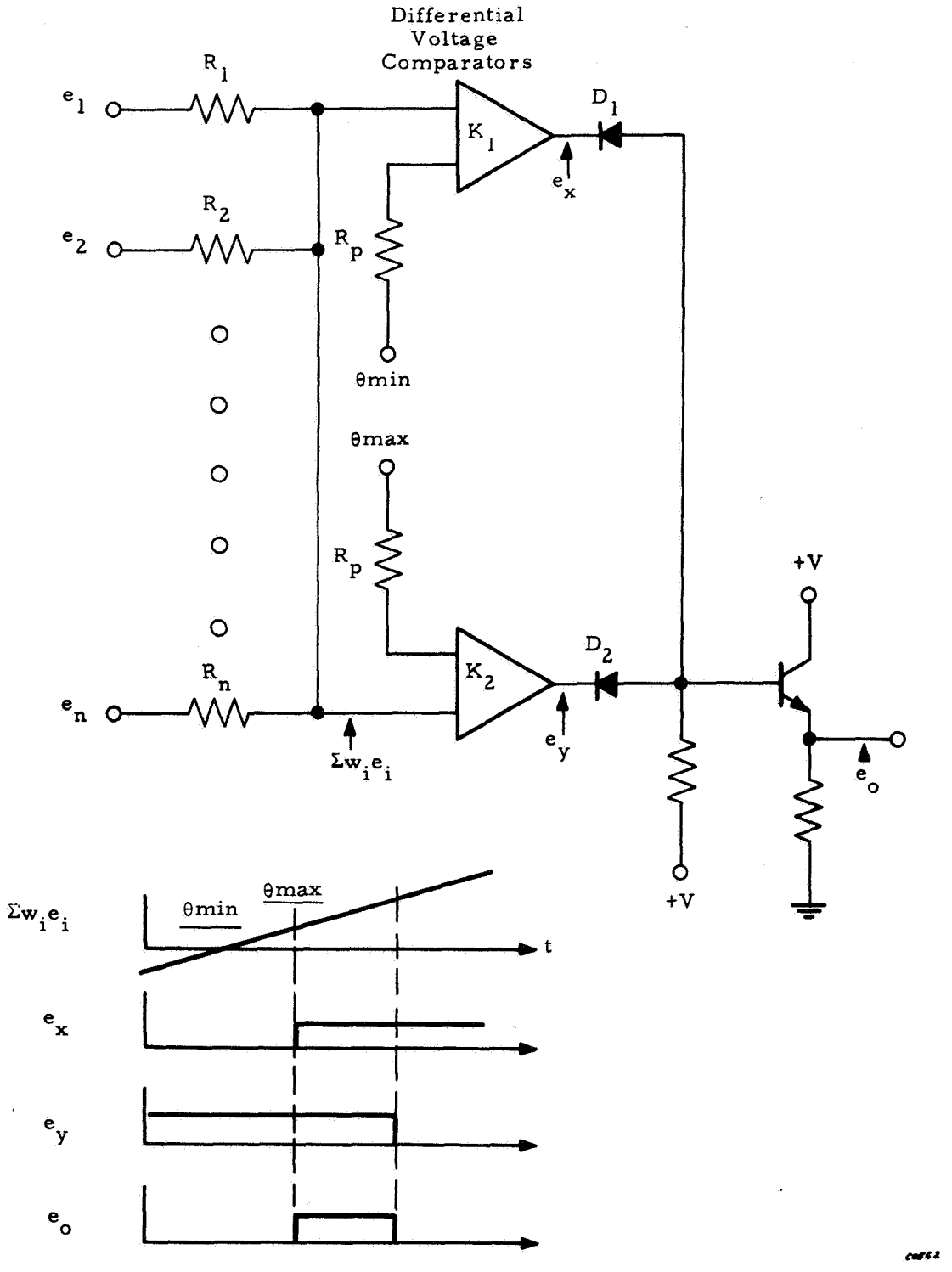


Figure 9. Parallel Hyperplane (I. C.)

points obtained in the laboratory. The circuit used in the evaluation is shown in Figure 10 where:

$$w_i = \frac{R_p}{R_i}$$

and

$$R_p = \frac{1}{\frac{1}{R_1} + \frac{1}{R_2} + \cdots \frac{1}{R_n}}$$

The inverse input voltage, $\bar{e}_2$ (zero to minus 5 volts) is input to the circuit to produce a negative weight. Positive weights are implemented by causing each e_i to be between zero and plus 5 volts. In general

$$\bar{e}_i = -e_i$$

The comparison between the ideal and actual parallel hyperplane logic unit indicates that an RMS error of less than 3% may be expected. This error could easily be further reduced by the use of 1% resistors for the input weighting network.

Normalized Parallel Hyperplane

The circuit to be mechanized has the analytic form:

$$\frac{(\Sigma w_i e_i)^2}{\Sigma e_i^2} \leq \theta$$

Which is equivalent to

$$\left(\Sigma \frac{w_i}{\sqrt{\theta}} e_i\right)^2 - \Sigma e_i^2 \leq 0$$

If we let $w_i = \frac{w_i}{\sqrt{\theta}}$

Then the expression may be more simply mechanized by the form:

$$(\Sigma w_i e_i)^2 - \Sigma e_i^2 \leq 0$$

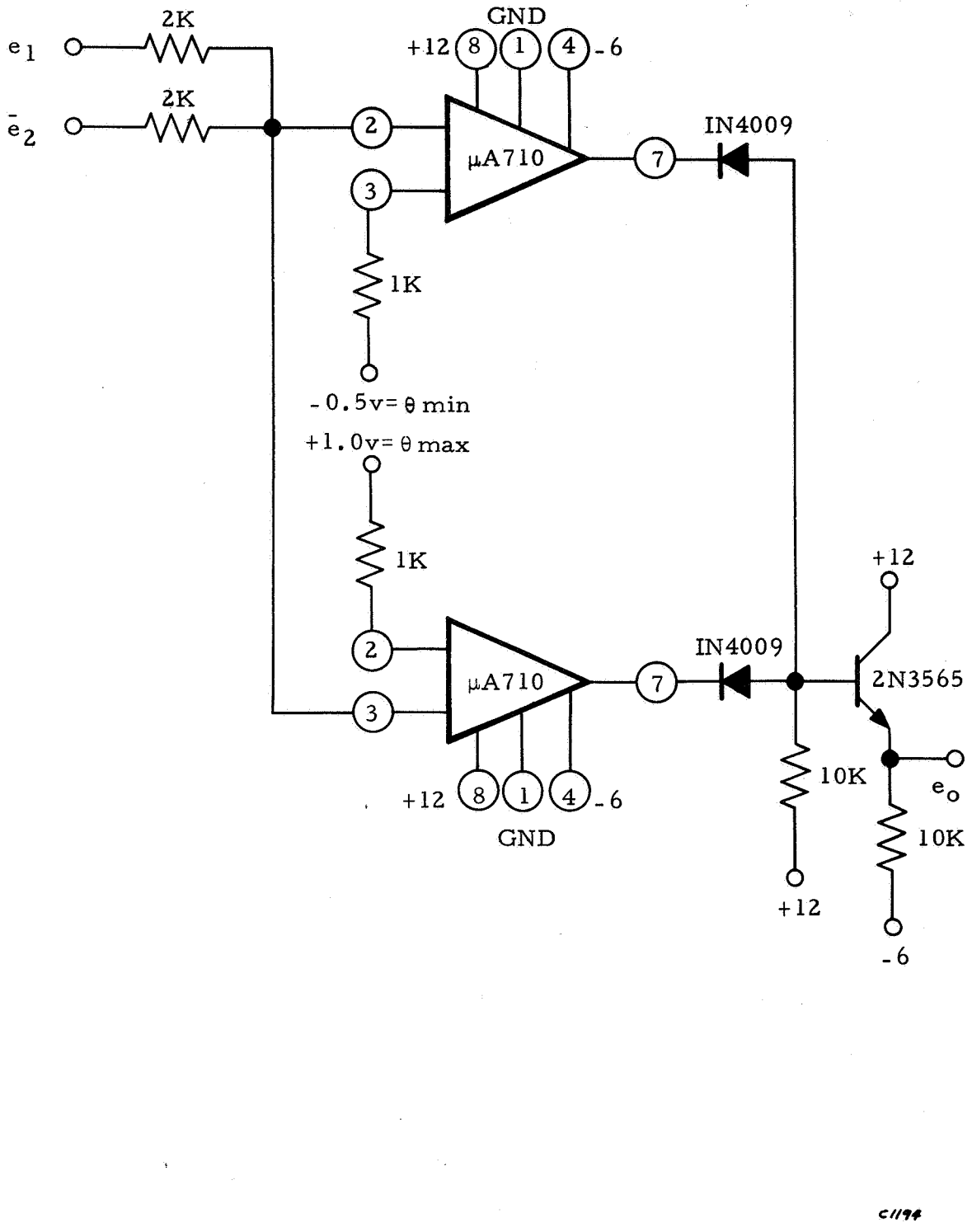


Figure 10. Parallel Hyperplane (laboratory model)

This expression involves the use of a squaring circuit. Some of the well known squaring techniques are

1) Diode approximation
2) Hall generators
3) Thyrites

The diode approximation and Hall generator techniques are not economically desirable when many individual terms are to be squared. Thyrite appears to be a reasonably economical alternative, particularly for mechanizing Σe_i^2. The relation between the current and voltage in a thyrite resistor is given by

$$I = Ke^n \qquad \text{where} \qquad 2.5 < n < 6.5$$

Notice that $n > 2$ so that the thyrite characteristic does not produce a current proportional to the square of the applied voltage. An economical means for reducing the current is to add a resistor in series with the thyrite and approximate the squared curve.

When a resistor is in series with the thyrite

$$e_{in} = iR + Ci^{1/n}$$

$$\text{where } C = \frac{1}{K^{1/n}}$$

$$\text{So that} \qquad R = \frac{e_{in} - Ci^{1/n}}{i}$$

R should be chosen such that the current which flows is related to the square of the input voltage, that is

$$i = Ke_{in}^2$$

The corresponding resistance value is

$$R_{ideal} = \frac{e_{in} - e_{in}^{2/n}}{i} = \frac{1}{Ke_{in}} \left[1 - e_{in}^{-\left(\frac{n-2}{n}\right)} \right]$$

The values of the ideal resistor are given in Table II for $K = 6 \times 10^{-6}$ and $n = 2.78$ corresponding to the typical performance of GE 6505010.

TABLE II

Series Resistor Values

e_{in}	R_{ideal}
1/4	-315KΩ
1/2	- 71KΩ
1	0
2	+14.6KΩ
3	+14.7KΩ
4	+14.5KΩ
5	+12.7KΩ

A graph which illustrates the effectiveness of a resistor in series with the thyrite is shown in Figure 11. The actual value of the series resistor used to approximate the ideal square characteristic is most readily determined empirically due to variation of the thyrite. If a function generator which generates the ideal curve is input to an oscilloscope and the actual curve is displayed simultaneously, the value of the resistance required may be easily determined, with the aid of a decade box, by visual comparison of the two curves.

A laboratory arrangement used to mechanize the normalized parallel hyperplane is shown in Figure 12. The signal inputs should be positive valued voltages having a source impedance small relative to 20KΩ. The output voltage is given by

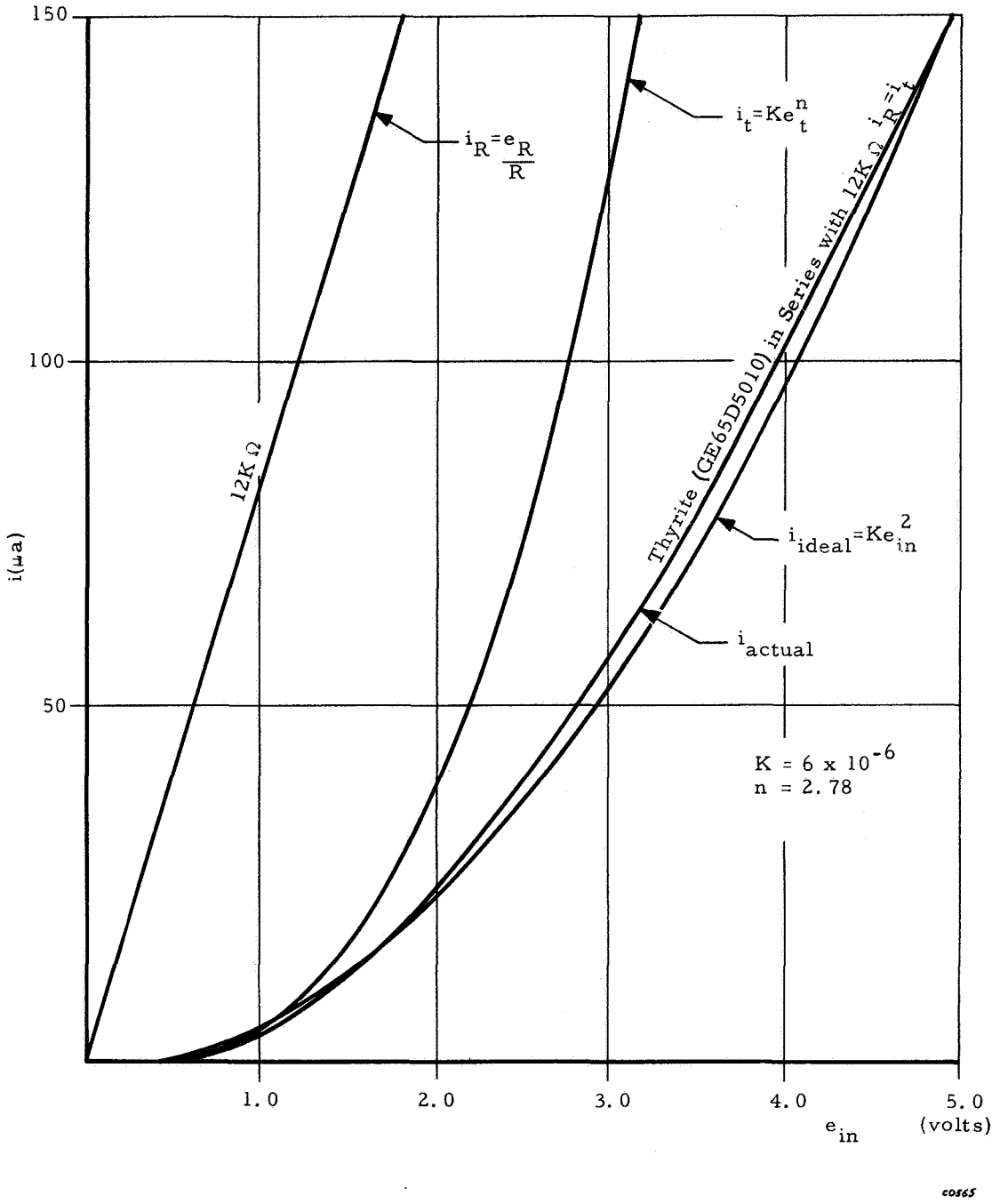


Figure 11. Function Generated by Thyrite with Series Resistor

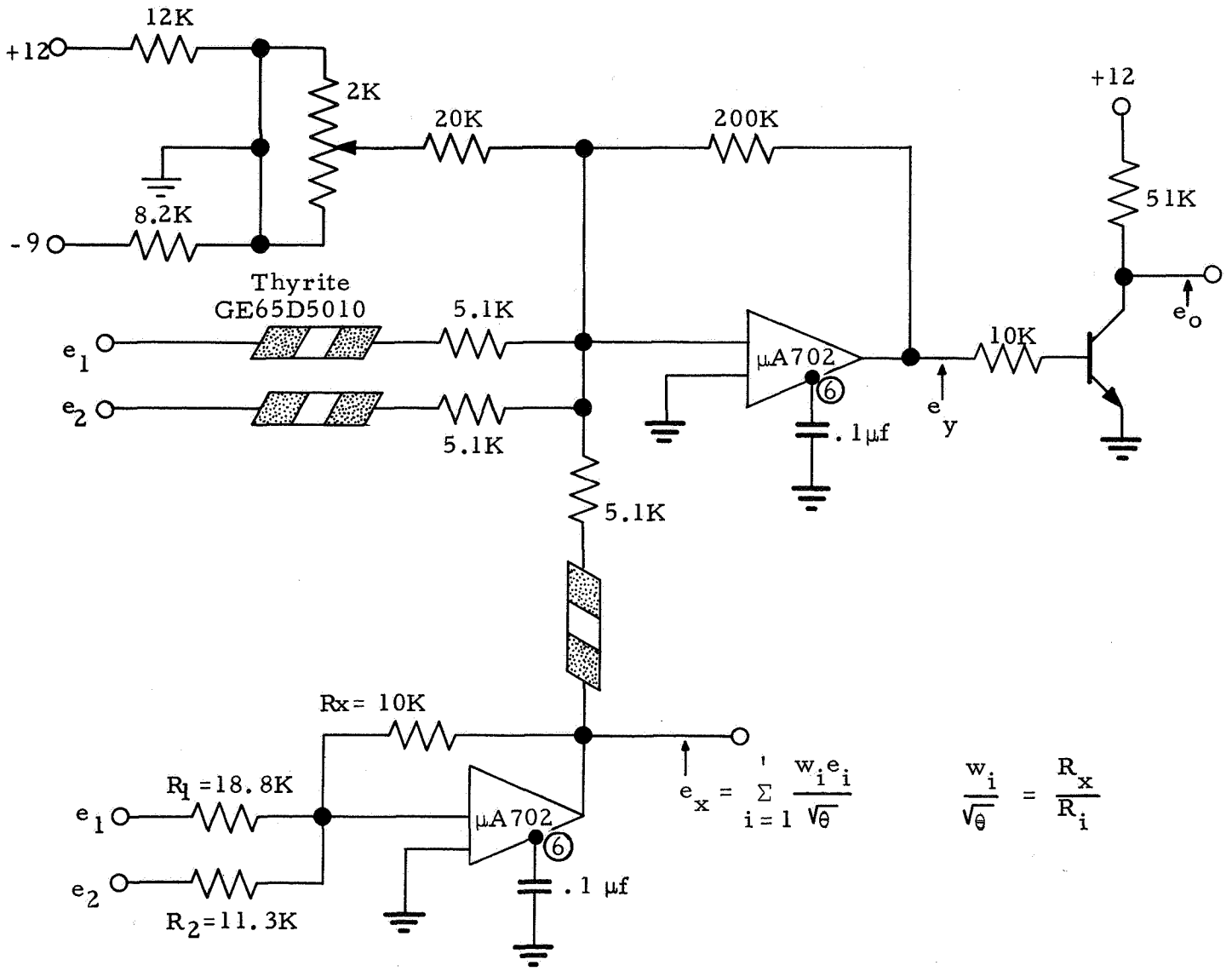


Figure 12. Normalized Parallel Hyperplane

$$e_o = +V \quad \text{if} \quad \frac{(\Sigma w_i e_i)^2}{(\Sigma e_i^2)} \leq \theta$$

$$= 0 \quad \text{if} \quad \frac{(\Sigma w_i e_i)^2}{(\Sigma e_i^2)} \geq \theta$$

The amplifiers are readily implemented with an integrated circuit operational amplifier similar to that previously illustrated. The size of the disc-shaped thyrite resistors is 0.55 inch diameter by 0.21 inch thickness. This permits construction of a relatively compact module.

In n-dimensional space the boundary is described by a cone with the apex located at the origin of the space. The output voltage, e_o, should be in the "ON" state when the input voltages (coordinates of the space) project into a point outside the cone and in the "OFF" state when they project inside the cone. For the laboratory model the number of inputs was limited to two for ease of evaluation. This results in the switching surface demonstrated in Figure 13. To evaluate the circuit performance input voltage, e_1 was held constant at a specified value and e_2 varied until e_o switched states. Input e_2 was then further increased until the output voltage switched back to its original state. For the circuit of Figure 12 the calculated equations of the lines forming the boundary are

$$e_2 = e_1$$

$$e_2 = 3.28\, e_1$$

Table III shows the predicted and actual value of e_2 where switching occurred for specified values of e_1.

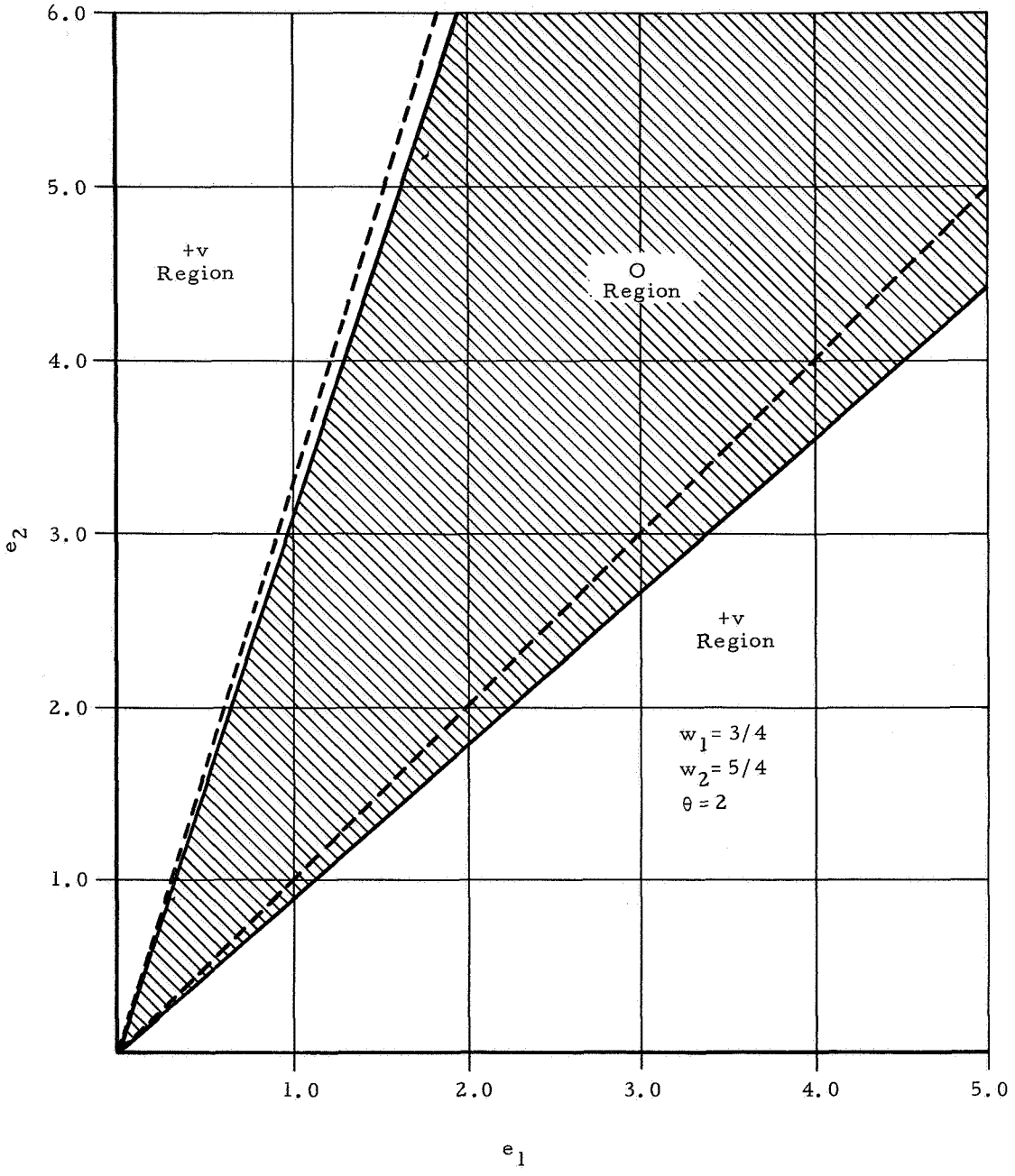


Figure 13. Switching Surface for Normalized Parallel Hyperplane

TABLE III

Operation of Normalized Parallel Hyperplane Circuit

Input Voltage e_1	Input Voltage e_2 (points at which e_o switched states)			
	Calculated	Measured	Calculated	Measured
0.5	0.5	0.44	1.64	1.55
0.75	0.75	0.74	2.46	2.29
1.00	1.00	1.00	3.28	3.14
1.50	1.50	1.42	4.92	4.84
2.00	2.00	1.78	6.56	6.14
3.00	3.00	2.53		
4.00	4.00	3.40		
5.00	5.00	4.50		

It should be pointed out that this performance was achieved only after careful selection and matching of thyrite varistors. Even at that it must be regarded as poor. The primary cause of this large deviation from ideal is attributed to the lack of uniformity of the thyrite properties. The value of K and n of 25 units of the 65D5010 were measured. It was found that K varies from 5.3 to 18.4 μa/volt and n from 2.048 to 2.351. The best performance was obtained where units having a reasonably similar K were selected.

Hyper-Quadratic Surface

The equation of a hyper-quadratic surface is given by

$$\sum_{i=1}^{N} \sum_{j=1}^{N} e_i e_j w_{ij} \gtrless \theta$$

The mechanization technique shown is for N=4. It will be clear that for additional cost the values of N may be larger.

If N=4 then

$$\Sigma\Sigma e_i e_j w_{ij} = e_1 f_1 + e_2 f_2 + e_3 f_3 + e_4 f_4 = e_x$$

where

$$f_1 = w_{11}e_1 + w_{12}e_2 + w_{13}e_3 + w_{14}e_4$$

$$f_2 = w_{21}{}^2{}_1 + w_{22}e_2 + w_{23}e_3 + w_{24}e_4$$

$$f_3 = w_{31}e_1 + w_{32}e_2 + w_{33}e_3 + w_{34}e_4$$

$$f_4 = w_{41}e_1 + w_{42}e_2 + w_{43}e_3 + w_{44}e_4$$

A possible technique for mechanizing this function utilizes four Hall generators to obtain the product e_1f_1, e_2f_2, e_3f_3, and e_4f_4. The sums f_1, f_2, f_3 and f_4 are readily obtained by summing the currents due to e_1, e_2, e_3 and e_4 in the coils associated with each generator. The summation of $e_1f_1 + e_2f_2 + e_3f_3 + e_4f_4$ is obtained with an operational amplifier. This amplifier boosts the low level Hall voltages as well.

A simplified diagram of this method is shown in Figure 14. The output voltage of the Hall generator using a toroidal core is given by:

$$v_h = \left(\frac{R_h}{d} i_s\right)\left(\frac{0.4\pi kN}{\ell + (k-1)g} i_f\right) = K_h K_m i_f i_s$$

where

k = relative permeability

g = air gap width (cm)

ℓ = total length of the mean magnetic path (cm)

R_h = Hall constant

d = thickness of semiconductor

i_s = current in the Hall generator

i_f = current in the magnetic field coil

K_h = open circuit Hall coefficient (volts/amp kilogauss)

K_m = magnetic circuit coefficient (kilogauss/amp)

for a Magnetic, Inc. core #55251 which has an OD = 1.570", I.D. = .950" and thickness = .570", and mean path length = 9.87 cm, and relative

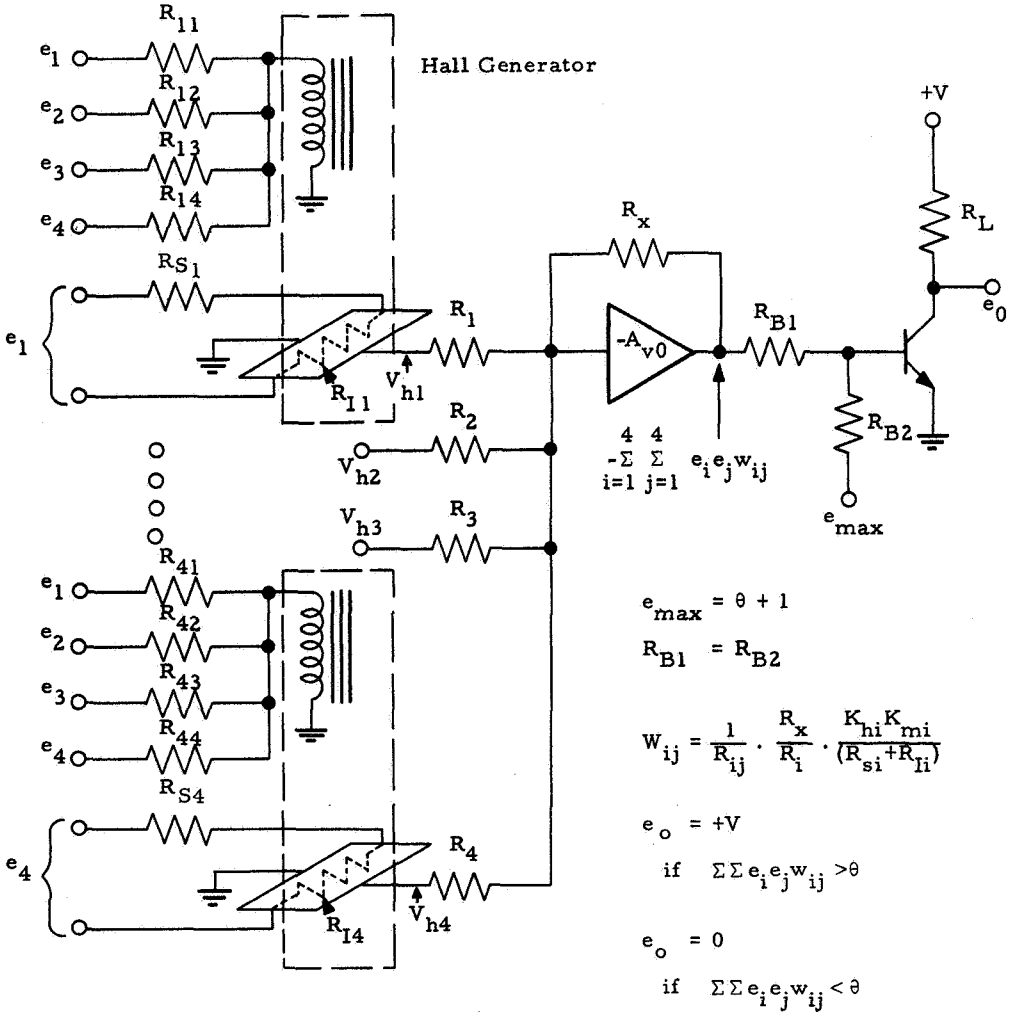


Figure 14. Quadratic Surface Mechanization

permeability k = 200. With this core k will be essentially constant up to 4000 gausses which may be taken as B_{max}. Employing the Beckman model 331 Hallefex device with thickness .020" and allowing .010" for insertion of Hall generator then g = .030' $\cong$ .075 cm.

So that $K_m = 10.5N$

This core will accommodate 1900 turns of AWG #28 wire with a current capacity of .266 amps based on 750 circular mils/amp. The resistance of the coil will be approximately

$$R_{coil} = (1900 \text{ turns})\ (.188 \frac{\text{ft}}{\text{turn}})\ \ (.0653 \frac{\text{ohms}}{\text{ft}}) = 23\ \Omega$$

The inductance of the coil will be

$$L_{coil} = N^2 A \frac{0.4\pi k}{\ell+(k-1)g}$$

for 1900 turns and since $A = 1.056\ cm^2$

$$L_{coil} = 41.4\ mH$$

The value for K_m with N = 1900 is:

$$K_m \cong 20 \text{ kilogauss/amp}$$

One type of the model #331 Hallefex has K_h varying from .2 to .5 volts/amp kilogauss. The lack of uniformity of these coefficients may be compensated for by R_1, R_2, R_3, and R_4. Using a value of K_h = .5 volts/amp kilogauss gives

$$v_h = 10 \frac{\text{volts}}{(\text{amp})^2}\ i_f i_s$$

By choosing $R_{coil} << \dfrac{1}{\frac{1}{R_{11}} + \frac{1}{R_{12}} + \frac{1}{R_{13}} + \frac{1}{R_{14}}}$

$$i_{f_1} = \frac{e_1}{R_{11}} + \frac{e_2}{R_{12}} + \frac{e_3}{R_{13}} + \frac{e_4}{R_{14}}$$

The input resistance of the #311 Hallefex device, R_I, is from 100 to 200 Ω. If we designate R_{S1} as the source resistance

and R_{I1} as the input resistance of the first unit then

$$i_{S1} = \frac{e_1}{R_{S1} + R_{I1}}$$

if $R_{S1} >> R_{I1}$

$$i_{S1} = \frac{e_1}{R_{S1}}$$

$$v_{h1} = e_1 \left[\frac{K_{h1}K_{m1}e_1}{R_{11}R_{S1}} + \frac{K_{h1}K_{m1}e_2}{R_{12}R_{S1}} + \frac{K_{n1}K_{m1}e_3}{R_{13}R_{S1}} + \frac{K_{h1}K_{m1}e_4}{R_{14}R_{S1}} \right]$$

Since this term is multiplied by gain of the operational amplifier then

$$e_1 f_1 = e_1 \left[\frac{R_x}{R_1} \frac{K_{h1}K_{m1}}{R_{11}R_{S1}} e_1 + \frac{R_x}{R_1} \frac{K_{h1}K_{m1}}{R_{12}R_{S1}} e_2 + \frac{R_x}{R_1} \frac{K_{h1}K_{m1}}{R_{13}R_{S1}} e_3 \right.$$

$$\left. + \frac{R_x K_{h1}K_{m1}}{R_1 R_{14}R_{S1}} e_4 \right]$$

$$= e_1 \left[w_{11}e_1 + w_{12}e_2 + w_{13}e_3 + w_{14}e_4 \right]$$

equating coefficients it is seen that in general

$$w_{ij} = \frac{R_x K_{hi}K_{mi}}{R_i \cdot R_{ij} (R_{Si} + R_{I1})}$$

The value of this approach is questionable with current devices if large values compared to unity are required for w_{ij}.

4.7.3 System Considerations

This section discusses possible implementation techniques for parallel logic recognition systems. The structure of the recognition system is similar to Figure 5 and consists of an input sensing or pattern display device, a parallel layer of property filters and an output classification or response unit with weighted interconnections between logic layers. It will be assumed that by means of computer simulation

the weights and thresholds of all property filters and the response unit are known. In addition the connection arrangement between the elements of the pattern and the property filter is known. Each property filter connects to the response unit through a known weight. The approach to this problem that is considered requires that the input pattern be available in a parallel form for a fixed time duration.

General Purpose Sequential Recognition System

This approach is general purpose in the sense that various types of property filters may be mechanized. In addition, a multitude of classification problems may be implemented with the same equipment. The complete design of this system is beyond the scope of this discussion. The intent here is simply to define the function of the constituent parts of the system as illustrated in Figure 15. It is assumed that the image is displayed by a film compatible with the TIROS frames with 256 x 256 lines/frame. Each property filter has ten weighted input connections, a threshold and an output weight.

The tape recorder is used as a memory device to store the parameters (weights, connections and threshold) of the property filters and response units. The information stored on the tape is presented to the tape buffer and control, in a format similar to that shown in Figure 16, eight data bits at a time plus an odd parity bit. The X and Y coordinates of each point are each 8 bits to accommodate the 256 x 256 lines and are recorded in binary code. The weights, w_i, are recorded as 6 bits. Negative weights are written on the tape in 2's complement form. Following each set of ten data coordinates and corresponding weights the threshold, θ_i, of the logic unit being considered and the output weight, u_i, of the particular logic unit is recorded as a 6 bit number. In addition, binary information is recorded on the tape to allow sensing of the first coordinate to be entered as well as the end of tape which follows the threshold of the response unit, φ_n. The beginning and end of a tape is determined by forcing a parity error to exist in the odd parity bit in addition to a particular 8 bit digital pattern. . A blanking space is

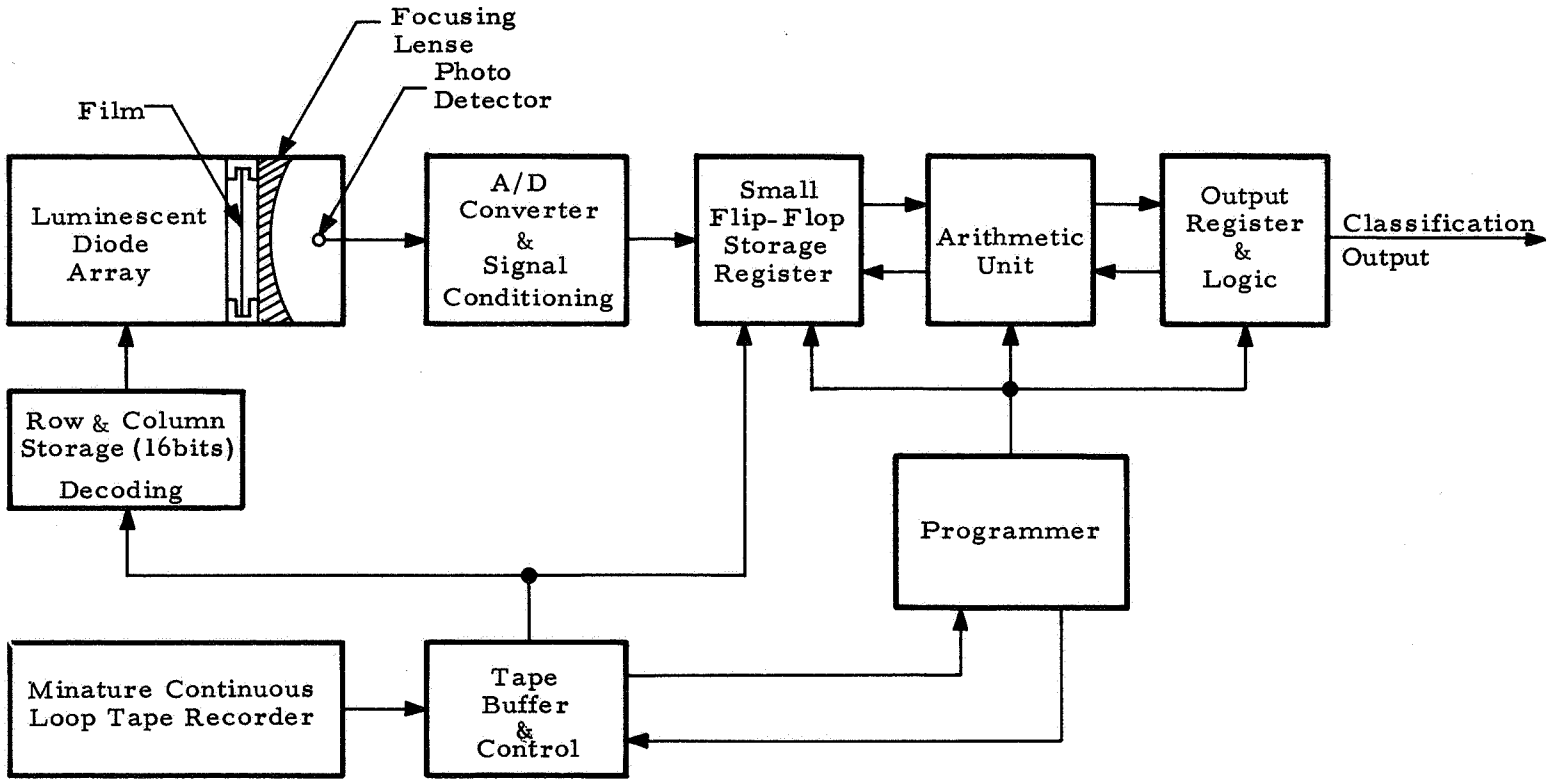


Figure 15. General Purpose Recognition System

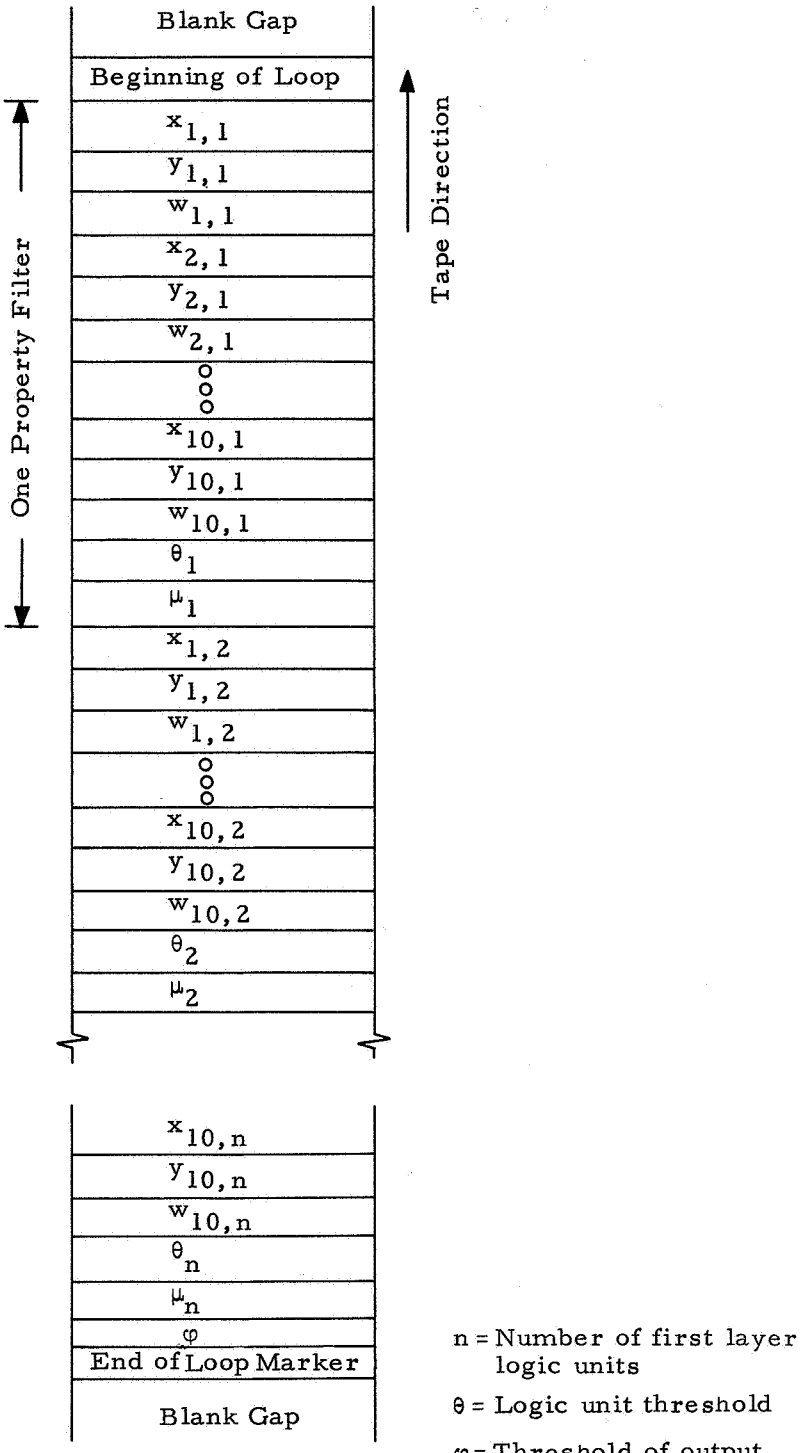


Figure 16. Tape Format

provided on the tape to allow time for advancement of the film to the next frame after pattern classification. The 8 bit characters from the tape are loaded into the tape buffer unit. The tape can be generated directly as a result of the computer simulation program. Different recognition systems can be implemented simply be entering a new tape into the machine.

The amount of tape required is surprisingly small. For example, assume that information is packed at a reasonable density of 200 characters/inch along the horizontal axis of the tape. Then to implement a machine having 2000 property filters each of 10 inputs and one output would require (2000)x $(\frac{32 \text{ characters}}{\text{logic unit}})$. This is only 64,000 characters in a single track or 320 inches of tape. If the tape speed is taken as 60 inches/sec then 5.33 secs per film frame is required. Allowing for a blanking gap of approximately 40 inches corresponds to .67 secs. A new frame may be analyzed every 6 secs. The maximum allowable time for accessing 10 data points, encoding these points, and arithmetically generating a new logic unit is $\frac{32}{200} \cdot \frac{1}{60} = 2.66$ msec.

The luminescent diode array provides a means of emitting dots of light at discrete points in a two-dimensional array. When a reverse voltage is applied to any particular diode in the array the diode avalanches and emits light having a spectral characteristic similar to that of a black body radiator at 2500°K. The light energy emitted is proportional to the current through diode. Typical diode sensitivity is 7×10^{-8} watts/amp. The current through the device is normally maintained at less than 100 ma. The voltage to cause a breakdown should be greater than 6 volts but should not exceed 13 volts. If, for example, the interrogation signal is 50 ma then the visible light energy emitted would be about 3500 pico watts or, in terms of light flux, approximately 1 microlumens.

The light emitting diode size is typical .002 x .002 inches. The illumination in foot candles is in the order of 36 foot candles prior to transmission through the film frame being analyzed. This appears

to be suitable for detection with cadmium sulfide photoconductive cells. However this type of cell is slow acting with typical time constant of 70 msec when illuminated with 10 footcandles. A more suitable detector would be a silicon photoconductive diode. The sensitivity of such cells in the visible spectrum is approximately .05 microamperes/foot candle. It is apparent that very low currents would result when the transmissibility of a point being interrogated is low. For example, assuming transmissibility of a point on the film as .1, then the resulting current is in the order of .18 μ a. The use of a photomultiplier tube instead of the above would increase the current output. However the requirements for a high voltage power supply and the fragility of the photomultiplier tube would seem to justify amplification of the current output of the silicon diode. If all luminescent diodes of the array are interrogated by an identical amount of current then ideally the emitted light flux from any element in the array would be identical. Unfortunately the present uniformity of current efficiency is about $\pm$ 15%. Semiconductor manufacturers have indicated that substantial improvements in this uniformity can be expected. If the photodetector output is quantized into 16 gray levels the uniformity of current efficiency should be maintained at approximately $\pm$2-3%.

A further consideration regarding the luminescent array is its size. Typical center to center spacing is .018". For the 256 x 256 resolvable elements this yields a square array of about 4.6 x 4.6 inches. Certainly this is not directly compatible with 35mm film which has a frame size of roughly .825" x .600". The results of the simulation program indicate that a full resolution of 256 x 256 lines is not required for cloud pattern classification. Indeed resolutions in the order of 150 x 150 or less may suffice. This consideration reduces the size of the resulting array by about a factor of two. Additional size compatibility of the film and the array could be obtained by simple enlargement of the negative.

For systems which utilize either single or parallel hyperplanes for property filters it is not necessary for the arithmetic unit to form the product of $w_i e_i$. This product may be readily obtained

by applying a current through the i-th diode proportional to w_i, assuming only positive weights between 0 and 1 are employed. Negative weights are also possible using dual interrogation and sampling. However such procedures are not within the objectives of a general purpose system.

An alternate replacement for the luminescent diode array would be a flying spot scanner. The resolution of the flying spot scanner is comparable to that of good 35mm motion picture film. This is quite adequate for implementing the 256 x 256 array. This approach would require two digital-to-analog converters, one each for the X and Y axis, whereas the luminescent array may be accessed directly from digital information.

In the arithmetic unit it is assumed that negative numbers are represented in 2's complement form. The weights w_i and μ_i are initially written on the tape in 2's complement form if they are negative. The output of the photodetector is simply

$$e_i = K \cdot f_i t_i = C t_i \qquad 0 \leq t_i \leq 1$$

where
- t_i = transmissibility of the i-th element of the film
- f_i = flux radiated by an i-th diode
- K = photo-detector constant
- $C = Kf_i$ where $f_1 = f_2 = f_n$

Since e_i is between zero and C then the full scale encoding of the converter corresponds to the input where $e_i = C$. The sign bit is always a zero since each e_i takes on only positive values. The resulting conversion is a six binary bit positive number. However since the first bit is zero the input range from 0 to C is divided only into 32 increments. The maximum error of any conversion is $\pm$ 1/64. The arithmetic unit (shown in Figure 17) is capable of addition, subtraction, multiplication and division. The addition and subtraction is done in a parallel fashion and requires only one bit time, excluding the time required to access a particular storage register. Multiplication and division require n bit

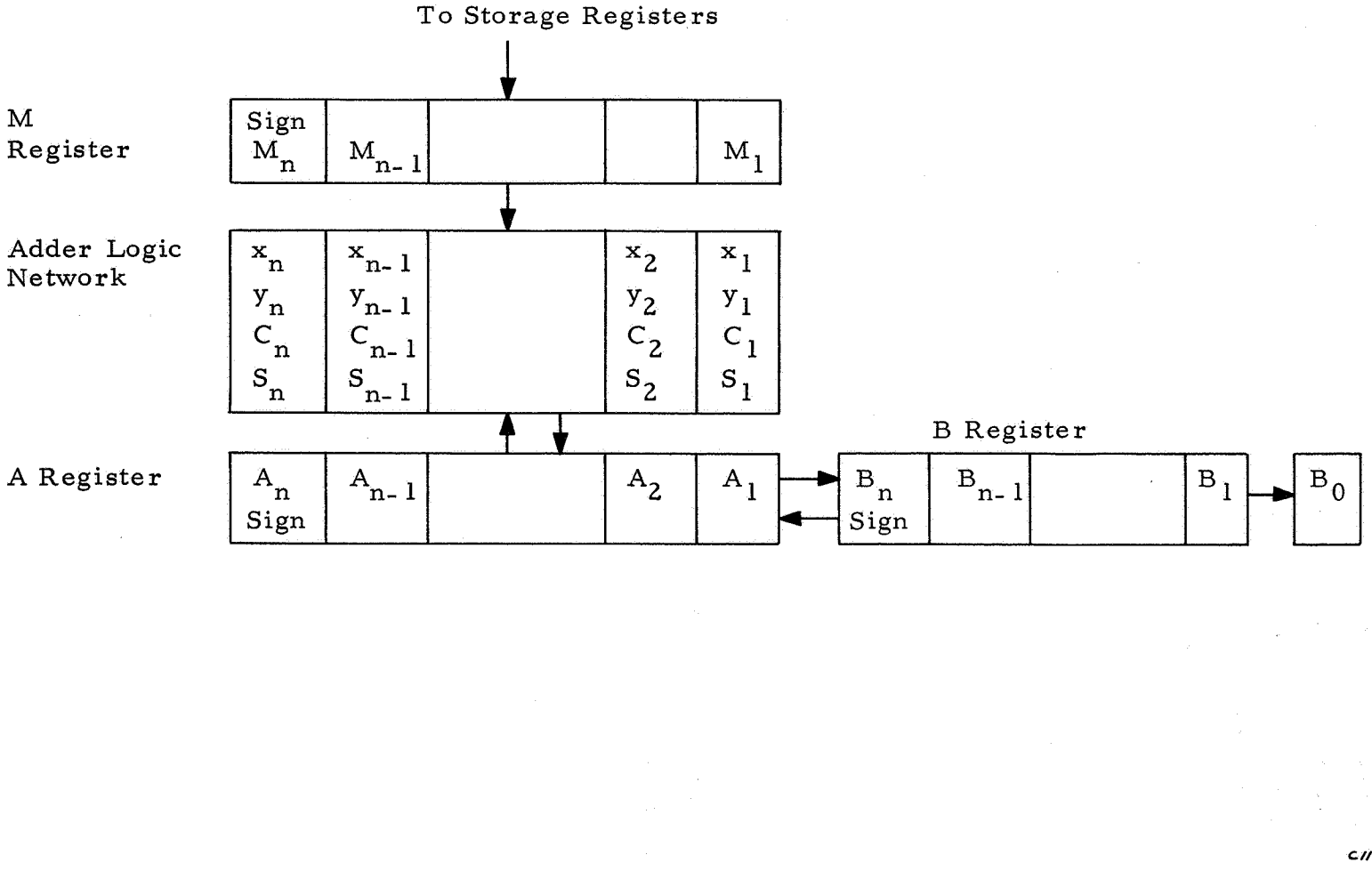


Figure 17. Arithmetic Unit

times where n equals the number of flip-flops in the M register or the A register. If subtraction is to be performed the 2's complement of M is added to A. This is accomplished by using the complement side of M as the X inputs to the adder and in addition forcing $C_1 = 1$. Since negative numbers are represented in 2's complement it is necessary to specify special rules for multiplication and division.

The programmer contains an instruction counter, an address counter and a variable bit time counter which is conditioned by the instruction being executed. At this time it appears that 32 instructions (5 flip-flops) are more than adequate for generating the algebraic expressions describing a property filter or response unit and for input/output control. A list of typical instructions is given in Table IV. The specific property filters and response unit used define the logic equations governing the programmer and the output logic.

The maximum estimated size of the storage register is 256 bits. The programmer, arithmetic unit (including storage register), array control, output logic, A/D converter and input tape buffer could be assembled on less than 72 plug in cards. This assumes that digital assemblies are in integrated form and each connector has 44 pins. The size of such an electronic black box would be in the order of 18 x 24 x 4 or roughly 1 cubic foot.

4.8 Summary of Results

This section gives a summary of the results and the conclusions on the design techniques drawn from the work described in Section 4. Additional conclusions, based on experimental investigations, are to be found in Sections 5.5 and 6.6.

1. Automatic pattern recognition is accomplished by representing a pattern as a set of numbers (a measurement vector), and then computing a classification for the pattern from these numbers. A series of transformations may be applied to the measurement vector in

TABLE IV	
Instruction List	
Instruction	Definition
1. Initialize	Clear M, A, B and ____ storage locations
2. Input	Put contents of e_i into the M register and _____
3. CLA	Clear the A register
4. LDA	Load ____ into A register
5. LDB	Load ____ into B register
6. LDM	Load ____ into M register
7. CAM	Copy the contents of A into M
8. CAB	Copy the contents of A into B
9. CTR	Transfer to labeled instruction if the content of A are <0
10. TRA	Transfer to label instruction
11. Add	Add M to A
12. Sub	Subtract M from A
13. Mult	Multiply B times M
14. Div	Divide M into A_n through B_1
15. LDC	Load converter output into storage register location ___
16. LDB	Load tape buffer output into storage register location ___ and into the array control registers
17. STA	Store A in location ____

the process of computing the patterns classification. Each time such a transformation is applied, a new measurement vector representing the pattern is acquired. The computation of the classification from this new vector is presumably easier. Such transformations may be regarded as a "preprocessing" step, or as part of the computation of a classification function. Very little is known about optimum preprocessing techniques and it appears that suitable preprocessing must be tailored to the specific data being analyzed.

2. Design techniques for decision mechanisms are characterized by the amount of information of the distributions of the measurement vectors for the various pattern classes which is known or assumed. When complete knowledge is assumed, the design technique becomes an analytic problem. When knowledge of the distributions is incomplete, a sample of classified patterns is used to estimate the missing information. The extent of the incompleteness assumed helps determine the number of sample patterns required, and the degree of "closedness" (feedback) in the design process. When the form of the distributions is assumed known, except for a limited number of parameters, a relatively few samples are required for estimation — the estimates are then substituted into a predetermined discriminant function. When little is known a priori about the distributions, a larger sample is required, and a common procedure is to accomplish the design by optimizing some function of the sample patterns (usually with some constraints, such as linearity, on the classification function. Minimization of the number of errors, or a loss function have been utilized. Since the design depends upon the performance of the decision mechanism on the sample patterns, such techniques are called "closed."

3. The design techniques used in this study attempt to introduce some of the advantages of an "open" technique (one in which strong distributional assumptions are made) into a technique which is perhaps more "closed" than any other suggested to date. This approach

is summarized as follows. Strong distributional assumptions are made (multivariate normal, with constraints on the means or covariances) and "optimum" discriminators determined for randomly selected subspaces of the measurement space. These discriminators with the multivariate normal assumption apply a threshold to a linear or quadratic function of the measurement vector. The decisions of these discriminators (called property filters or logic units) are used as coordinates for a transformed measurement vector. A linear decision function is derived from the transformed measurement vectors of the sample patterns with the aid of a loss function which reflects how well the samples are classified. Only the best of the logic units derived are used, the evaluation being in terms of the loss function. Individual pattern losses are used as weightings for the sample patterns, and new logic units are derived by discriminant analyses on the original measurement vectors, using the pattern weightings in deriving the statistical averages used. The best of these units (in terms of improvement in the loss function) is added to the set of property filters, and the linear discriminant on the augmented transformed measurement vector established. The process is repeated until all of the sample patterns are correctly classified. The loss function used will yield a linear discriminant that classifies all of the sample patterns correctly, whenever this is possible.

4. For the classification task at hand (the separation of frames containing vortices from nonvortex frames) it can be shown that the means of the two distributions are equal, and that the components of the mean vectors are all equal. In this case, discriminant analysis with the normality assumption gives quadratic input threshold units for the property filters. These units are extremely difficult to implement in analog hardware, and require relatively large amounts of memory in digital implementations. These units are optimum under the normality assumption. The analysis is identical with one suggested by Fisher in 1936 as a heuristic approach for arbitrary distributions.

5. An approximation to the quadratic input units has been derived. A principal axis of differentiation is computed from the covariance matrices. Logic units using this principal axis solution are linear input devices with two thresholds. They are easily implemented in analog hardware, and require relatively small amounts of memory in digital implementations. Their performance is investigated in Section 5. As with the quadratic units, principal axes solutions have been suggested in connection with other than normal distributions.

6. Invariance to changes in brightness and contrast of the patterns is desirable, as vortex patterns are differentiated from ordinary cloud cover by form. A single normalization of the pattern is not satisfactory as it does not compensate for variations within the input field such as those caused by the extent to which the horizon is visible. Techniques for making the individual logic units invariant to linear changes in the gray scale have been devised. The restrictions on the logic unit functions for invariance have been developed, and changes in the discriminant analysis to ensure that these restrictions are met have been worked out. For three of the four logic unit generating techniques, these changes are equivalent to contrast and brightness normalizations on the sample patterns. The resulting logic units have the same form as their noninvariant counterparts (linear or quadratic input threshold units). For the fourth technique the linear input dual threshold unit described in 5 above, the form of the unit also changes. The resulting logic unit has a right circular cone for a switching surface (more precisely, a cylinder set with a right circular cone for a base). This conic unit presents the same difficulties in an analog hardware representation as does the quadratic unit, but on a more limited scale. Digital simulation remains easy, however. The costs and benefits of the invariant units are investigated in Section 5.

7. Mechanization of the simple discriminant functions, perpendicular bisector and the oriented hyperplane, can be readily accomplished in integrated circuit form. The parallel hyperplane approximation to the quadratic surface unit can also be simply constructed. However,

the normalized parallel hyperplane and the quadratic surface units require squaring and multiplication circuitry. Present state of the art of the required components results in a relatively large, expensive circuit with limited accuracy and precludes the development of networks of such units.

8. The resolution required for analysis of cloud photographs can readily be accomplished using storage tubes such as a flying spot scanner. The scanner can also be used as a temporary medium for storing the image. Digital scanning circuits allow addressing any location on the image in conjunction with a logic unit input connection. An array of light emitting diodes can also be used to interrogate any resolvable element on the film image and project the resulting illumination onto a light sensing device for computing the weighted average of a point in the image. These techniques illustrate that input sensors for the recognition tasks are available despite the inherent high resolution required for photographic image processing.

9. Preliminary design considerations have been given to a general purpose digital sequential machine for processing cloud photographs. The system utilizes the results of the simulation program to generate a tape of property filter parameters. Different types of property filters and different recognition systems can be accommodated by preparation of a new tape. The preliminary design of the associated conversion, memory and arithmetic units demonstrates that such a system is within the state of the art and can be constructed as a compact unit. Several input sensors have been considered as mentioned in 8 above.

5. STUDIES ON ALPHABETIC CHARACTERS

5.1 Topics

A sample problem, the classification of hand-printed alphabetic characters, has provided the framework for an investigation of certain aspects of the design technique. This simplified situation permitted a more economical study than would the classification of TIROS frames. Because of the simplicity, however, certain problems could not be treated. These problems will be discussed in Section 6.0 where the analysis of the satellite photographs is presented.

The subjects most suitable for study in a simplified situation are those that are related to the design technique itself, rather than to its application. A number of these design technique-oriented studies are described in this section. In addition, several studies which are specific to the given sample problem are also discussed. In these cases, special care is taken to limit the conclusions drawn.

The topics, and sections describing the results, for which the sample problem seems most applicable were:

1. Loss function characterization of the problem (5.4.1) — Does the loss function value provide a usable measure of how well a partially designed network deals with the sample patterns and with a generalization sample?
2. Random variability of network design (5.4.1) — Because the subspaces are selected randomly, several design runs will produce several different networks. How great is the performance variations of these networks?
3. Design technique parameters (5.4.2) — The parameters considered are the number of iterative cycles in the relaxation process for minimizing the loss function, and the half-angle of the right circular cones which form the gray invariant version of the parallel hyperplane technique.

4. Gray invariance effects (5.4.3) – The advantage of gray invariant logic when brightness and contrast changes occur was tested, and the increase in the network size was assessed.

5. Technique complexity effects (5.4.5) – The Oriented Hyperplane and Perpendicular Bisector techniques exploit the same pattern distribution differences, the latter providing a simpler design calculation. Similarly, the Quadratic Surface and Parallel Hyperplane techniques exploit similar differences in the distributions, the latter providing simpler implementation. The cost of the simplifications was investigated.

6. Sensitivity (5.4.6) – Unlike error correcting procedures, the loss function approach attempts to place the decision boundary as far as possible from the sample patterns. The sensitivity of several network designs to random perturbations of the input weights (to both the logic units and decision units) was tested.

In addition to these studies, the following topics were investigated. The results are felt to be more strongly dependent on the particular sample problem.

1. Design technique parameters (5.4.2) – Two parameters which are more dependent on the particular problem, are the selection ratio* (since the probability of finding good units is dependent on the complexity of the problem) and the number of input connections per logic unit.

2. Distributional assumptions effects (5.4.4) – The Oriented Hyperplane and Perpendicular Bisector

*The ratio of the number of logic units considered to the number actually incorporated in the network.

techniques exploit differences in the centroids of the distributions, the Quadratic Surface and Parallel Hyperplane techniques exploit differences in the covariance matrices. While the suitability of the technique depends on the actual problem, it is shown that the effects appear primarily in the generalization capabilities.

Two additional studies, a correlation analysis of the sample patterns (5.2.4) and an application of standard perceptron techniques (5.4.7), are included to provide a means for evaluating the discriminant analysis-iterative design approach.

5.2 The Sample Problem

5.2.1 The Design Techniques

A detailed description of the design techniques used in this sample problem was given in Sections 4.2 through 4.5 A summary is given here.

Eight variations of the basic discriminant analysis-iterative design approach are used. In addition, several standard perceptron methods are used in a limited study to provide a basis for comparison. The eight variations differ in the means by which logic units are generated, using discriminant analysis.

The discriminant analysis-iterative design approach is based on a loss function. Each sample pattern is assigned a loss number which is a function of the value of the input to the decision element. When adjusted for the decision element threshold, a loss number less than one corresponds to a correct decision for that sample pattern, and a loss greater than one to an incorrect decision. The loss numbers vary exponentially with the input to the decision element. A total system loss is defined to be the sum of the loss numbers of all of the sample patterns.

Logic units are added to the network one at a time. Each unit added is selected from a pool of candidate units, the selection

being based on the criterion of greatest reduction in the total system loss. To simplify the calculation, the output weights of all units already selected are held fixed during the selection process. At the time of selection, an output weight is assigned to the new unit, again based on minimizing the system loss subject to the other output weights being held fixed.

Once a new unit is added to the network, the output weights of all of the units are readjusted to minimize system loss. This is accomplished by means of a relaxation process, adjusting each weight in turn while holding the remaining ones fixed. A fixed number of cycles through the weights are used (three, except where the effects of varying this parameter are studied), and then the addition of a new unit is considered.

Each time a logic unit is to be selected, a new pool of candidate units is created. This is accomplished by performing a discriminant analysis on each of a number of subspaces. Each subspace is selected randomly, by choosing a fixed number of sensory field points (six points, unless this parameter was specifically varied) with no restriction other than to insure that any given subspace had no duplicate coordinates.

Four techniques for generating logic units on a subspace using discriminant analysis were described in Sections 4.3 and 4.5. These were named Oriented Hyperplane, Perpendicular Bisector, Quadratic Surface, and Parallel Hyperplane, the names indicating the nature of the logic unit switching surface in the subspace. These techniques are summarized in Table V. In addition to these four techniques, four variations which produce logic units invariant to changes in brightness or contrast in the input are described in Sections 4.4 and 4.5. These variations are given the same names with the words Gray Invariant appended. Thus there are a total of eight methods for generating pools of logic units.

TABLE V

TECHNIQUE COMPARISON

Technique	Difference Exploited	Remarks
Oriented Hyperplane	Mean Vector	Linear Input Threshold Unit
Perpendicular Bisector	Mean Vector	Same – Easier Computation
Quadratic Surface	Covariance Matrix	Quadratic Input Threshold Unit
Parallel Hyperplanes	Covariance Matrix	Linear Input – 2 Threshold Unit

In deriving, from the sample patterns, the statistical averages (estimates of the mean vectors and covariance matrices) required by the discriminant analyses, the loss numbers are used to weight the contributions of the individual patterns. This insures that those patterns which are most poorly classified by the partially designed network are counted most heavily in designing the next logic units. The pool of candidate units is thus kept responsive to the remaining problem.

5.2.2 Sample Patterns

Twelve alphabetic characters — A, D, E, H, I, L, N, O, R, S, T, and U were selected. With the exception of U, these are the most common letters in English text. Hand-printed examples of these served as sample patterns.

A set of 240 sample patterns was acquired by having 20 individuals provide one sample of each character. Seventy bit patterns were produced by superimposing a seven-by-ten grid, and assigning to each square a "1" or "0" depending upon whether or not any part of the sample letter intersected the square. The resultant patterns are reproduced in Figures 18 through 22. The individuals were requested to produce reasonably sloppy characters, and it can be seen that most of them succeeded admirably. This set of patterns is designated Deck 1 and was used as the sample patterns from which the recognition network logic is derived.

A second set of patterns was obtained by selecting 20 more individuals and repeating the process. The 240 patterns thus obtained were designated Deck 2. Deck 2 serves as an independent set of sample patterns against which networks designed on Deck 1 may be tested. Deck 2 is shown in Figures 23 through 27. The Deck 2 characters appear to be at least as sloppy as those of Deck 1. Both pattern decks are available on punched cards, magnetic tape, and punched paper tape.

Four other decks of patterns have been derived from the original two decks. Two of these represent a brightness and contrast

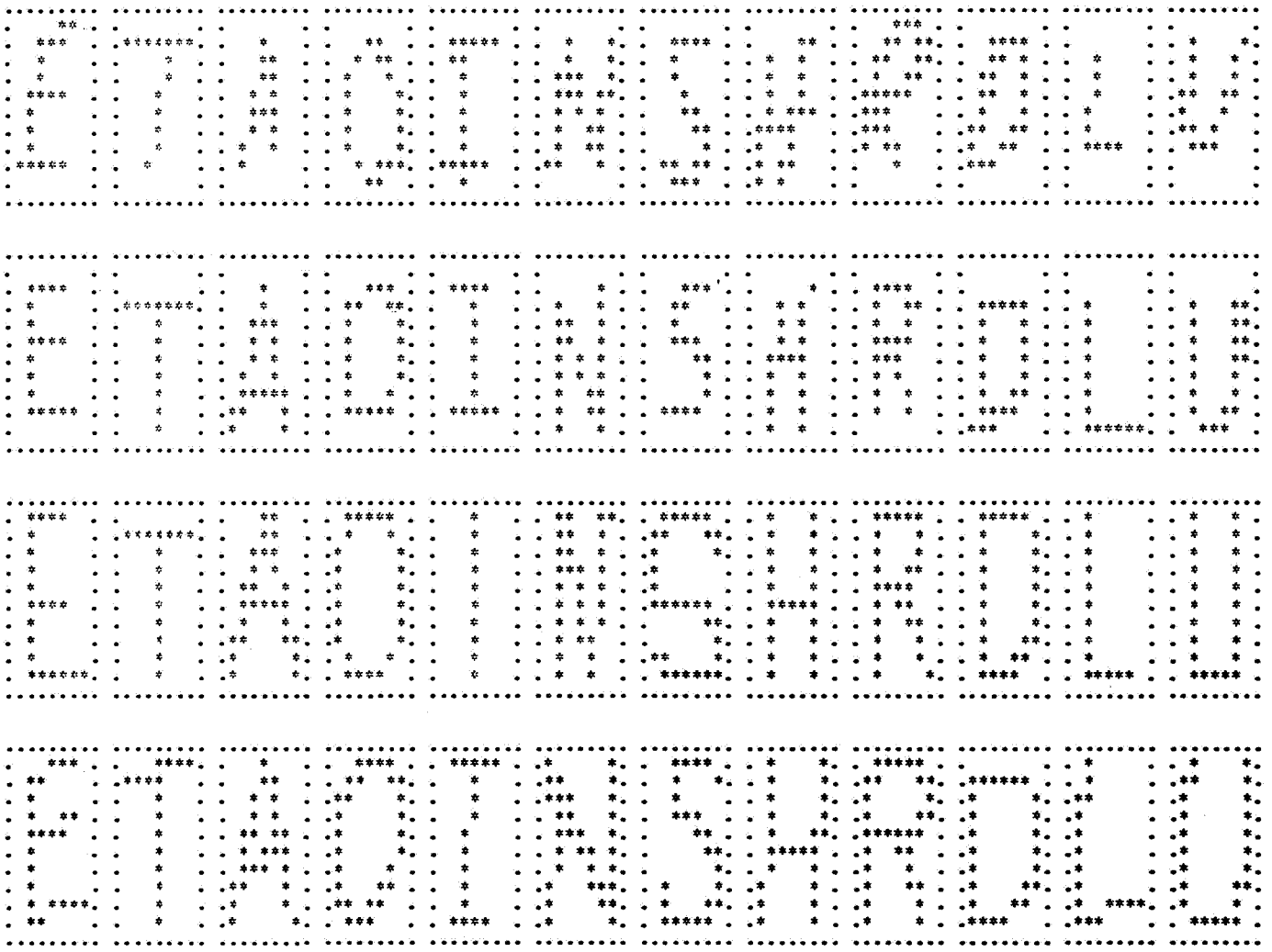


Figure 18. Deck 1 Sample Patterns

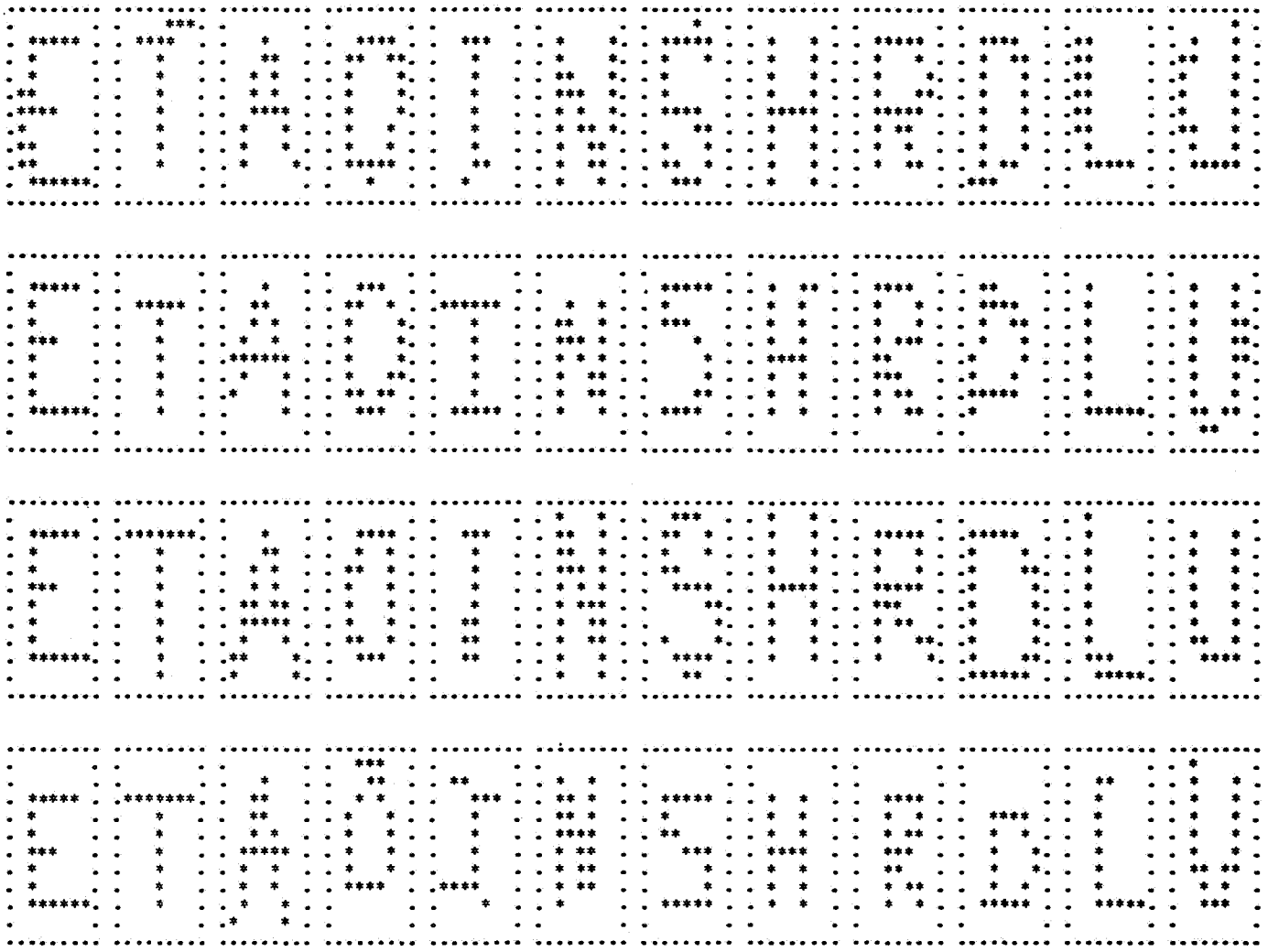


Figure 19. Deck 1 Sample Patterns

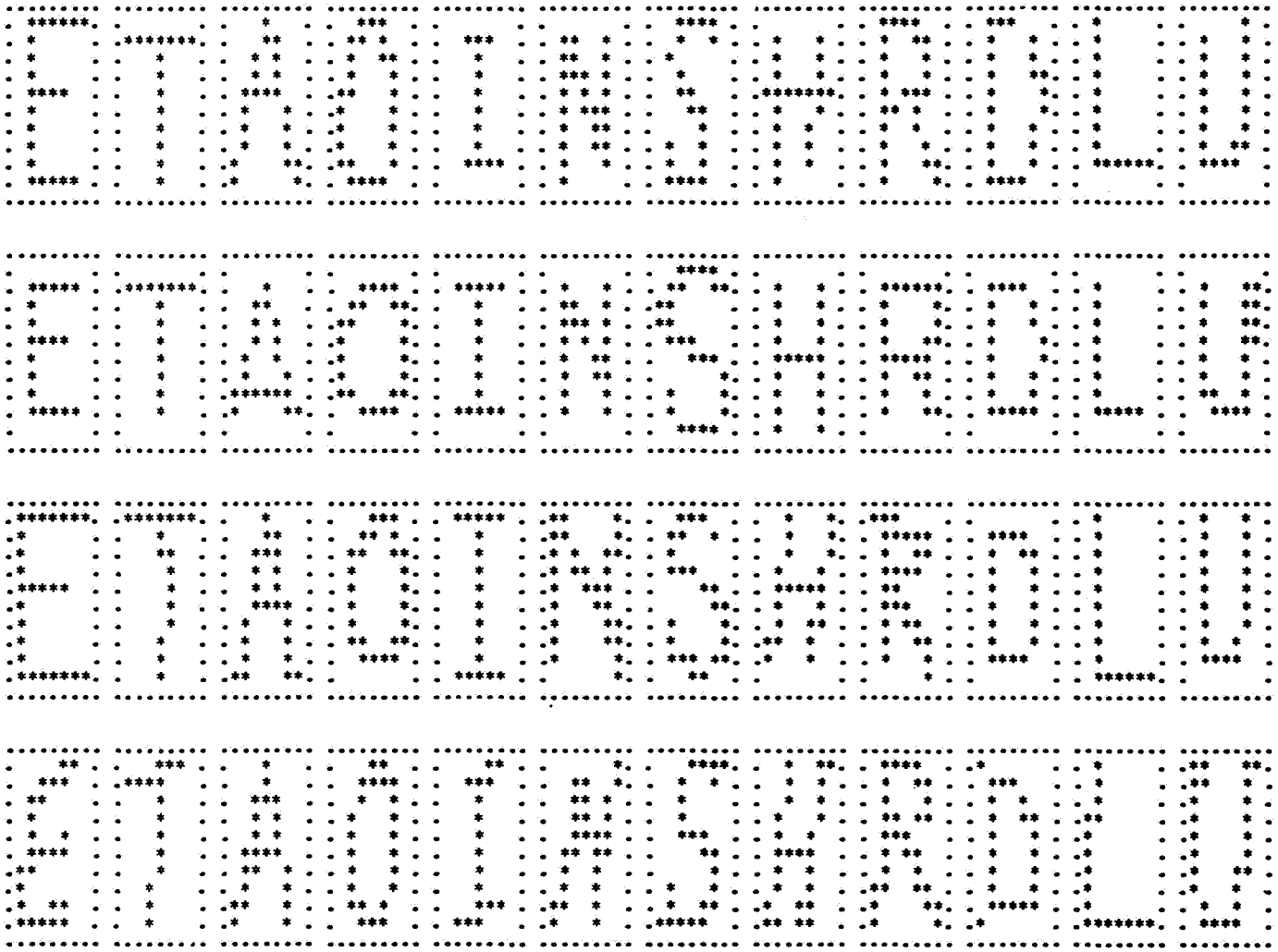


Figure 20. Deck 1 Sample Patterns

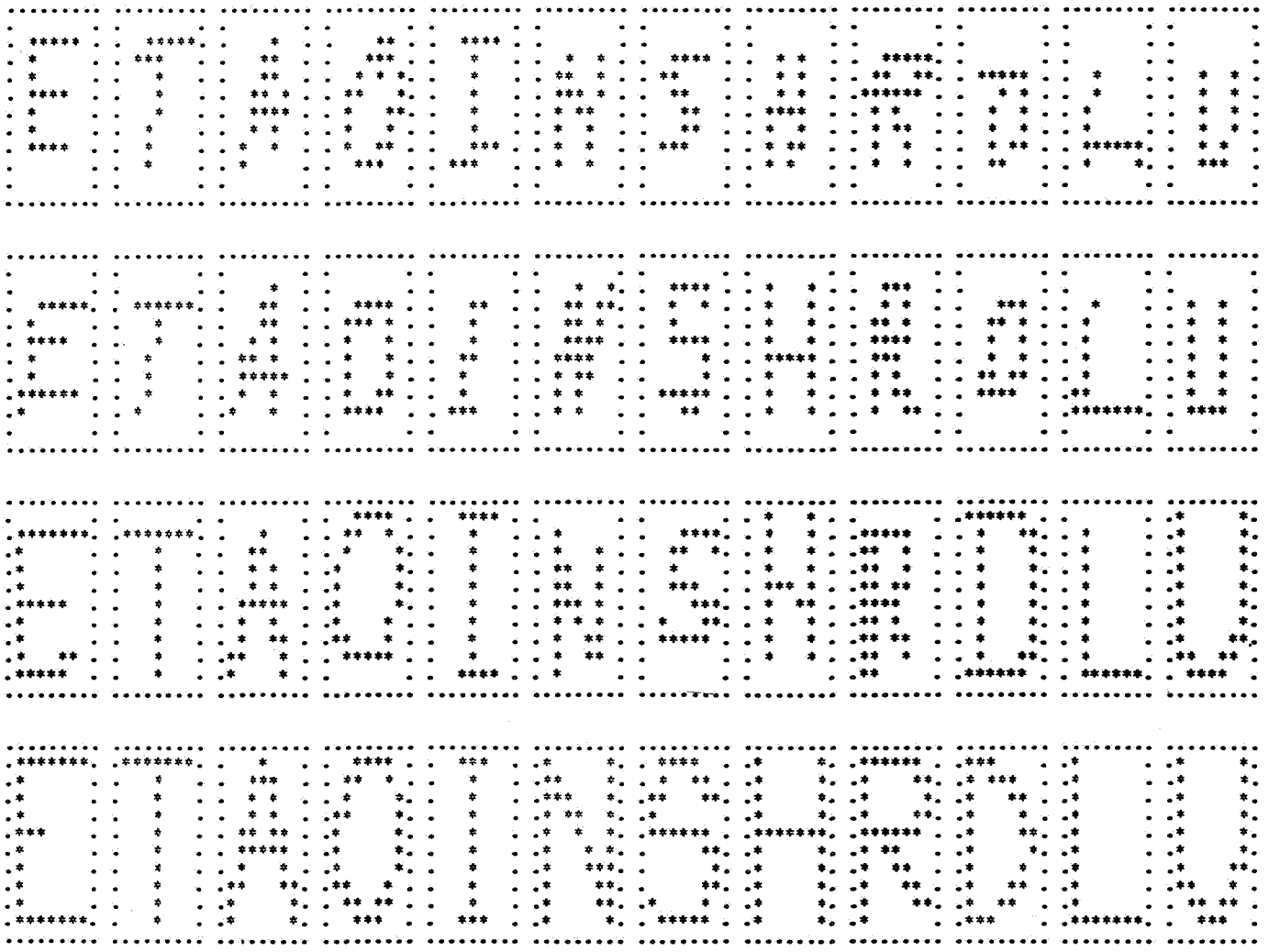


Figure 21. Deck 1 Sample Patterns

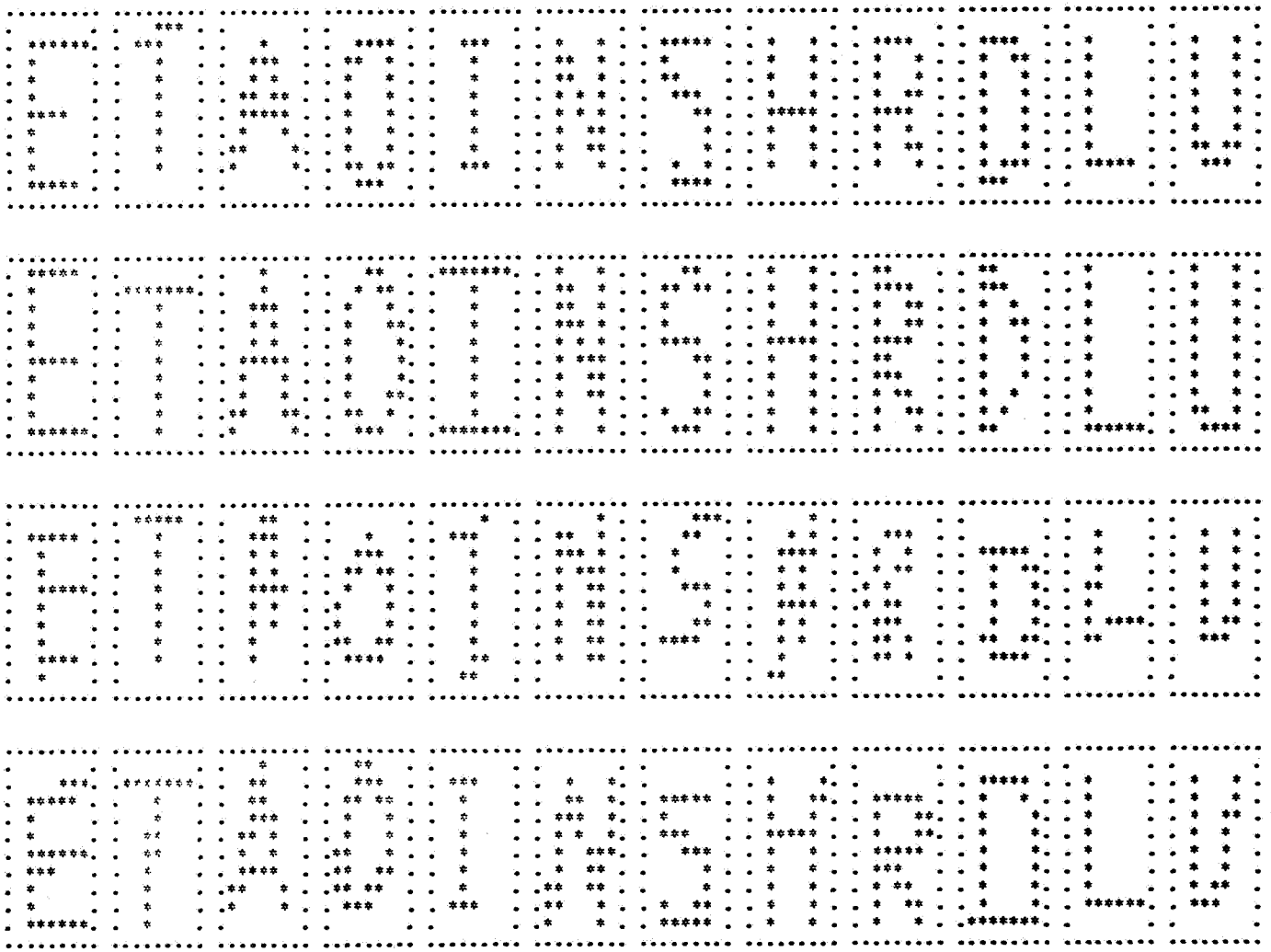


Figure 22. Deck 1 Sample Patterns

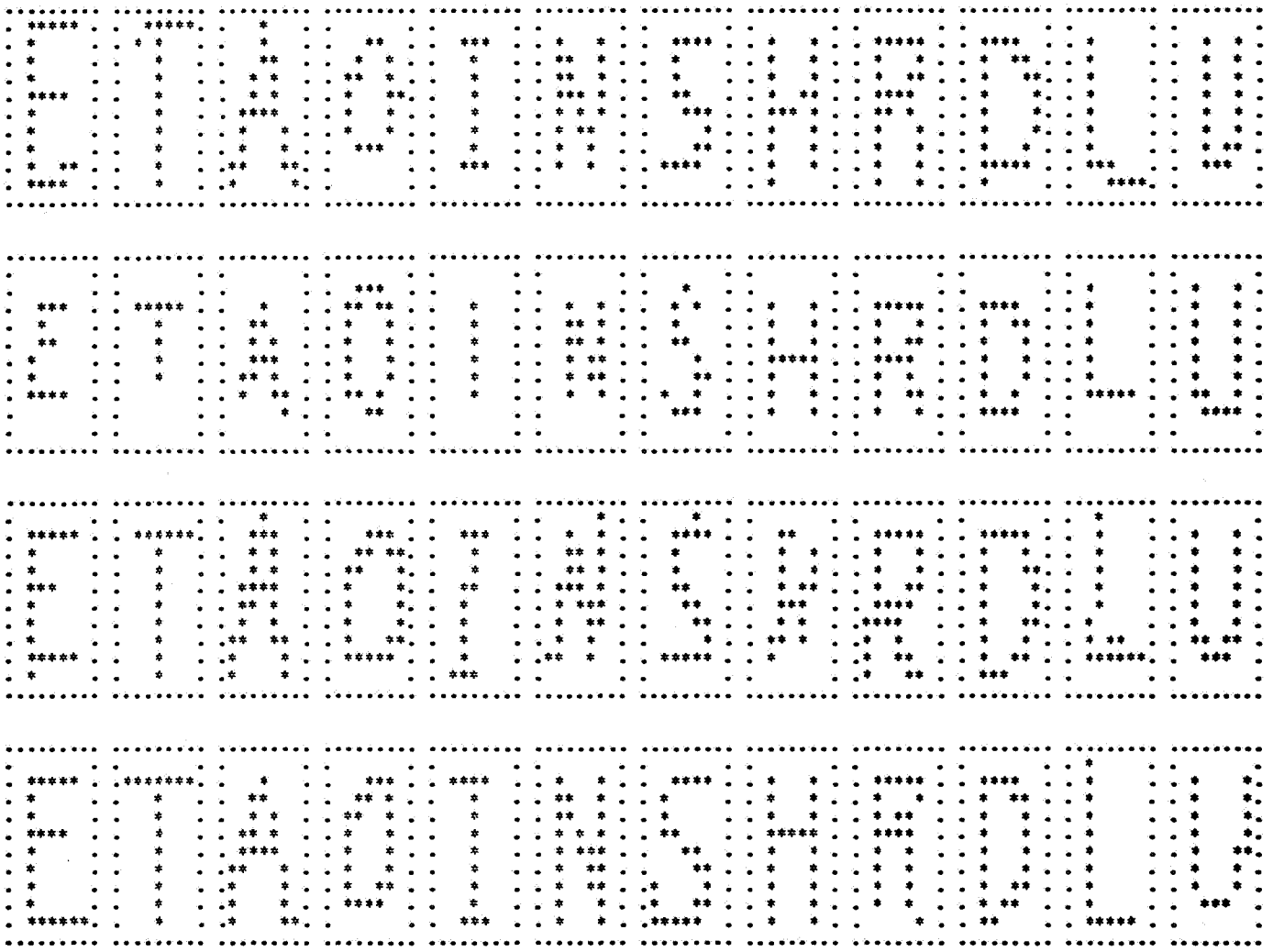


Figure 23. Deck 2 Verification Patterns

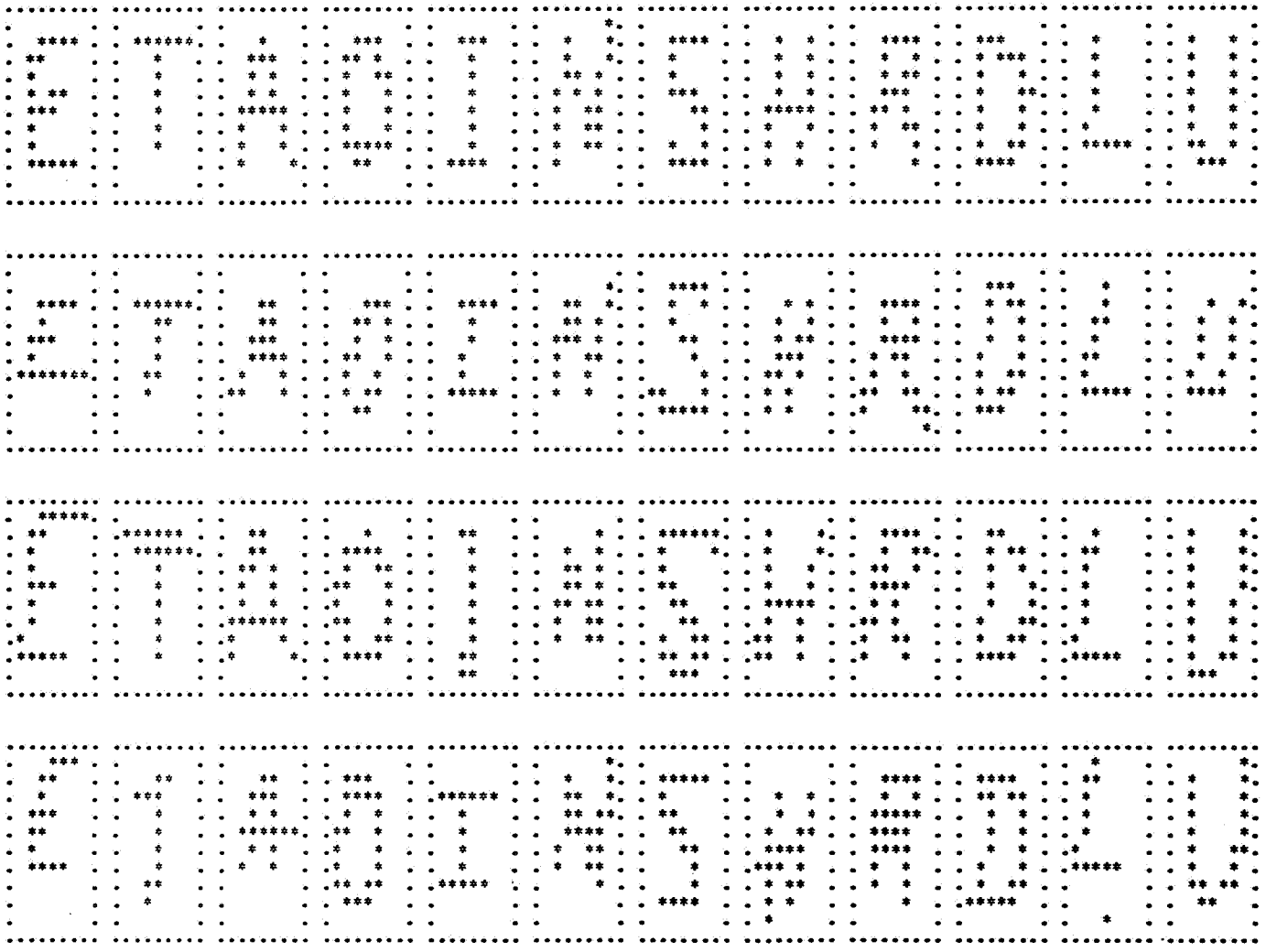


Figure 24. Deck 2 Verification Patterns

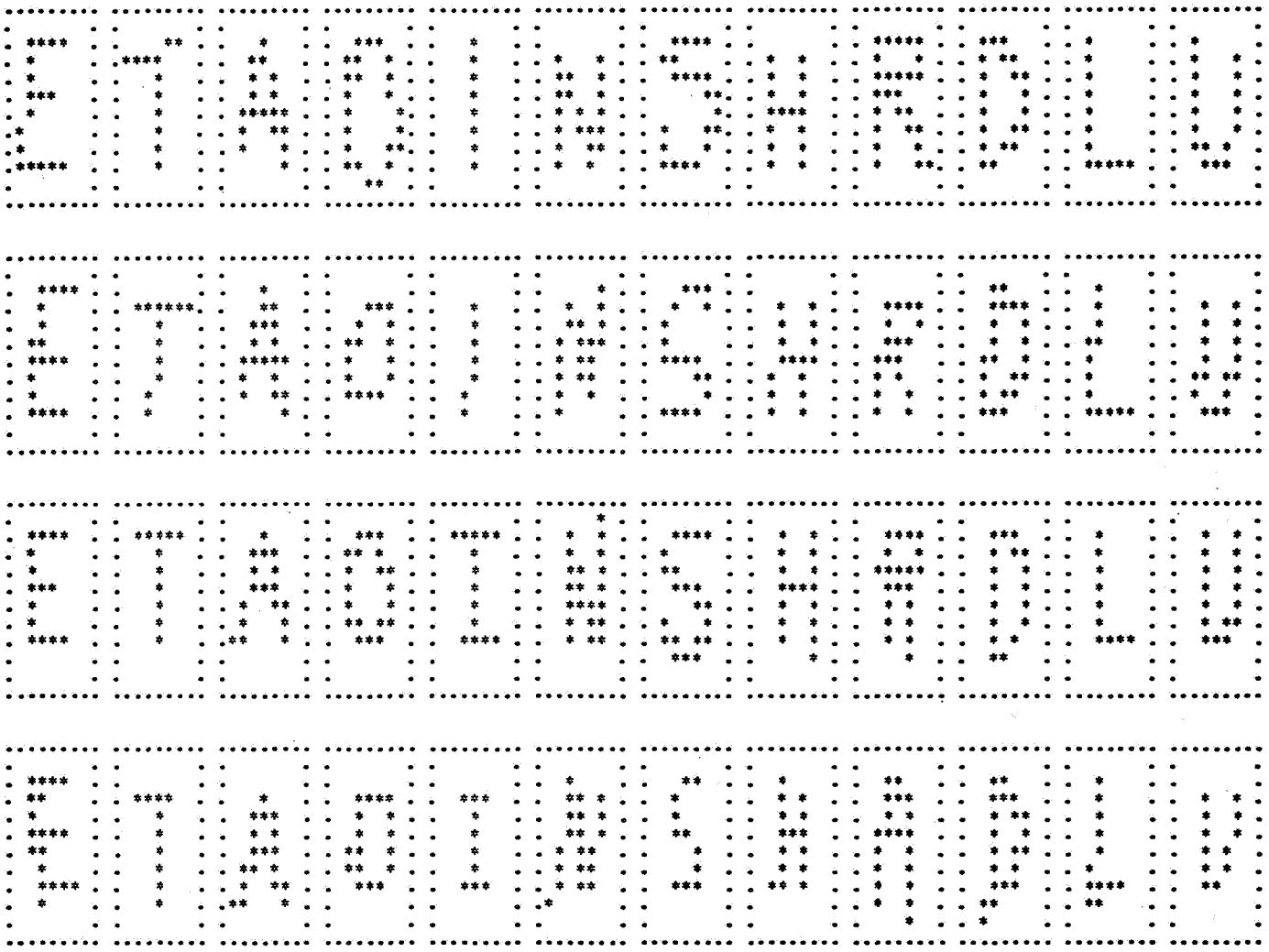


Figure 25. Deck 2 Verification Patterns

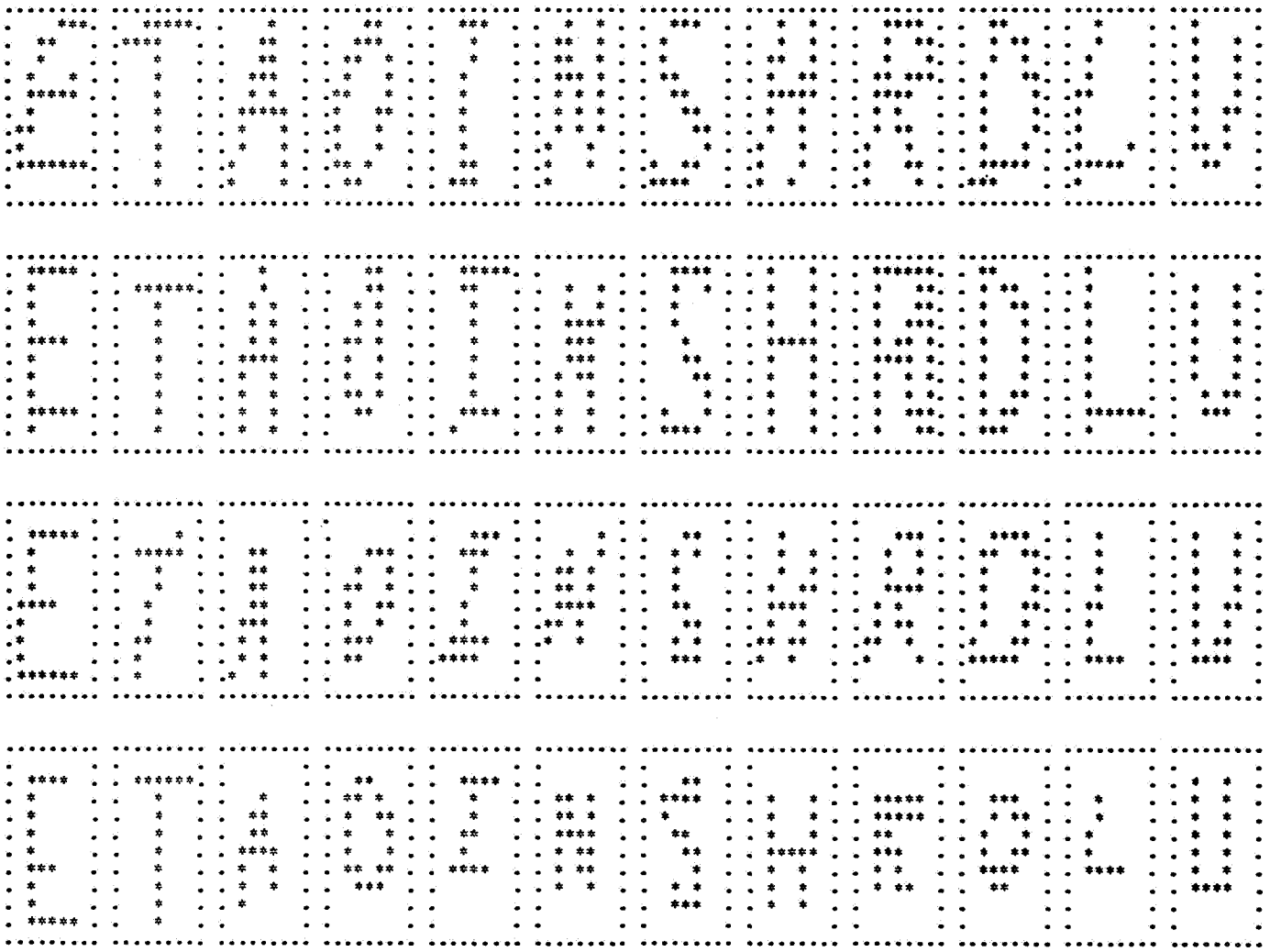


Figure 26. Deck 2 Verification Patterns

C0612

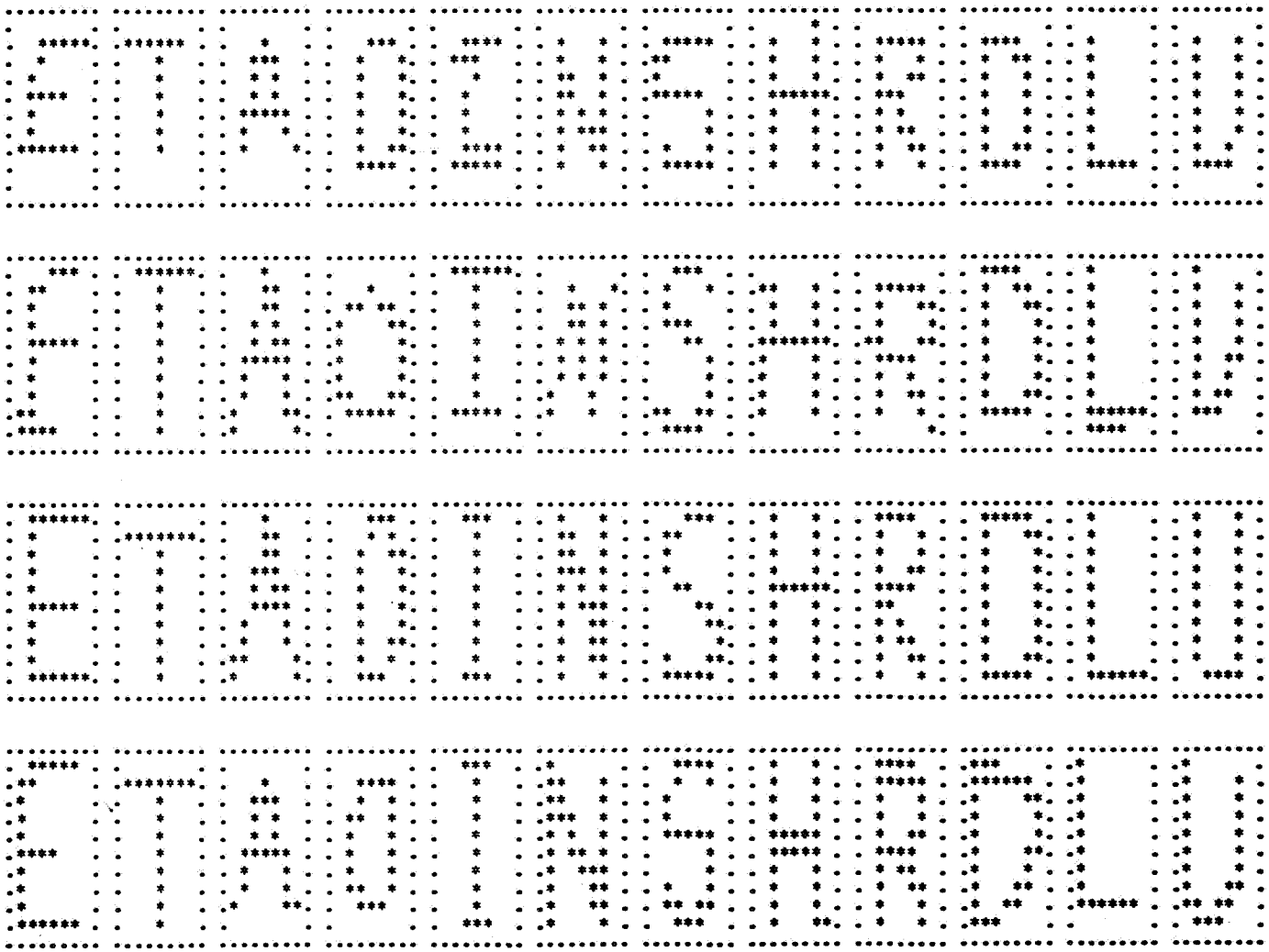


Figure 27. Deck 2 Verification Patterns

change from the basic decks in which the black level is represented by 0.7 rather than 1.0 and the white level by 0.3 rather than 0.0. The remaining decks use a sliding change in the scale in which the black is represented by 1.2 and the white by 0.0 on the left side of the input grid, varying gradually to 1.8 for black and 0.6 for white on the right side. The performance gray invariant versions are not affected by the first change, but are not invariant to the second change.

5.2.3 The Sample Task

The twelve characters were divided into two equal classes, and recognition networks were designed to solve this two class problem on Deck 1 using the eight design techniques. Four such binary classifications were selected originally, to provide complete separation of the characters. Early runs indicated that two of these classifications were difficult for the recognition networks, and two were easy. One of the easy classifications and one of the difficult ones were retained. Most of the study was performed on the easy classification, with a skeleton set of runs on the hard classification to show that the conclusions derived from the easy classification hold. The easy classification placed the letters D, E, L, O, S, and U in one class, and A, H, I, R, N, and T in the other class. The harder classification put A, D, E, I, L, and R in one class, and H, N, O, S, T, and U in the other.

A standard set of parameters was selected, based on the early runs. These were varied one at a time in subsequent studies. The standard procedure calls for 20 logic units to be generated for each one selected, for three iterative cycles following the selection of each logic unit, and for six input connections per logic unit.

FORTRAN II computer programs were used for generating the recognition networks. Each time a logic unit was added to the network the computer produced, as output, the logic unit specifications, the unit output weight and system loss prior to the iterative cycles, the system loss and the output weights of all units after the iterative cycles, the percentage decrease in the loss function due to the addition of the unit

and the iterative cycles, the average decrease over all units, the pattern losses for all 240 sample patterns, the response unit threshold giving minimum system loss and the number of classification errors on the sample deck using this threshold, a threshold range and the number of errors when the threshold was selected to give the minimum number of errors.

Due to memory restrictions, the program would stop when 20 logic units were included in the network. The program would also cut off if there were no errors with either threshold. The restriction to 20 logic units was not a great handicap, because only about five percent of the machine designs occurred in which the minimum number of errors was not zero within 20 units.

The minimum number of errors, the system loss, the average rate of decrease in the system loss, and the number of generalization errors against various other pattern decks have all been useful means for comparing the machine designs.

5.2.4 Correlation Analysis of the Sample Patterns

In the design of a recognition machine, the most important measure of performance is how well the machine "generalizes," that is, how well it recognizes new patterns. In this problem, generalization results must be interpreted carefully, as the data are critically dependent on how well the patterns of Deck 1 represent those of Deck 2. Since only 20 examples of each character are used, and since the characters are quite sloppy, generalization measurements may be expected to show significant numbers of errors.

The importance of the generalization measurements has led to two investigations designed to shed light on the similarity of Decks 1 and 2. One investigation involved the design of recognition logics by standard perceptron techniques to provide a comparison to the current processes. The results are given in Section 5.4.7. The second study involved the correlation of each pattern in each deck against each pattern in both decks.

The following recognition scheme was considered. All of the patterns of one deck were stored in memory as prototypes. When an unknown pattern is to be classified, it is correlated against each of the patterns in this deck. The unknown pattern is declared to be the same as the prototype giving the highest correlation coefficient. In four studies, both decks were used to provide the prototypes, and both decks were used to provide the unknown patterns. When the same deck served both purposes, the highest correlation coefficient was discarded (since it is the pattern correlated against itself), and the second highest coefficient used for classification.

The results of these studies are shown in Table VI. The results in the second column are of the greatest interest. It can be seen from the table that the patterns of Deck 2 (used as the generalization deck in later studies) are more irregular, as they give higher error rates as unknown patterns. In these cases the decision among the tied values was made randomly. It should be noted that this correlation scheme requires a large amount of memory for storing the paradigms and a substantial amount of computer time for obtaining the correlations, a situation which would become considerably worse with the high resolution Tiros frames.

Several other classification schemes were tried, such as considering the highest average correlation against all prototypes of a letter or class, but were dropped when they proved to be inferior to the above scheme. They would, however, require less memory.

5.3 Computer Programs

Two computer programs accounted for the bulk of the computing on alphabetic characters and will be discussed here in some detail. Nine other programs, used to lesser extents, will be described more briefly.

The primary program, called DAID, controlled the design of networks to classify the alphabetic characters. It is written mainly in FORTRAN II, and consists of a main program and eight subroutines.

TABLE VI

GENERALIZATION ERRORS USING HIGHEST INDIVIDUAL CORRELATION CLASSIFICATION SCHEME

Paradigm Deck	1	1	2	2
Unknown Pattern Deck	1	2	1	2
Task				
Separate 12 Characters	45-1/2	61-1/6	49-1/2	62
Separate DELOSU from AHINRT	11	13	9	10-1/2
Separate ADEILR from HNOSTU	29	34-1/3	35-1/2	33-1/2

The main routine is concerned primarily with the inputting and outputting of information, and with the sequencing of the subroutines. Two versions of the control program were necessary since the Quadratic Surface techniques required enough extra memory that one table in which the activity vectors of the logic units were recorded had to either be stored in packed form, or put onto magnetic tape. The latter course was chosen, and was therefore enough slower to warrant a separate control program for the other three techniques.

A flow chart of the main routine is presented in Figure 28. The main routine can accept sample patterns from cards or from magnetic tape. The options or network parameters are

Number of dimensions in signal space (up to 70)

Number of groups of patterns (up to 12)

Number of samples for a group (up to 20)

Maximum number of logic units per network (up to 20)

Number of candidates considered for each unit chosen (up to 50)

Number of input connections per logic unit (up to 50)

Number of iterative cycles each time a unit is added (unlimited)

The program thus provides far more flexibility than was actually utilized. The program provides a printout of the network values after each logic unit is added, along with a specification of how well the partial network works on the sample patterns, and an evaluation of the last unit added. The partial network specifications are also recorded on magnetic tape, and, optionally, on punched cards. The punched card output was actually used for the permanent record. The network design is terminated whenever a partially designed network classifies all of the sample patterns correctly.

Subroutine SUB generates subspaces for the logic units. The input connections are chosen randomly, and without replacement for a single logic unit. The selections for different logic units are independent.

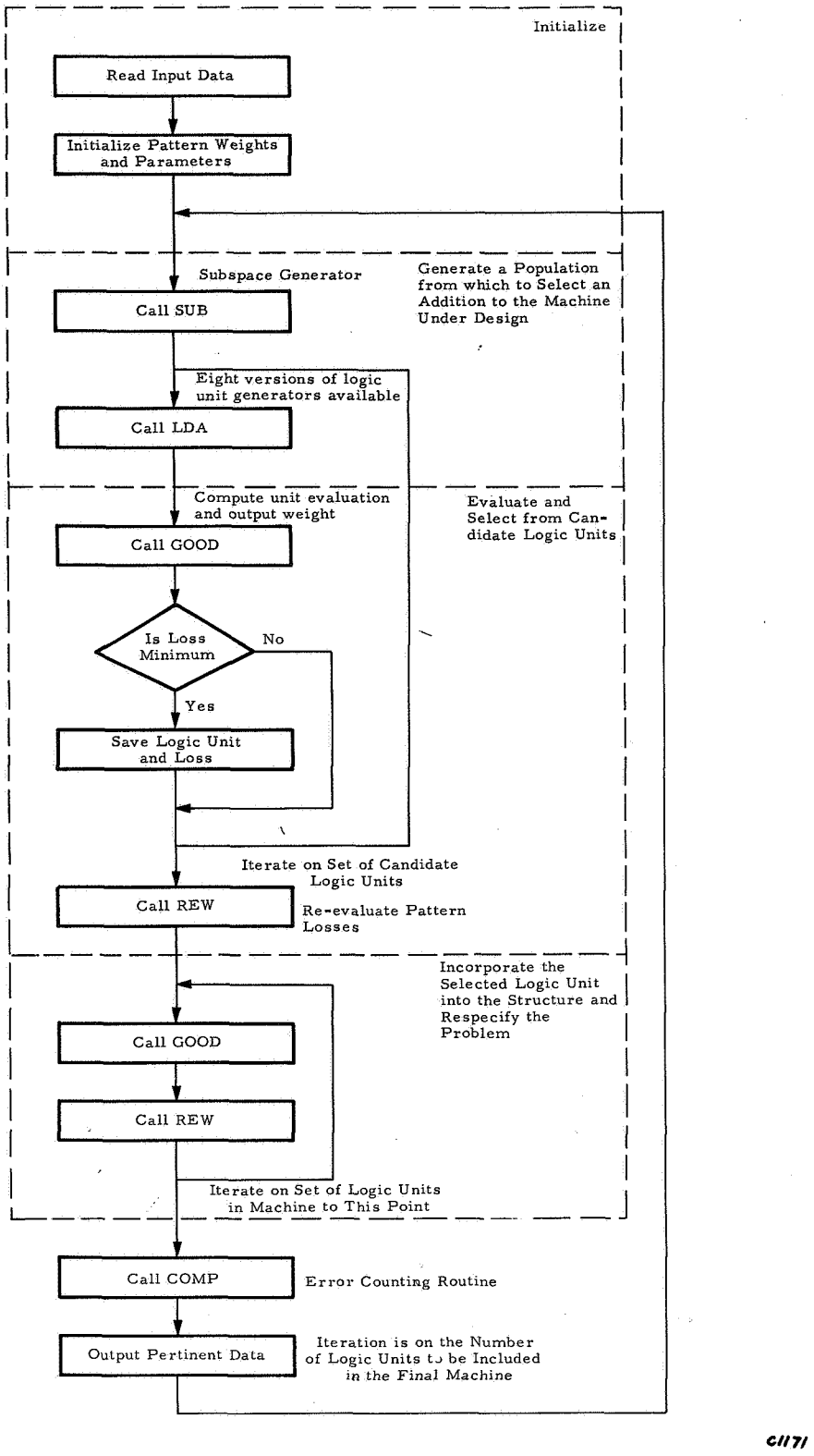


Figure 28. Flow Chart for Main Program

Subroutine LDA generates a logic unit for a given subspace. It provides the estimates of the mean vectors and the covariance matrices, and analyzes these to provide the unit specification. It also computes an activity table specifying for which sample patterns the logic unit is active, so that except for storing and outputting the machine designs, the remaining program is independent of the nature of the logic unit decision boundaries. Eight versions are available, one for each of the four design techniques and one for each of the gray invariant counterparts described in Sections 4.3, 4.4, and 4.5.

Subroutine GOOD evaluates a logic unit based on the activity table and the sample pattern loss table. It provides the increment to the output weight of the unit to minimize the system loss, and the minimum value which would be obtained.

If the change is accepted, subroutine REW updates the pattern losses and computes an increment to the decision element threshold. The losses are again modified for this increment.

Subroutine COMP analyzes the table of pattern losses to determine the number of sample patterns misclassified. It also computes the smallest number of errors occurring if the decision element threshold is varied.

Three additional subroutines that do not appear on the figure were used. Subroutine FEVV is the share routine EVV modified to be compatible with Fortran II and the 7094. EVV was written for the 704, but proved to be the fastest eigenvector routine of those available for a symmetric matrix. It can extract the eigenvalues and eigenvectors of a 10-by-10 matrix in under one-half second. This speed was essential for the program described in Section 6.3. Subroutine TRAN is a trivial routine inserted to correct an error in the Fortran compilation of the main routine, and subroutine DM9SCH is required by the monitor to obtain punched card output.

The second major program is called VERPR. It takes the partial network specifications produced by DAID, and applies each

partially designed network to a set of test patterns. The test patterns must be of the same dimensionality as the design patterns, but there may be any number up to 240 patterns. The program produces a sequence of performance specifications for each partial network leading to the final network, specifying the total input to the decision element for each test pattern, and the minimum number of errors obtainable by adjustment of the decision element threshold.

Two versions of this program are available – a Fortran IV version for the IBM 7094, and a Fortran II version for the SDS 930. The former version was used to test the networks against Pattern Decks 1 and 2 (tests against Deck 1, the design deck, were necessary, since the card punch was not that reliable). The latter version was used to test the machine designs against pattern decks with modified gray scales. The latter version was more convenient for this purpose, since it had a provision to modify the gray scale internally in the program. A flow chart of this program is given in Figure 29.

In addition to the two programs described above, a number of other routines were utilized on a more limited basis. These are listed below with a brief functional description.

1. CORR – This program stores a deck of patterns in core, then correlates incoming patterns against each of the stored patterns. The program ranks the correlation coefficients obtained, and ranks the letters according to the correlation coefficient for any example of that letter with the unknown pattern.

2. PERC – This program designs decision networks using several standard perceptron techniques. It can generate logic units randomly, or by the Perpendicular Bisector technique, or it can accept logic units generated by the DAID program. It can assign output weights to these units by the "forced learning" rule or by the "logarithmic weight" rule (see Section 5.4.7). The output of this

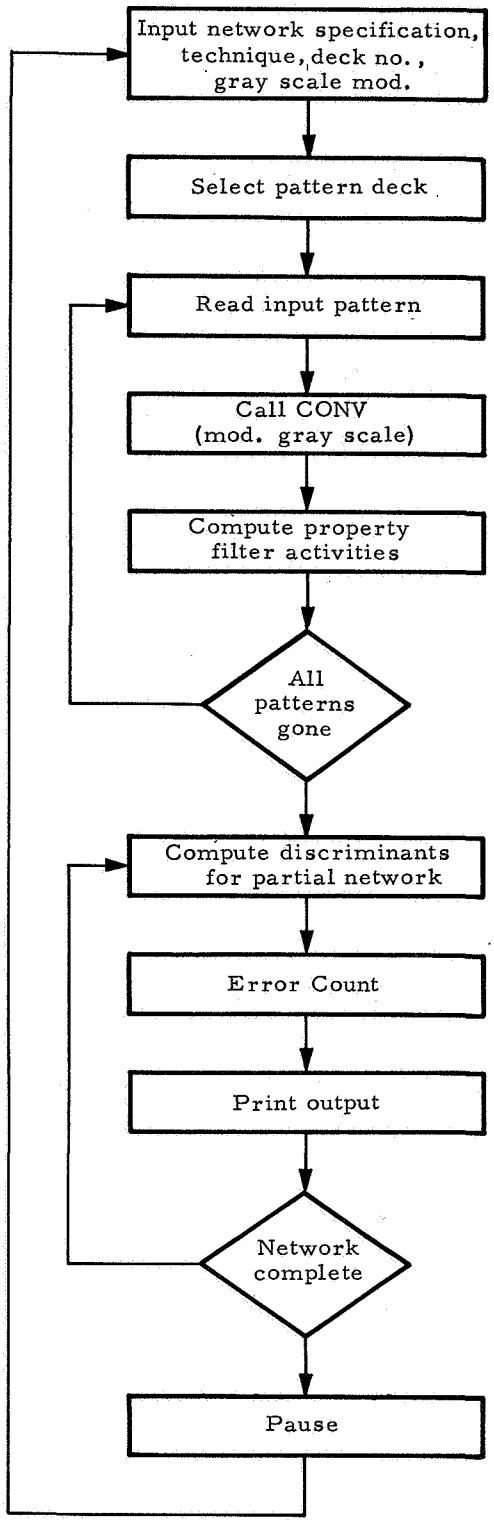


Figure 29. Flow Chart for Generalization Program (VERPR)

program is similar to the DAID output, and the partial networks, as recorded on punch cards, is compatible with the VERPR program.

3. TAPGN – This program accepts both binary pattern decks, in the form of punched cards, transforms the gray scale, and records the resulting patterns on magnetic tape. The program applies seven linear transformations to the gray scale, using a separate transformation for each column of the input raster. The seven transformations are arbitrary, and could be identical, of course. Due to system problems, operation with tape input of patterns was unreliable, and this program saw only limited service.

4. DECOUT – This program accepted a binary pattern deck from cards, and produced a pictoral version of the patterns, as shown in Figures 18 through 27.

5. RPER – This program accepts the punched card output from DAID and applies random perturbations to the input and/or output weights of the logic units. These perturbations are multiplicative factors, normally distributed with unit means and controllable variances. The variance for the input weights and the variance for the output weights are individually controllable. The output is the perturbed set of partial networks on punched cards compatible with VERPR.

6. LSQF – This program accepts up to 500 pairs of system loss-number of errors data, and fits by least squares all polynomials up to eight degree to errors as a function of loss.

7. Three programs are available for transferring patterns from punched cards to punched paper tape and magnetic tape, and from punched paper tape to magnetic tape.

5.4 Results

5.4.1 Random Variations

An attempt was made initially to compare the techniques based on individual runs. There is, however, an element of randomness in the design of a machine, as the subspaces for the discriminant analyses are produced with the aid of a random number generator. The same starting number was provided for each run, so that each design technique was supplied the same set of subspaces on which to operate. It developed that the set of subspaces was a very good set for some techniques, and a very bad set for others.

To determine the extent of these random variations, three additional networks were designed with the "Perpendicular Bisector" technique on the easy classification, using different initial numbers for the random number generator. It became clear that the random variations were of a magnitude comparable to the other effects sought. Additional network designs were obtained for the remaining seven techniques. Figures 30 and 31 show the extent of these variations for the system loss.

Data from the three or four networks for each technique were averaged, to provide an estimate of what an "average" network might do. Since most of the conclusions of this section are based on these 26 machine designs, and the remaining runs are primarily to provide confirmation, very few of the other runs have been replicated.

The analyses have been based heavily on the system loss as a means of comparison. The system loss generally provides a smoother presentation than does the number of errors on the design deck. The two are strongly related. As a rule for the 240 sample patterns, when the system loss is less than ten, there are no errors, and when the system loss is greater than ten there is at least one error on the design deck. This rule has held in all but about a dozen cases in the several hundred runs examined. The number of errors can be predicted reasonably well in terms of the system loss. Least square polynomials

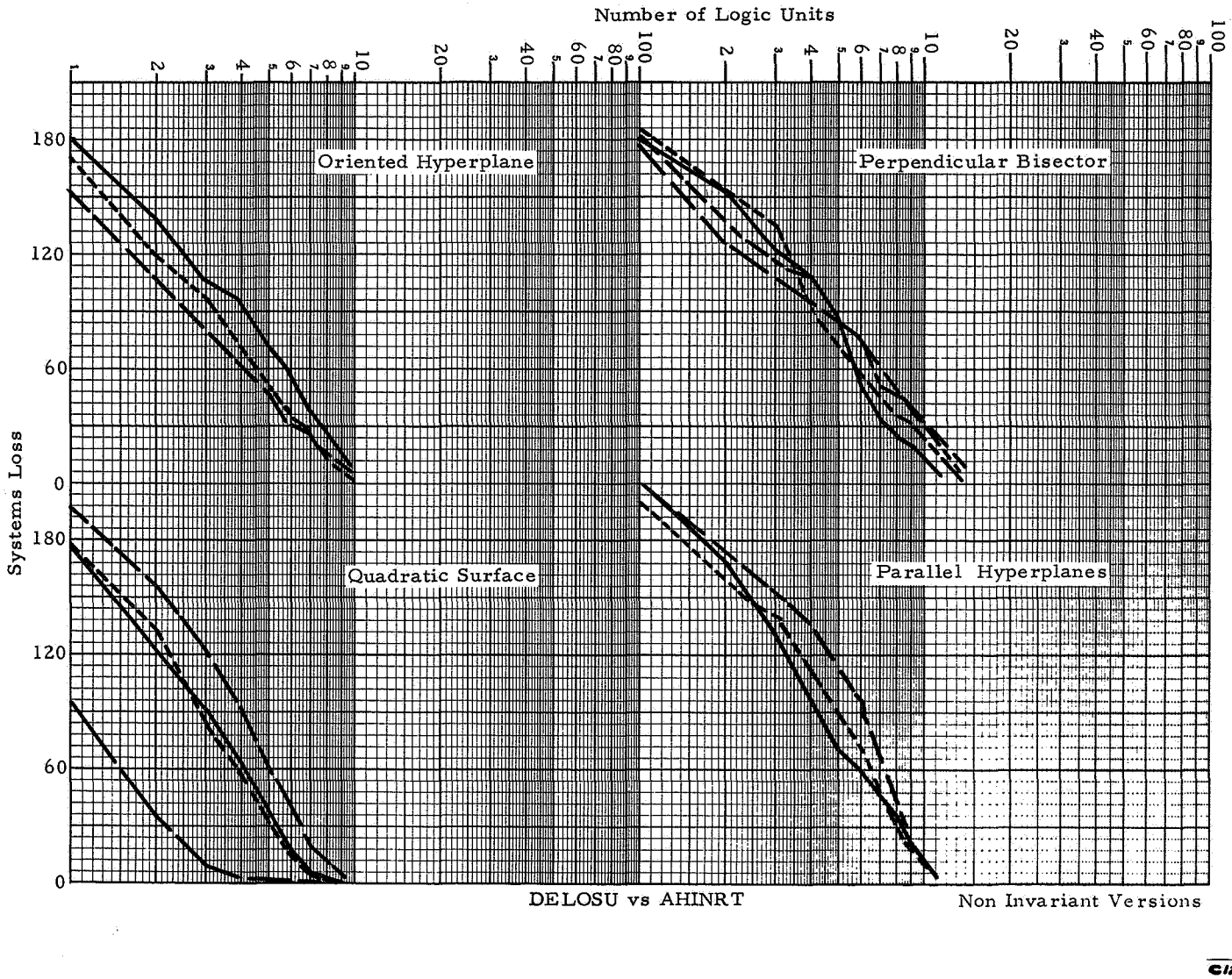


Figure 30. Effects of Randomness - System Loss (Curves are for individual networks)

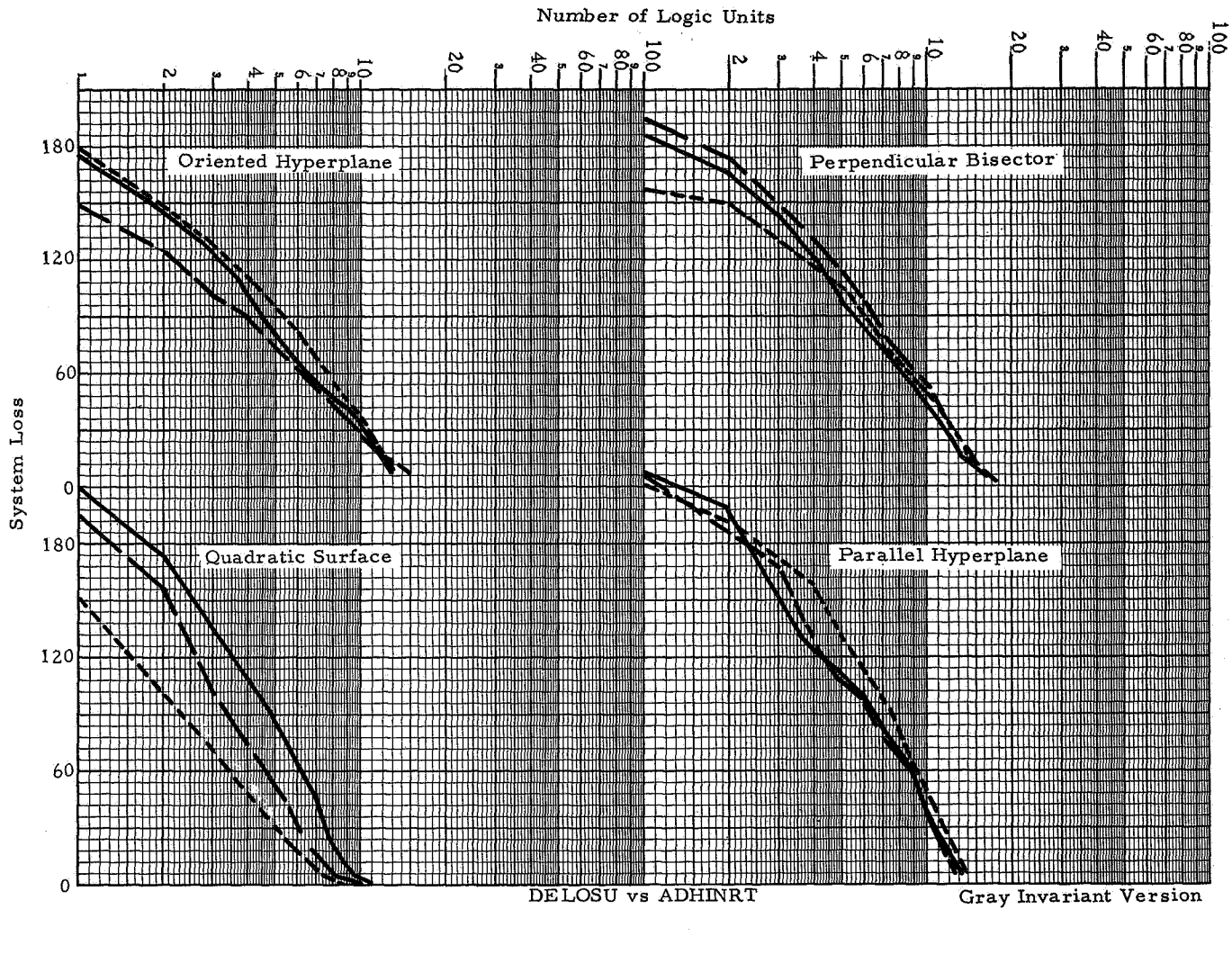


Figure 31. Effects of Randomness - System Loss (Curves are for individual networks)

up to seventh degree were fitted to the data provided by seven of the "average" machines. Over the range of interest, the sixth and seventh degree polynomials are virtually indistinguishable. This curve is shown in Figure 38. The standard deviation around this curve is 1.5 errors for the "average" machines. Figure 32 also presents the best (fourth degree) polynomial for generalization errors against design system loss. The standard deviation in this case is five errors.

5.4.2 Parameter Variations

A standard set of parameters was used in most of the network designs. These parameters were varied individually to determine their effect on the design procedure. In particular, the number of iterative cycles, the number of units generated for each one selected, and the number of input connections per logic unit were varied. In addition, there is a parameter in the gray invariant version of the "Parallel Hyperplane" technique that is not specified analytically. This is the half-angle of the right circular cone forming the logic unit decision boundary.

Iterative Cycles

An iterative cycle consists of one adjustment to the output weight of each logic unit. The number of iterative cycles following the addition of each logic unit was held constant throughout the design of a network. Since this parameter primarily affects output weights, only the "Perpendicular Bisector" technique was used in this investigation. This technique requires considerably less computer time for a machine design than the other techniques.

Figure 33 shows the results for two, three, and five iterative cycles after each unit is added. The curves are for individual runs. Table VII shows similar results for single machines designed on Deck 2 for two, three, four, five, six, and seven iterative cycles. All single runs were supplied with an identical start for the random number generator.

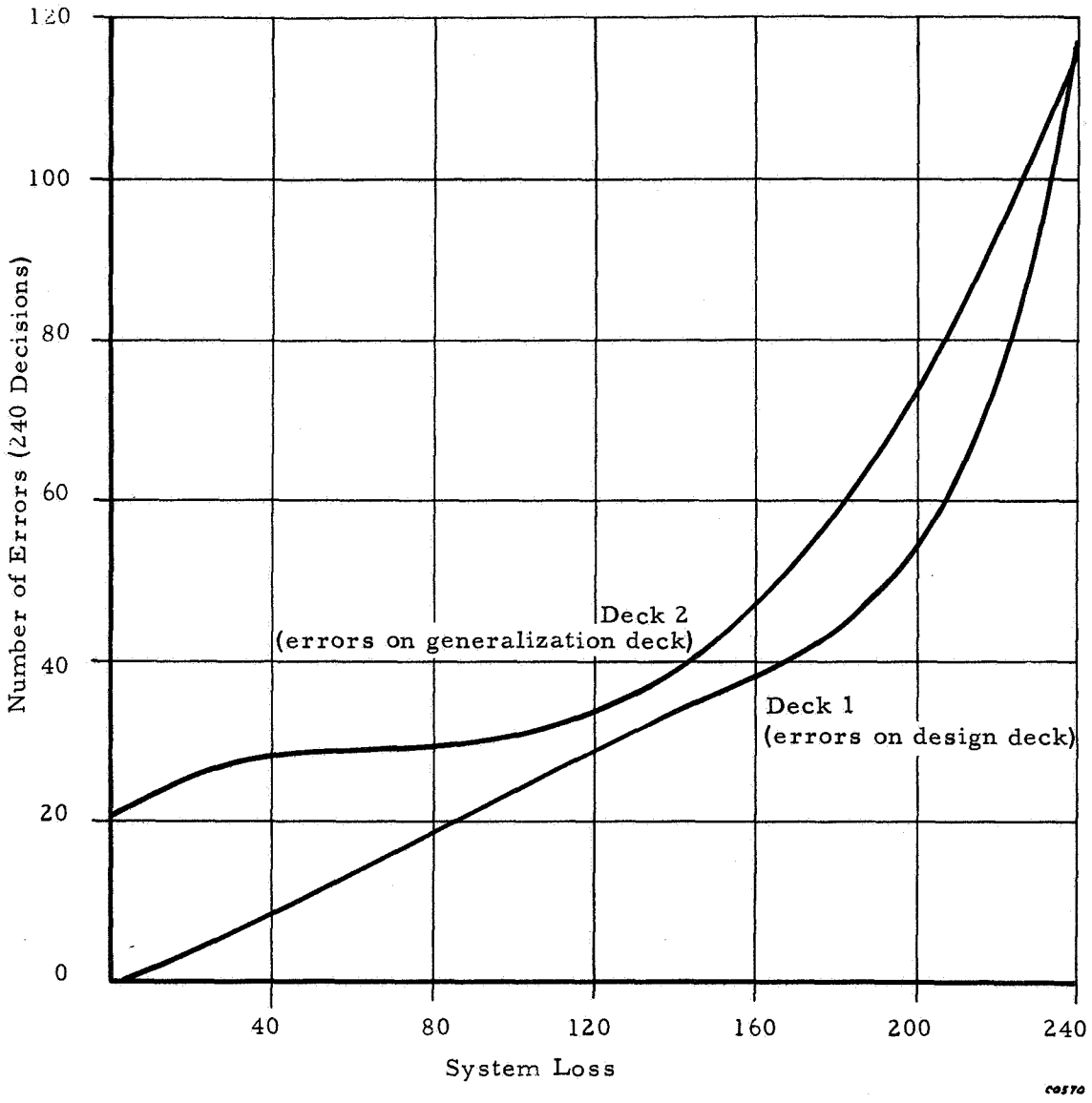


Figure 32. Errors as a Function of System Loss

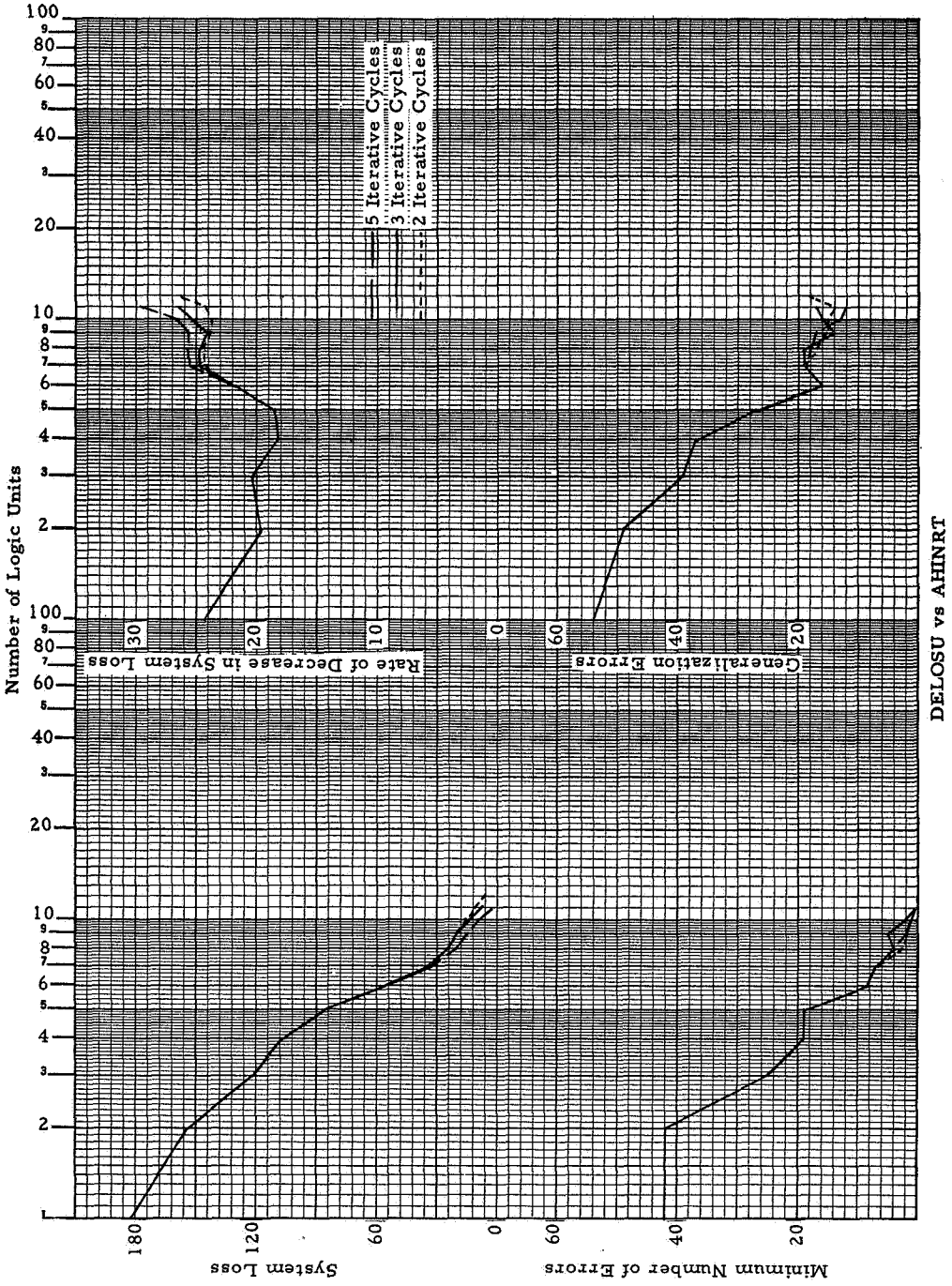


Figure 33. Effect of Iterative Cycles — Perpendicular Bisector Technique

TABLE VII

EFFECTS OF ITERATIONS ON SYSTEM LOSS PERPENDICULAR BISECTOR TECHNIQUE; DESIGNED ON DECK 2; DELOSU vs AHINRT

Number of Logic Units	Number of Iterations					
	2	3	4	5	6	7
1	187.18	187.18	187.18	187.18	187.18	187.18
2	153.61	153.61	153.61	153.61	153.61	153.61
3	114.81	114.80	114.80	114.80	114.80	114.80
4	93.04	92.88	92.84	92.84	92.83	92.83
5	74.01	74.00	72.43	72.43	72.43	72.43
6	59.07	58.91	56.19	56.17	56.16	56.16
7	51.55	51.33	43.31	43.31	43.31	43.31
8	34.05	32.71	32.64	32.55	32.50	32.47
9	22.90	21.08	24.26	25.09	24.89	24.75
10	16.67	15.23	19.84	18.48	18.08	17.76
11	12.51	10.95	12.95	7.20	6.47	5.88
12	9.11	4.60	9.58			
13	6.37		5.81			

The conclusion appears to be that there is little benefit in having more than five iterative cycles. Even then, the benefit is not derived until well along in the machine design. Each time a weight is changed, it is adjusted so that the partial derivative is zero with respect to that weight. When there are only a few units in the machine, there are not many other changes to disturb the "optimality" of the weight. With more units in the network, there are more changes between successive corrections so that one might expect that more cycles would be beneficial. In comparing the two-cycle and five-cycle runs designed on Deck 1 (Figure 33), the first five units selected were virtually identical. The next two units used the same subspaces, but had somewhat different hyperplanes in these subspaces (due to the somewhat different weighting of the patterns).

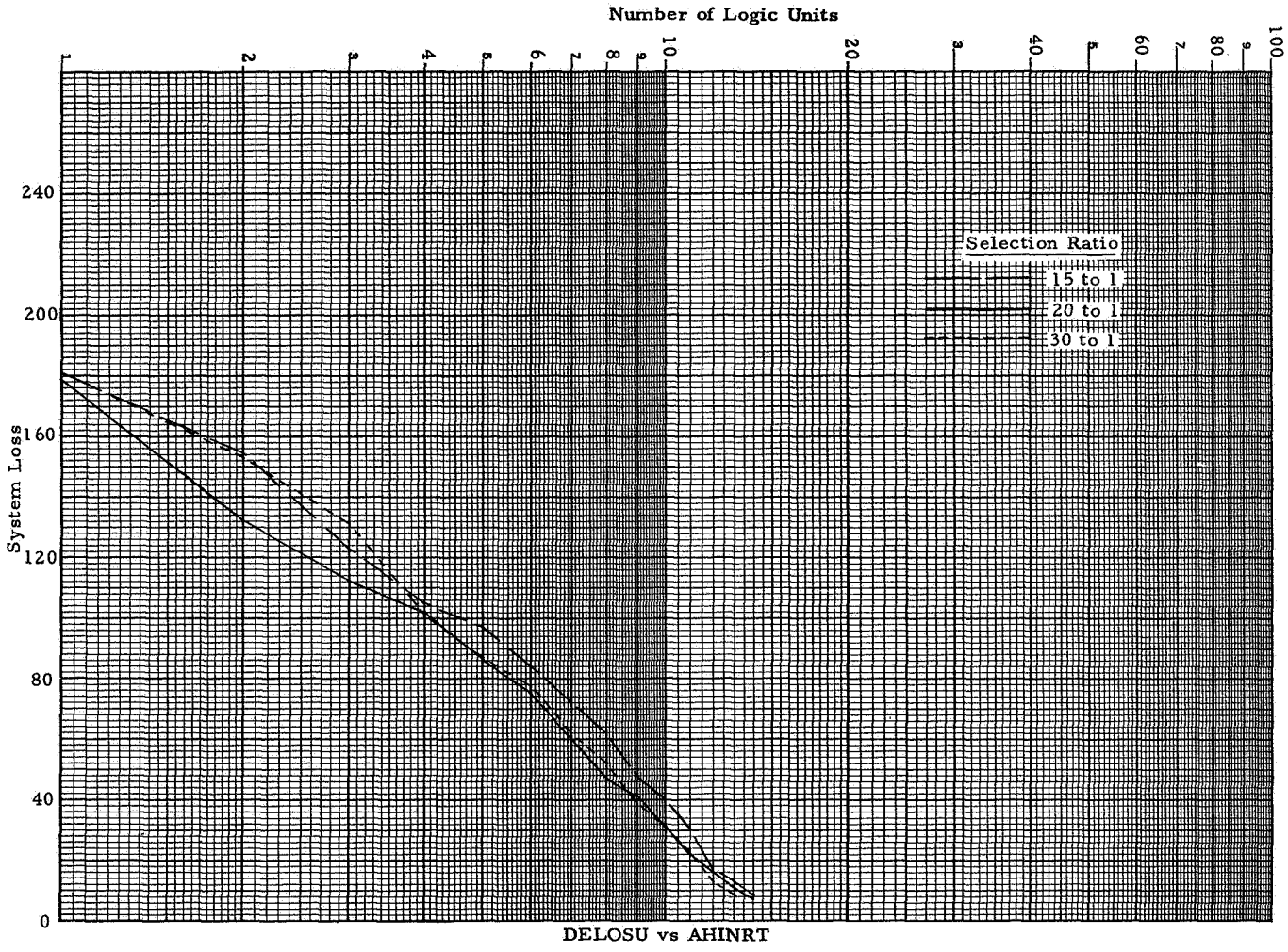
These results suggest that a better procedure would be one in which the number of iterative cycles increases as the number of units in the network increases. The iterative process used on the TIROS frames was somewhat different, and is described in Section 4.5.

It is of interest to note that the generalization results shown in Figure 33 are the best obtained. The five-cycle machine gives fewer errors than the correlation scheme described in Section 5.2.4. The "Perpendicular Bisector" technique provided the best generalization results of all the techniques tested.

Selection Ratio

The number of logic units generated for each one selected was in general taken to be 20. Figure 34 shows results for ratios of 15, 20, and 30. The curves for 20 represent a four-network average. These data show that for this problem, there is little difference between these selection ratios.

The high selection ratios are most likely responsible for the degree of randomness exhibited, as it is the tail of the distribution which is being sampled. If the unit of median or average effectiveness was selected, one would expect that these selection ratios would provide



C0588

Figure 34. Effects of Selection Ratio — System Loss

a great deal of uniformity from one run to the next. The notion of rejecting the most effective units in favor of one of average effectiveness makes more sense than would appear at first. In a problem of the complexity of satellite photo interpretation, it seems unlikely that any one unit could solve a large part of the problem. By insisting on highest efficiency, it is possible that units will be chosen which measure pattern features which are extraneous to the problem, but which exhibit spurious correlations with the classification of the sample patterns due to the limited extent of the sample.

In the design runs on the TIROS photographs, 150 logic units were generated at a time. Ten of these were retained in the machine.

Input Connections

The number of input connections per logic unit was varied for the "Perpendicular Bisector Technique." Figure 35 shows the system loss for four, six, and eight input connection noninvariant units, and six and eight input connection gray invariant units. The data for the six-input connection units are "average" networks; the remaining data are for individual networks.

These data fit the theory put forth in Section 5.4.3, as Figure 41 will show. This theory suggests that network effectiveness is a function of the number of freely adjustable input weights and the number of output weights. For the noninvariant "Oriented Hyperplane" and "Perpendicular Bisector" techniques, the number of freely adjustable input weights is equal to the number of input connections; for the non-invariant "Parallel Hyperplane" it is one more than the number of input connections. The gray invariant versions each have less freely adjustable weights than their noninvariant counterparts. Empirically, an output connection weight should be counted three times as heavily as an input weight.

If the theory suggested is true, then the indication is that each logic unit should have as many input connections as possible.

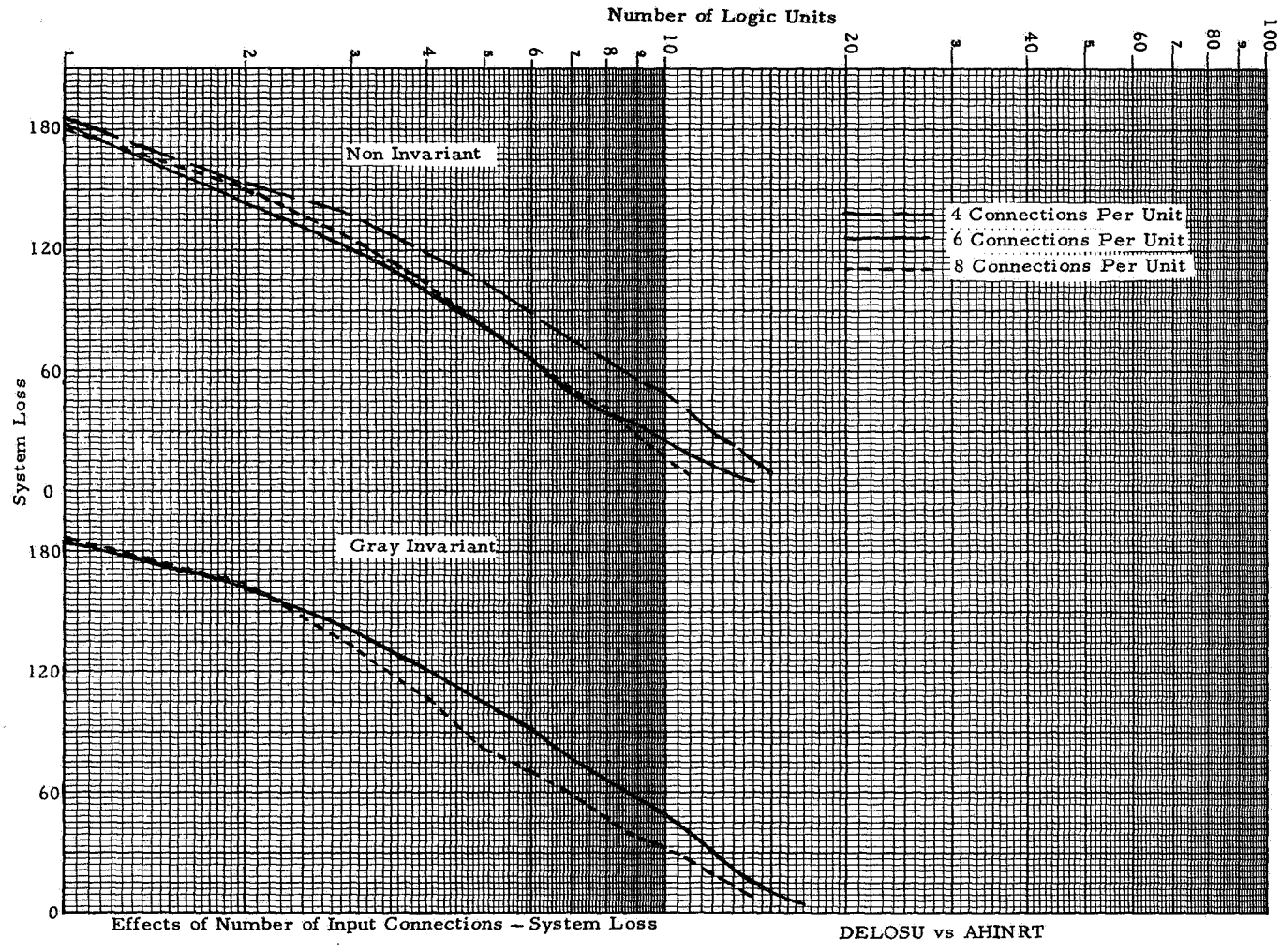


Figure 35. Effects of Number of Input Connections – System Loss

Considerations to be balanced against the economy of logic unit circuitry thus gained are:

1. Computer design time follows a cubic relation with the number of input connections for the "Parallel Hyperplane" techniques and the gray invariant version of the "Oriented Hyperplane," and a quadratic relation in the noninvariant version of the "Oriented Hyperplane."
2. Hardware costs rise as logic unit circuitry must be more accurate to accommodate more input connections.
3. A saturation effect undoubtedly occurs – there seems to be some evidence of this in the eight-and ten-input connection noninvariant data in Figure 35.

Cone Angle

The gray invariant version of the "Parallel Hyperplane" technique provides a right circular cone* as the decision boundary of the logic unit. One analytic result might suggest a desirable value for the half-angle of this cone – the derivation of the angle which would give a cone that divides spheres into two equal volumes. For five dimensional spaces this angle is 70 degrees.

This angle was investigated experimentally. The results are shown in Table VIII. The data for 67, 72, and 77 degrees are averages of two runs; the data for 70 degrees are averages of three runs. The experimental data seems to confirm the hypothesis that the cone angle which divides the space equally is a good one. This angle was adopted for the subsequent studies on alphabetic characters.

With this cone angle, one might expect that if the negative patterns were unstructured, then the resulting logic units would

*This is not strictly correct. The boundary is the set of points whose projection on a hyperplane is a right circular cone in that subspace. This hyperplane is one perpendicular to a vector with all components equal.

TABLE VIII

EFFECTS OF CONE ANGLE ON SYSTEM LOSS GRAY INVARIANT PARALLEL HYPERPLANES TECHNIQUE; INCOMPLETE NETWORKS; DELOSU vs AHINRT

Number of Logic Units	Cone Angle in Degrees							
	77*	72*	70**	67*	62	57	52	47
1	212.46	211.28	215.27	212.83	220.62	224.80	226.87	226.87
2	182.35	180.01	186.40	177.42	188.19	190.95	197.07	200.13
3	168.80	165.32	168.29	165.65	175.61	171.48	171.79	179.87
4	149.67	151.16	151.07	145.58	160.79	150.69	159.32	166.66
5	132.72	126.16	130.16	131.62	150.56	133.72	142.65	144.52
6	123.20	111.38	114.23	118.69	144.69	118.17	133.42	128.71
7	105.62	94.92	90.36	90.48	126.25	104.29	109.61	112.56
8	85.76	82.44	81.35	69.36	119.89	93.05	102.12	95.08
9	72.67	57.78	68.36	52.26	105.7	83.00	84.80	86.19
10	63.22	43.62	56.37	40.26	84.90	71.93	75.76	71.83
11	55.44	32.79	46.14	31.60	75.67	53.52	59.95	57.91

* 2 network averages

** 3 network averages

be activated by about half of the negative patterns, and more than half of the positive patterns. It seems reasonable that a more symmetric arrangement, which would result from a smaller cone angle, might be more desirable. The results in Table VIII are not precise enough to determine the optimum. In Section 6.0, the equal volume cone angle, and a smaller angle were tested on the TIROS frames.

5.4.3 Gray Invariance

The effects of adding gray scale invariance to the four basic techniques are discussed in this section. Except for Figure 47 all curves are for the "average" machines.

Figures 36 through 39 present the data on the eight "average" machines. The data are plotted in such a way as to facilitate the comparison of the basic techniques and their gray invariant versions are shown by broken lines.

In Figure 36, the four comparisons are drawn with respect to the rate at which the networks "learn" to classify the sample patterns. The number of errors in classifying the sample patterns is plotted as a function of the number of logic units in the network. Figure 37 presents a similar comparison, showing the system loss as a function of the number of logic units. As is to be expected, due to the constraints placed upon them, the gray invariant techniques solve the problem of classifying the sample patterns somewhat slower than do their noninvariant counterparts. When plotted on semi-logarithmic paper, the difference appears to be a shift to the right for the invariant versions. An explanation of this phenomenon will be attempted later in this section.

A comparison of Figures 36 and 37 shows the system loss provides a less erratic presentation of the data. Figure 38 presents the average rate of decrease in the system loss (R). For a fixed number of logic units (N), this measure is equivalent to the system loss (SL) since it is computed from the formula

$$R = 100 - 100 \left(\frac{SL}{240}\right)^{1/N}$$

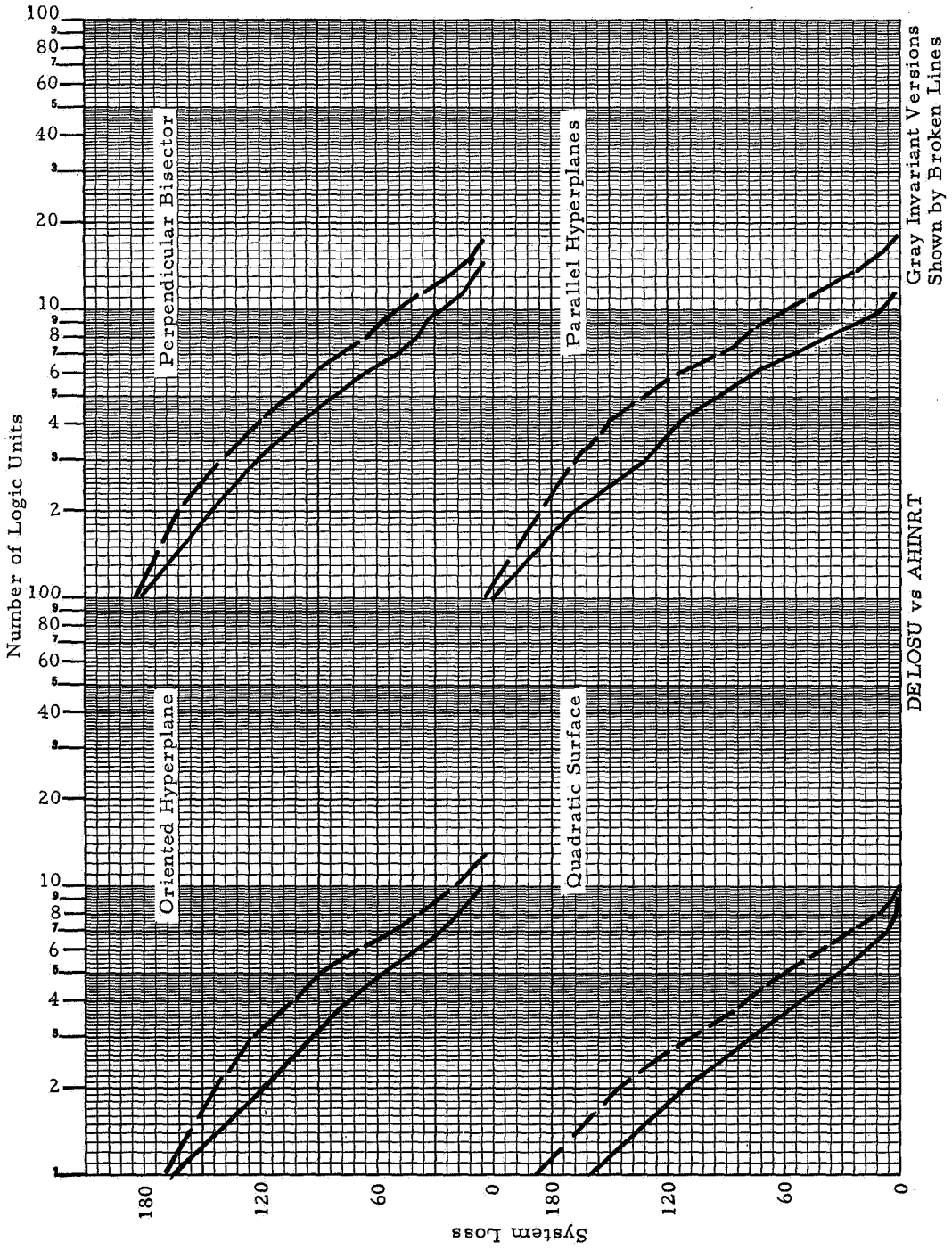


Figure 36. Gray Invariance Comparisons — System Loss

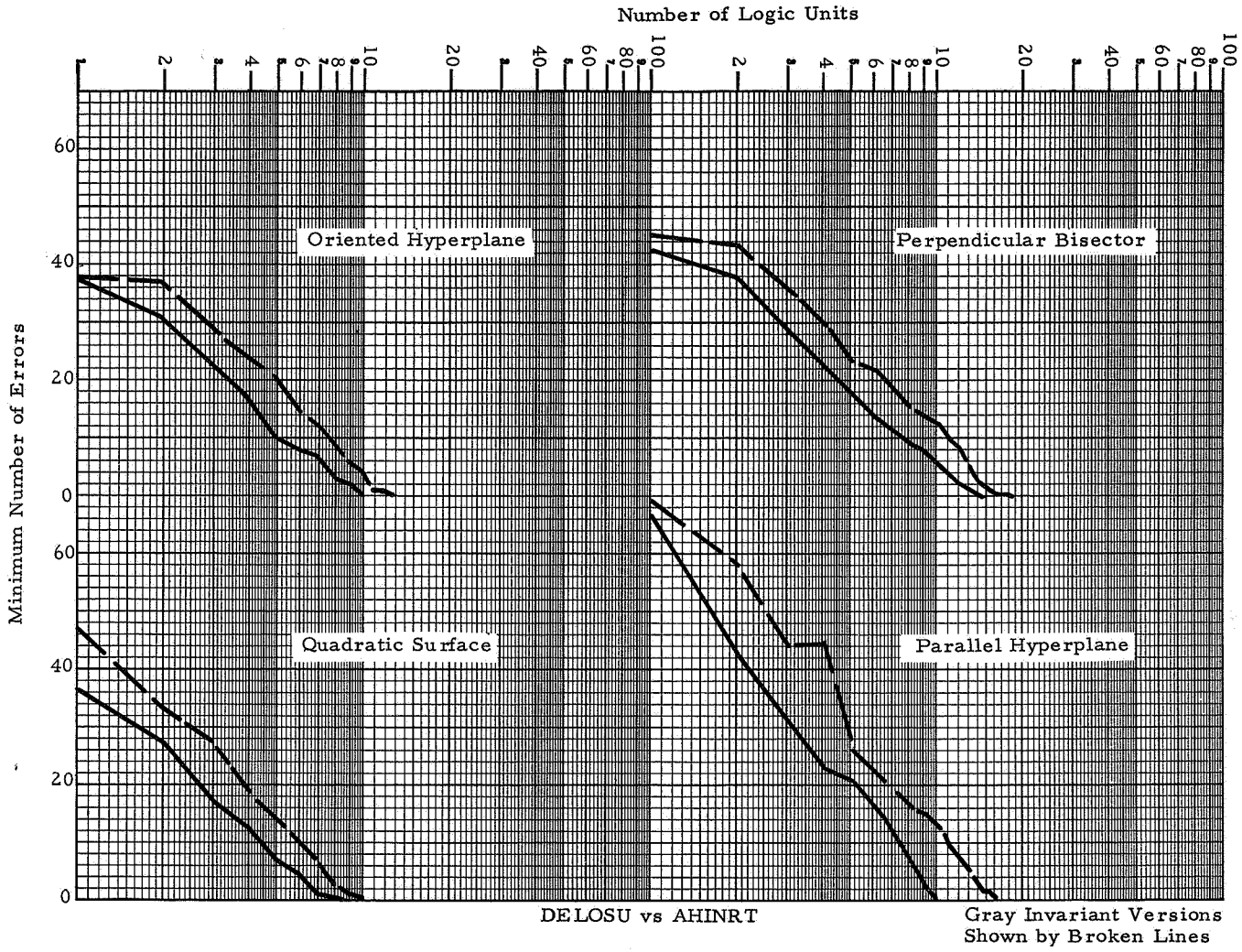


Figure 37. Gray Invariance Comparisons — Learning Rate

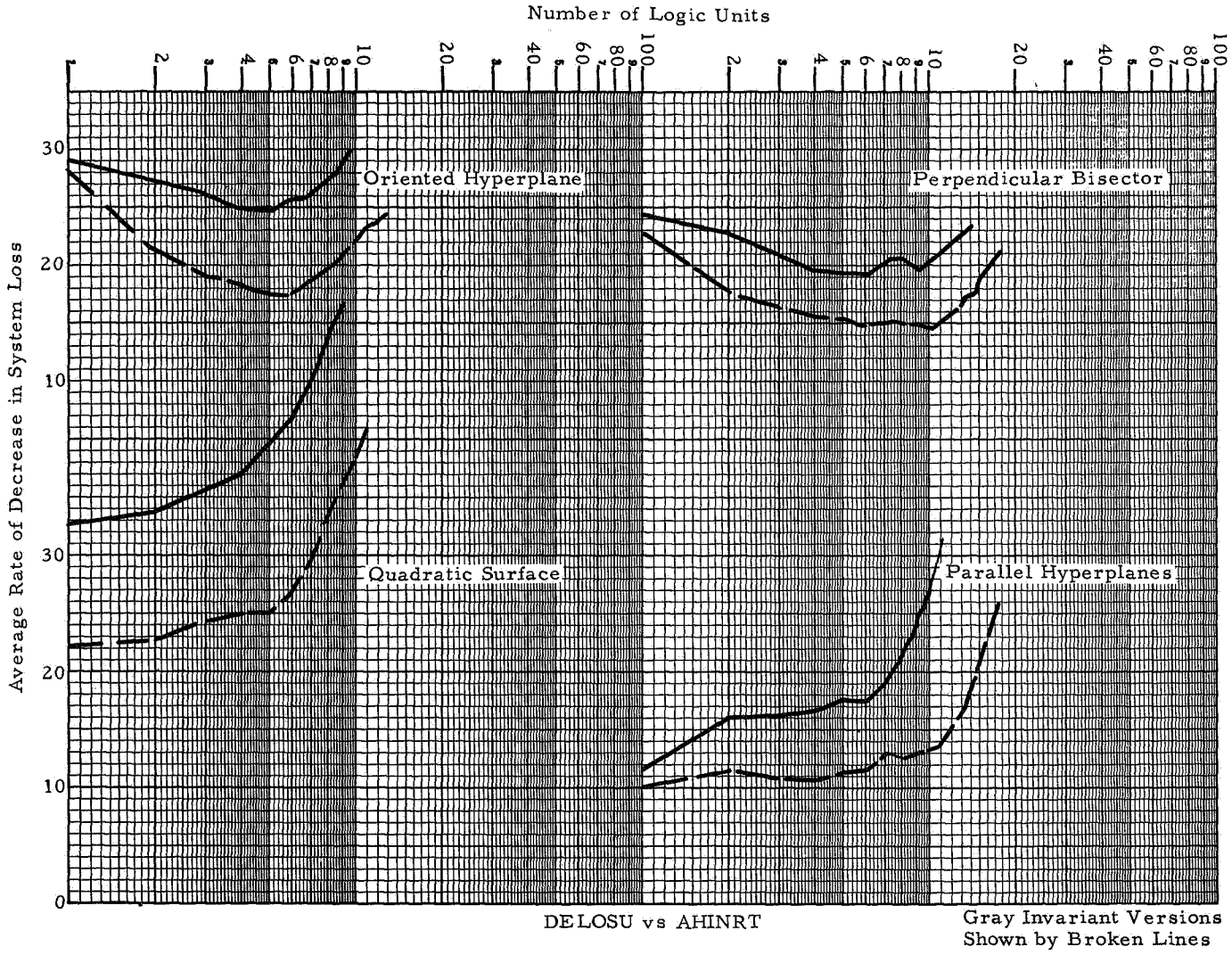


Figure 38. Gray Invariance Comparison — Rate of Decrease in System Loss

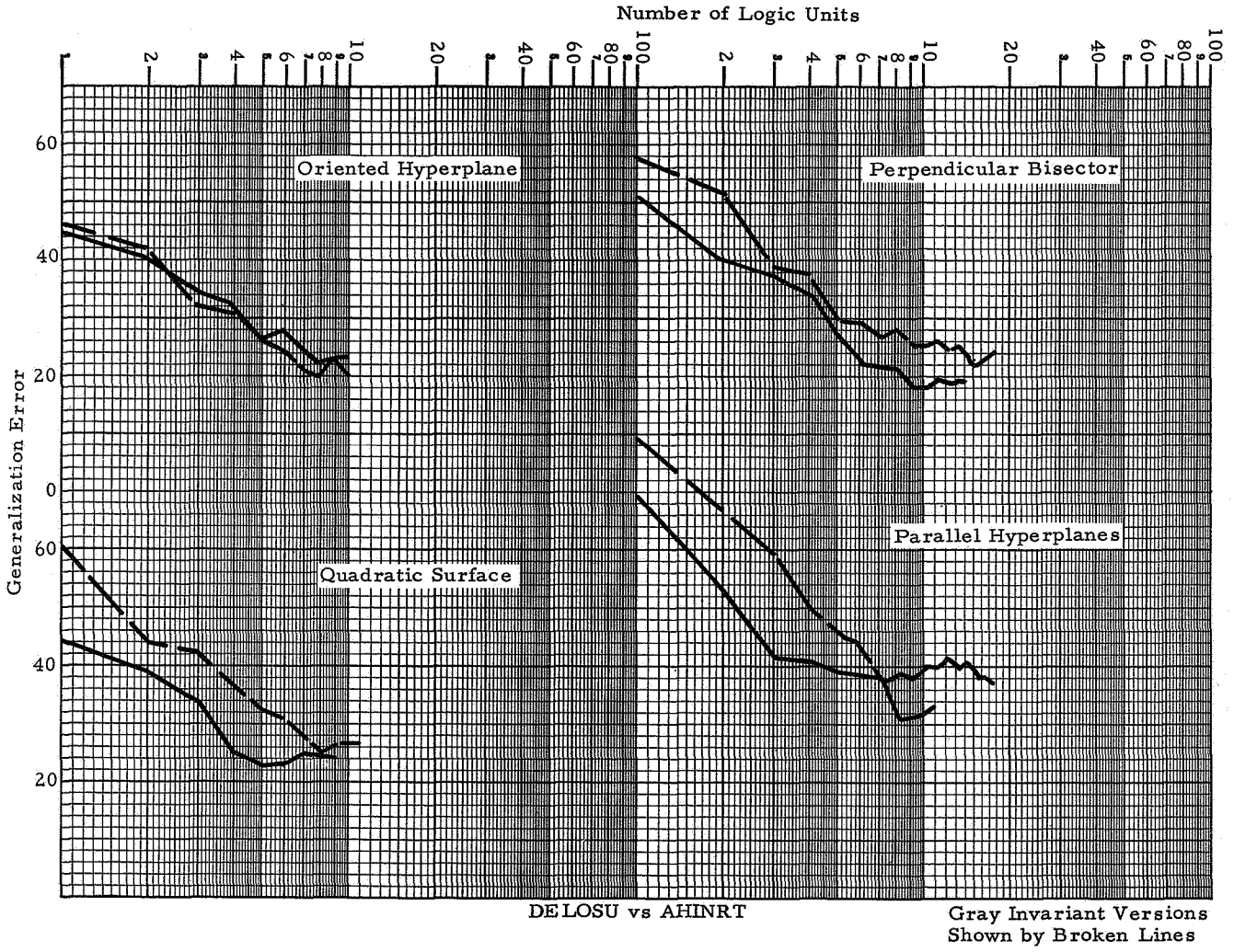


Figure 39. Gray Invariance Comparisons — Generalization Errors

It has the significance that if the system loss had decreased by exactly R percent each time a logic unit was added, the system loss would be SL when N units were in the network. These curves have been invaluable in the course of the study for a deviation from the characteristic shapes shown in the figure was indicative of either a program error, or the fact that the performance of the network will be unusually good or bad compared to the average for that technique.

Comparing Figure 38 and 39, it can be observed that with the addition of one unit after the rate of decrease curves take the sharp turn upward, the rate at which generalization errors are made ceases to improve. This coincidence is worthy of further investigation, for if the sharp rise in the R curve is indeed indicative of the time when the pattern weighting approach has so distorted the problem that the technique ceases to find significant pattern features, and instead finds spurious correlations, then such a rise could prove to be an invaluable indicator.

Figure 39 presents the generalization errors when the machines are tested against Deck 2. These numbers are also plotted against the number of logic units (the logic unit output weights, assigned as each logic unit is added, are saved and the partially designed machines are tested to determine the rate at which the techniques "learn" to generalize). It is seen that these recognition networks achieve a ten percent error rate on the patterns of Deck 2, noticeably more than the best correlation scheme mentioned in Section 5.2.4. The gray invariant versions acquire this accuracy more slowly than do the noninvariant versions, though there seems to be little difference between the terminal performances.

In the "Oriented Hyperplane" and "Perpendicular Bisector" techniques, gray invariance is achieved by placing two constraints on the parameters defining the logic unit hyperplanes – the input weights must have zero sum, and the threshold must be zero. Thus with six input connections there are six degrees of freedom in defining a logic unit hyperplane with the noninvariant version; the gray invariant version

has only four degrees of freedom. In the "Parallel Hyperplane" technique the gray invariant version has five degrees of freedom as opposed to seven for the noninvariant version for specifying the decision boundary. For the "Quadratic Surface" technique the number of degrees of freedom are 14 and 21 for the invariant and noninvariant versions, respectively.

Figure 40 presents the system loss plotted against the degrees of freedom in the network. In this figure, one extra degree is allowed per logic unit for the specification of the output weight of the unit. The fit for the three linear input techniques is quite good. Additional evidence is offered in Figure 41, in which the data in Figure 35 are replotted against degrees of freedom.

The assumption that the input weights for a unit contribute as much as the output weight is probably false. The data in Figure 40 would show even better agreement if the output weight was given a value equal to three to six times that assigned to an input weight for the noninvariant version.

A better test, one which does not depend on the relative value assigned to the output weight, is to compare networks with different numbers of input connections per logic unit. Figure 41 does this for four, six, and eight-input connection "Perpendicular Bisector" networks, and six and eight-input connection invariant version networks. The six-input connection networks are "average" machines, the others are individual runs. The fit in the left-hand panels would be improved by counting output connections more heavily than input connections; the remarkable fits in the right-hand panels would be unaffected.

A comparison between the invariant and noninvariant versions would not be complete without a comparison on patterns in which the brightness and contrast have been altered, since invariance to this change is the purpose of the invariant versions. Four such test decks were used.

One pair of these decks consists of the standard patterns (Decks 1 and 2) modified so that the binary values 0 and 1 in the

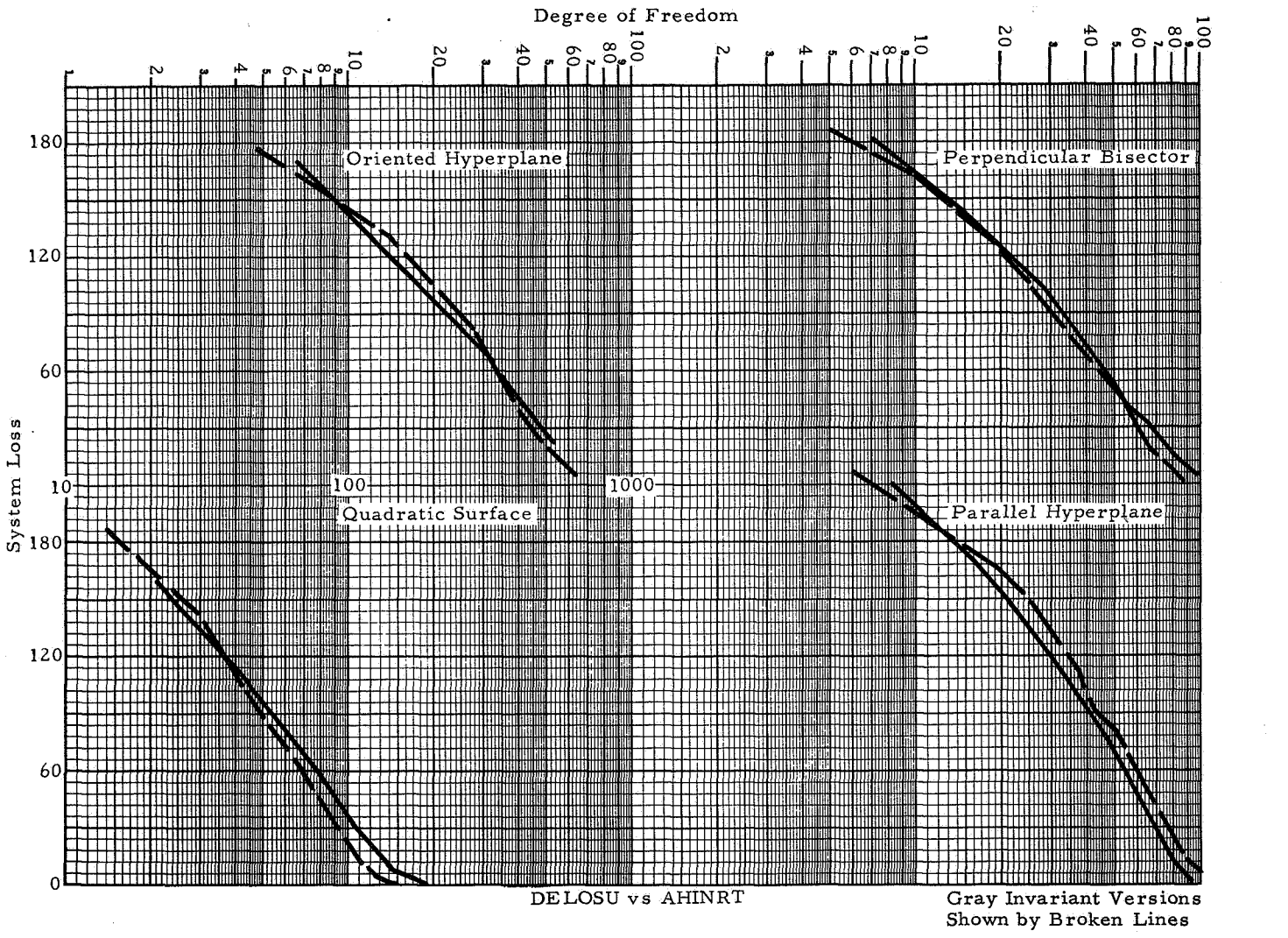


Figure 40. Normalized Comparison of Gray Invariance — System Loss

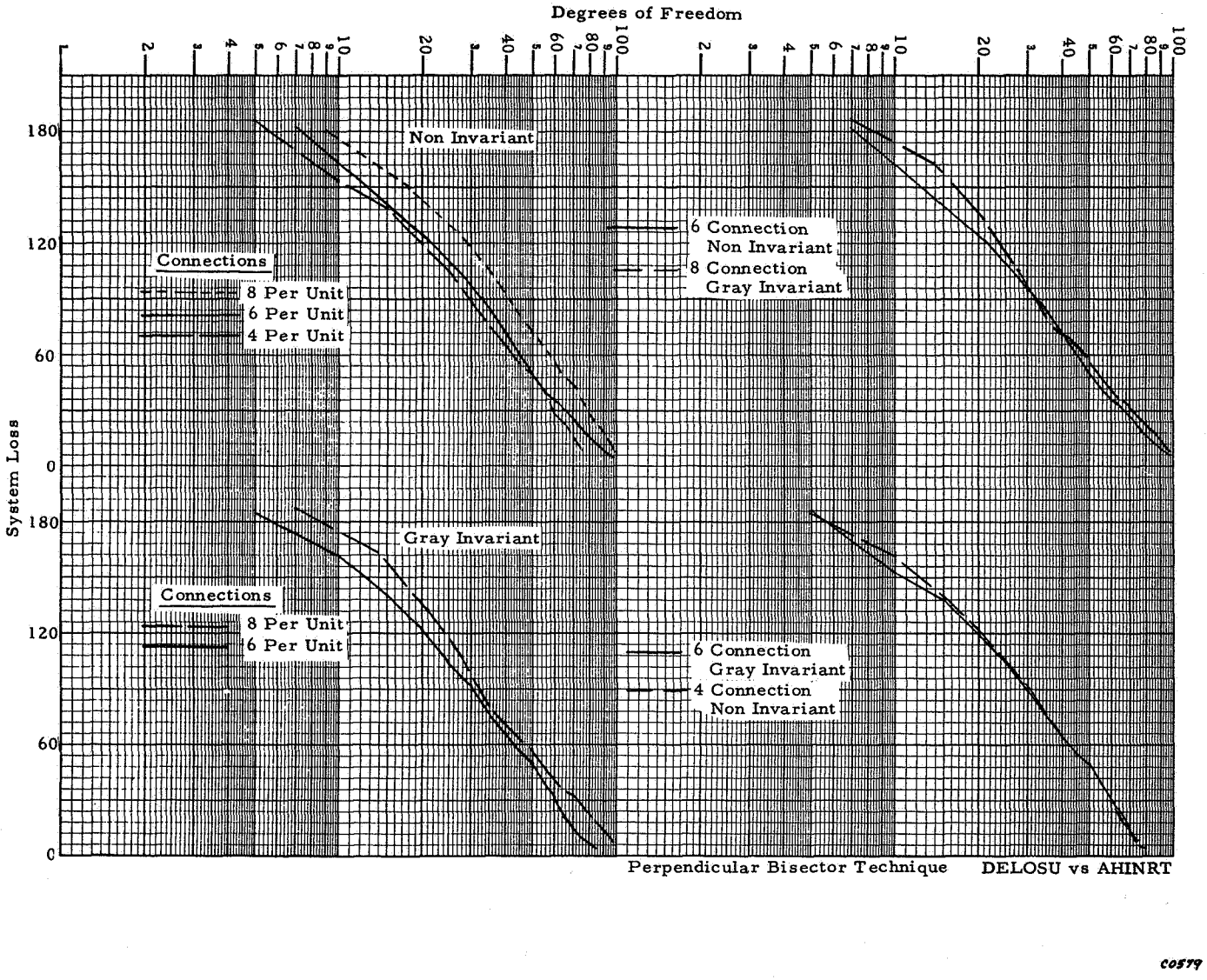


Figure 41. Degrees of Freedom Comparisons — System Loss

original patterns correspond to .3 and .7 respectively in the new patterns. This represents a linear change in the gray scale, so that the performance of the invariant versions is unaffected. Figures 42 and 43 show the performance of the recognition networks on these decks of patterns, which are designated Deck 1, 7 X 3, and Deck 2, 7 X 3. As shown in these figures, the effect of the gray scale change is quite pronounced on the performance of the noninvariant versions.

The other pair of decks consists of the standard patterns modified to give a variable gray scale. In these patterns, the original values of 0 and 1.2 become .6 and 1.8 on the right side of the patterns, and varying linearly to 0 and 1 on the left side of the patterns. This is a nonlinear change in the gray scale, so that none of the network designs are invariant. Figures 44 and 45 show the average network performances of these pattern decks, which are designated Deck 1, VAR, and Deck 2, VAR. Except for the Parallel Hyperplane and its invariant version when tested on Deck 2, VAR, the invariant versions, are somewhat better than the noninvariant versions. This is particularly true for the Perpendicular Bisector technique. Differences are not as pronounced as for the linear change in gray scale case, nonetheless the value of using gray invariant versions is evident.

5.4.4 Distributional Assumptions

The error rates for the "Oriented Hyperplane" and the "Perpendicular Bisector" techniques in Figure 42 are about half those for the "Quadratic Surface" and "Parallel Hyperplane." Figure 39 shows similar results for generalization errors on Deck 2. This probably indicates that for the sample problem, the distributional assumptions of the first two techniques are more valid than for the latter two techniques. That is, for the sample problem, the mean vectors provide a better basis for separating the pattern classes than do the covariance matrices. The fact that the patterns are reasonably centered on the input grid differentiates the sample problem from a photo interpretation problem in this respect.

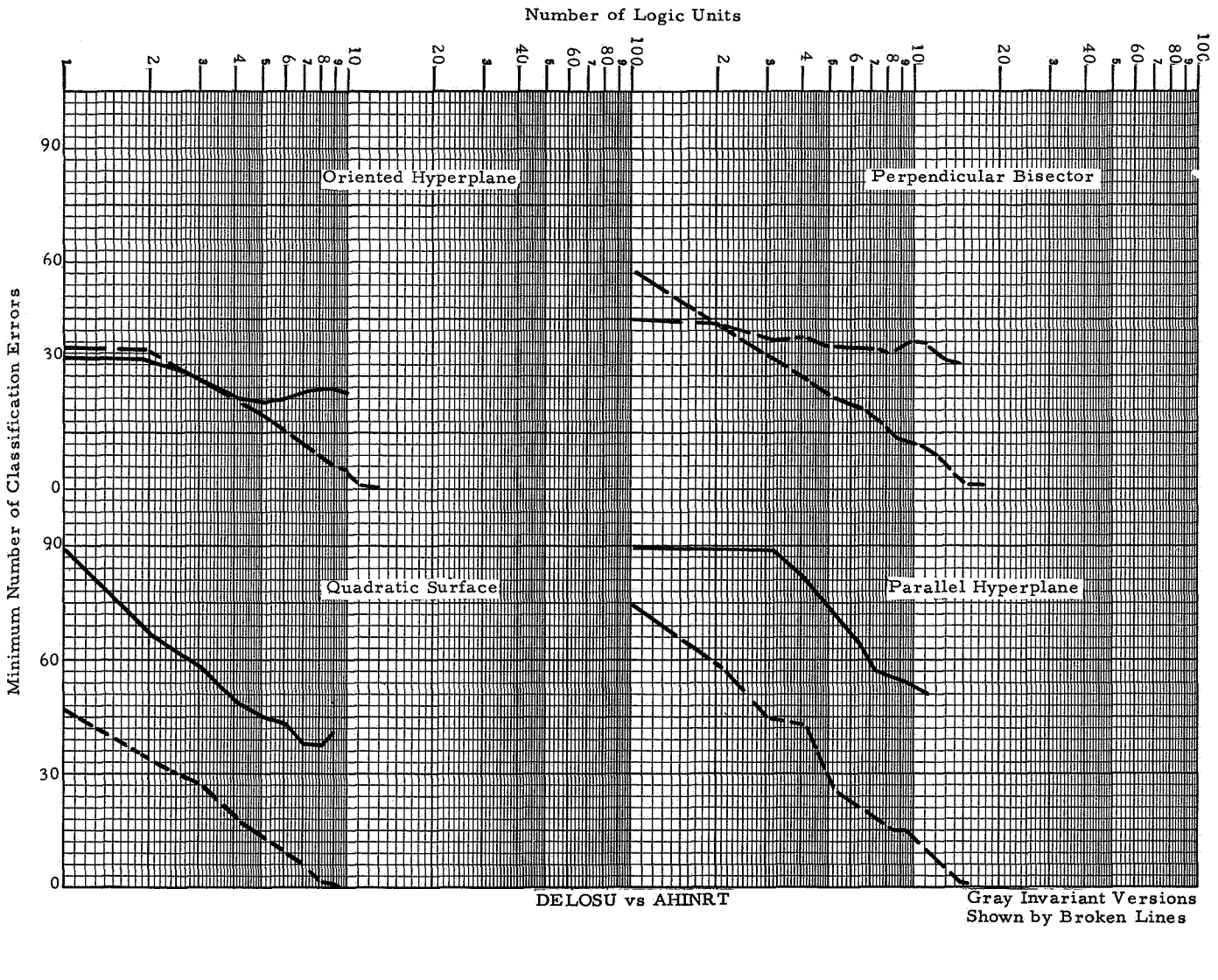


Figure 42. Effects of Gray Scale Patterns — Deck I, 7x3

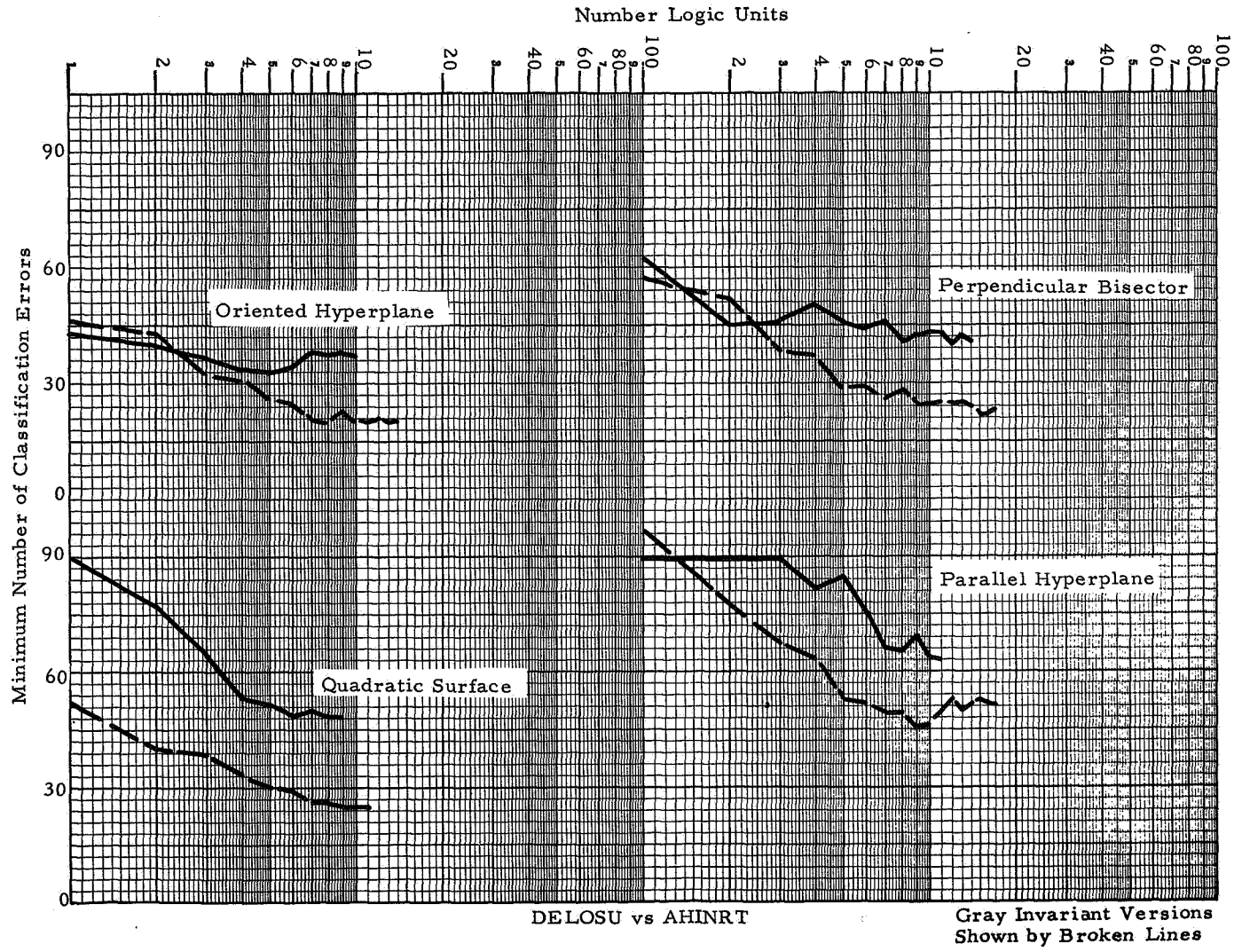


Figure 43. Effects of Gray Scale Patterns – Deck II, 7x3

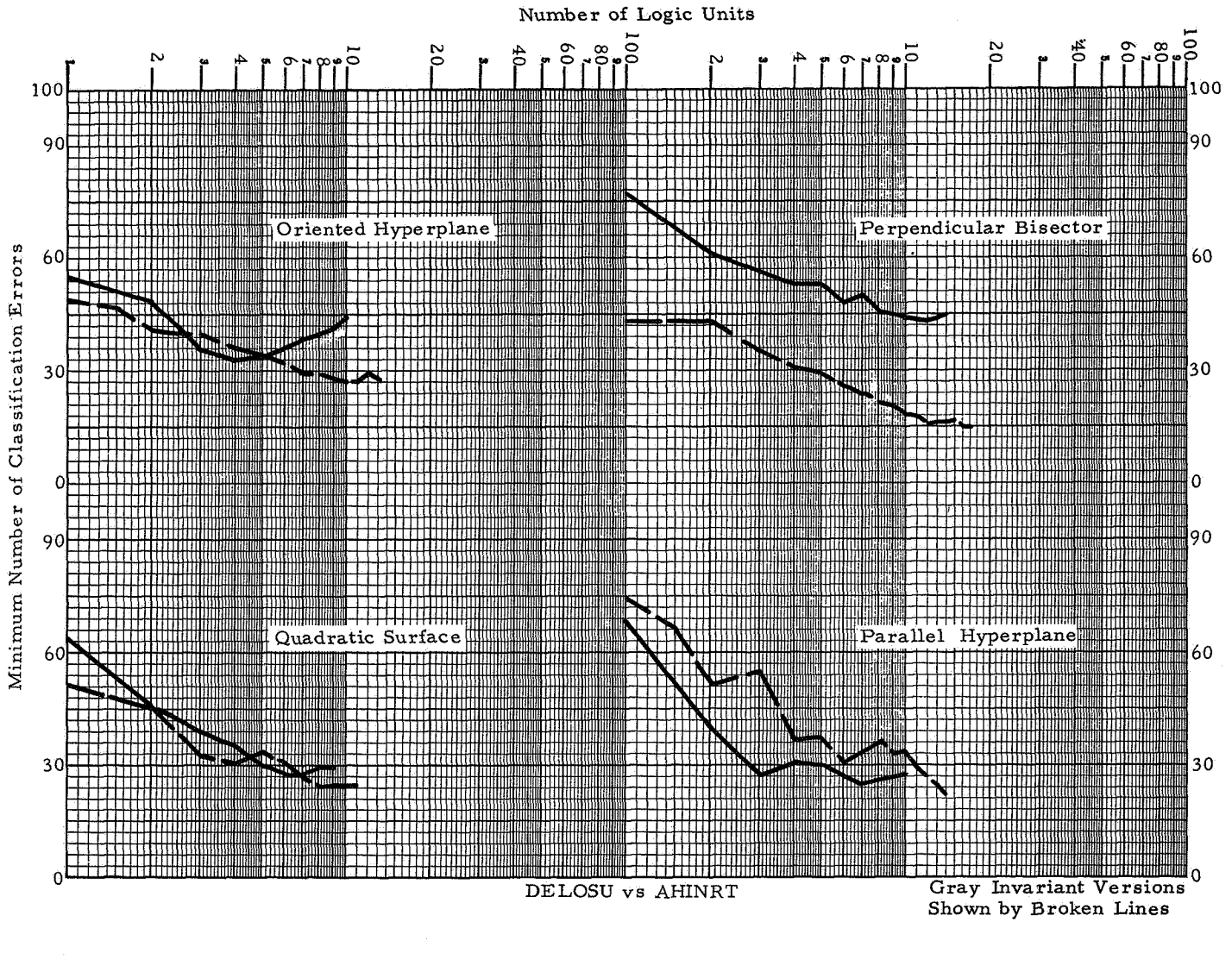


Figure 44. Effects of Gray Scale Patterns — Deck I, VAR

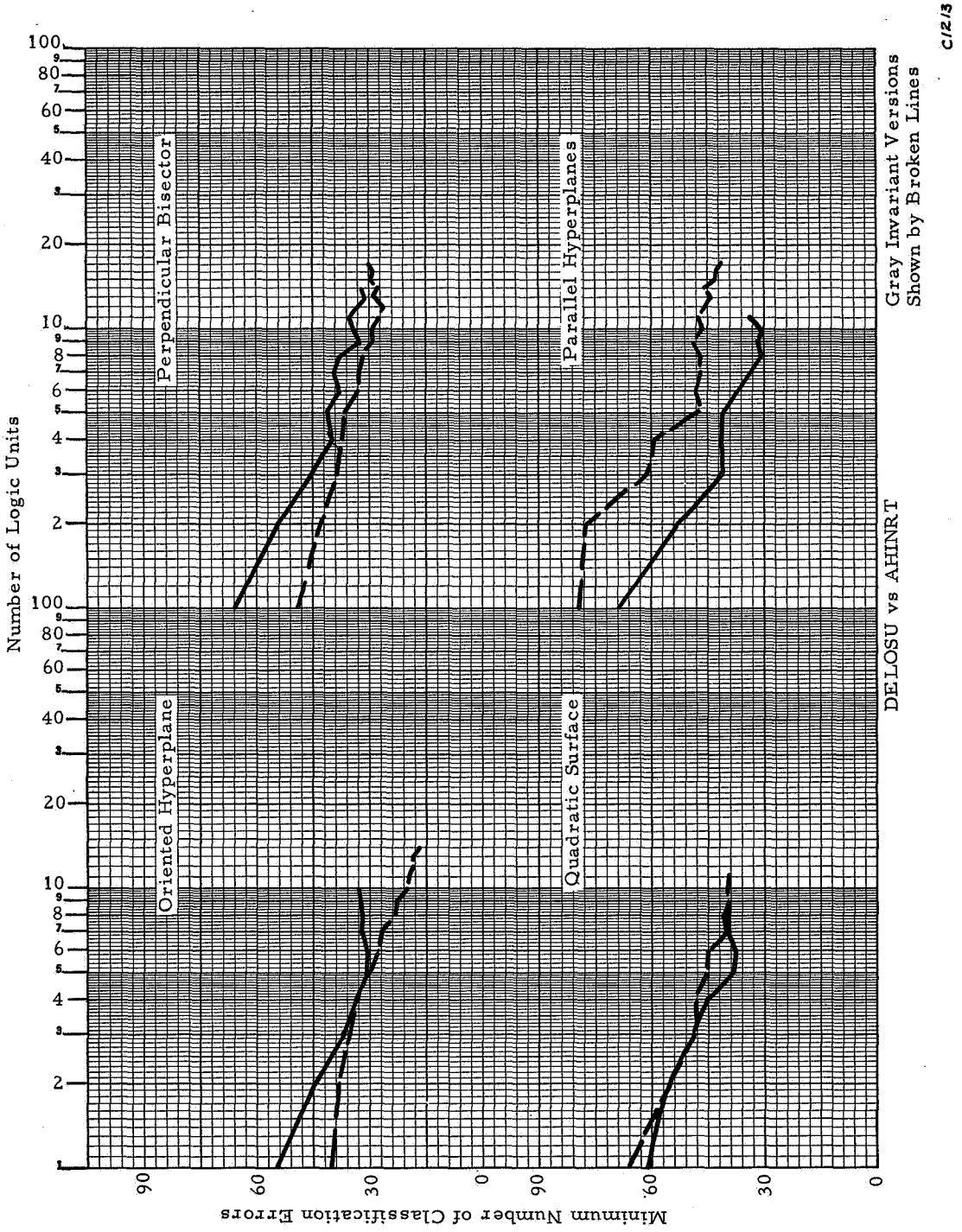


Figure 45. Effects of Gray Scale Patterns – Deck II, VAR

The system loss data of Figure 37 is replotted in Figure 46 to facilitate the comparison of the effects of the distributional assumptions. This comparison is not deemed to be of great importance here, as the sample problem on alphabet characters does not have the same structure as does the cloud picture interpretation problem. It is of interest to note, however, that the effects of the distributional assumptions do not particularly manifest themselves in the rate at which the sampled patterns are learned. There is enough information in these patterns for the classification of the sample patterns regardless of whether the differences in the mean vectors or in covariance matrices are exploited. The effects must be sought in the generalization data, where the spurious correlations which result from the limited number of "learning" samples used (and which can be used to classify the learning samples) are of little help.

5.4.5 Technique Complexity

There are two techniques for each of the two distributional differences exploited. One of these two techniques offers greater simplicity than the other.

When the difference in mean vectors is to be utilized, the techniques are "Oriented Hyperplane" and "Perpendicular Bisector." Both techniques lead to linear input threshold units, and hence the resulting networks are equally easy to implement. However the "Perpendicular Bisector" method yields a considerably simpler design routine, as there are no covariance matrices to estimate or invert.

The techniques which capitalize on differences in the covariance matrices — the "Quadratic Surface" and "Parallel Hyperplane" techniques — provide equal computational difficulties. The implementation of the "Parallel Hyperplane" networks, even in its gray invariant version is much simpler than the "Quadratic Surface" machines.

As might be expected, there is a penalty for simplification. Figure 47 represents the data of Figure 37 replotted to emphasize

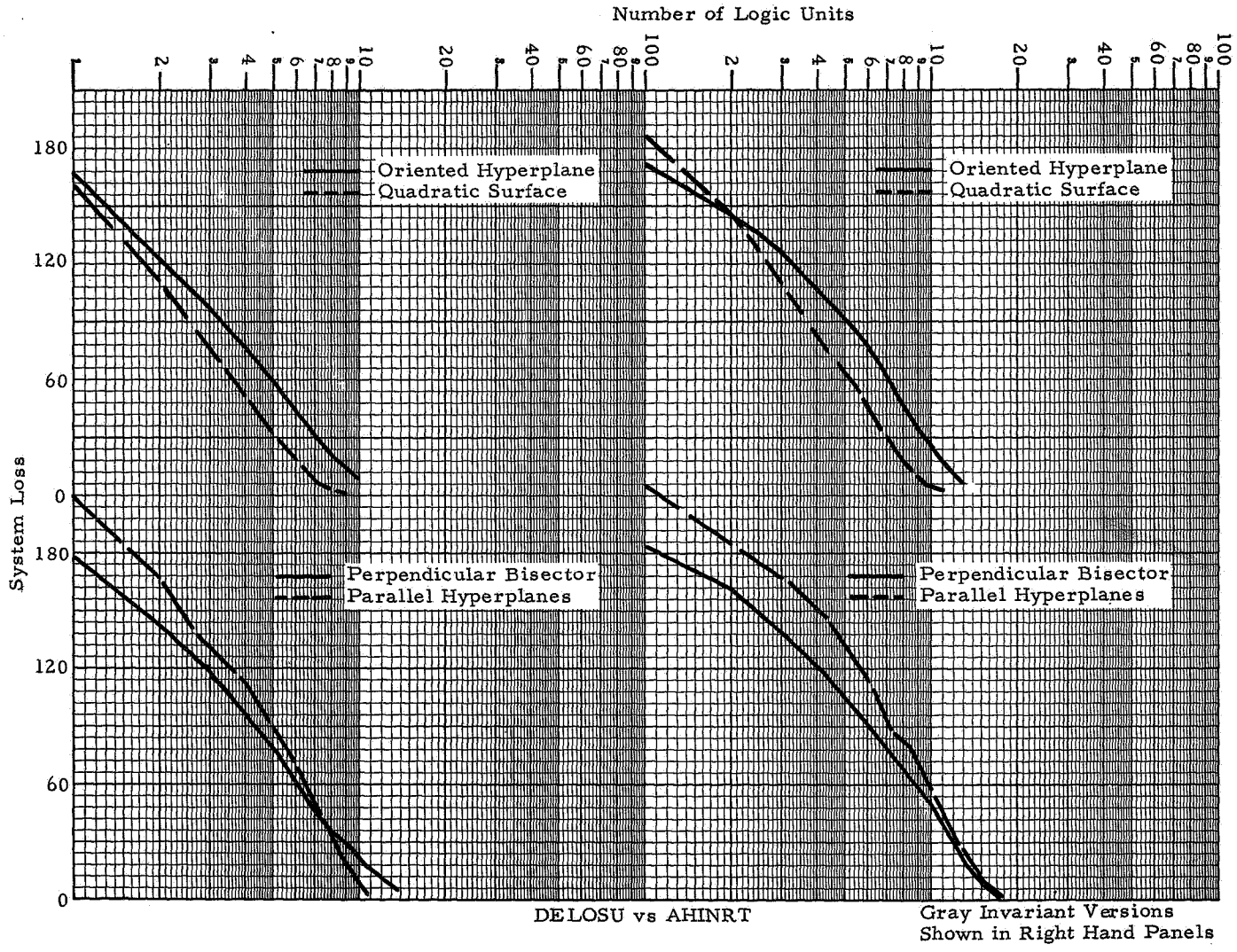


Figure 46. Effects of Distributional Assumptions — System Loss

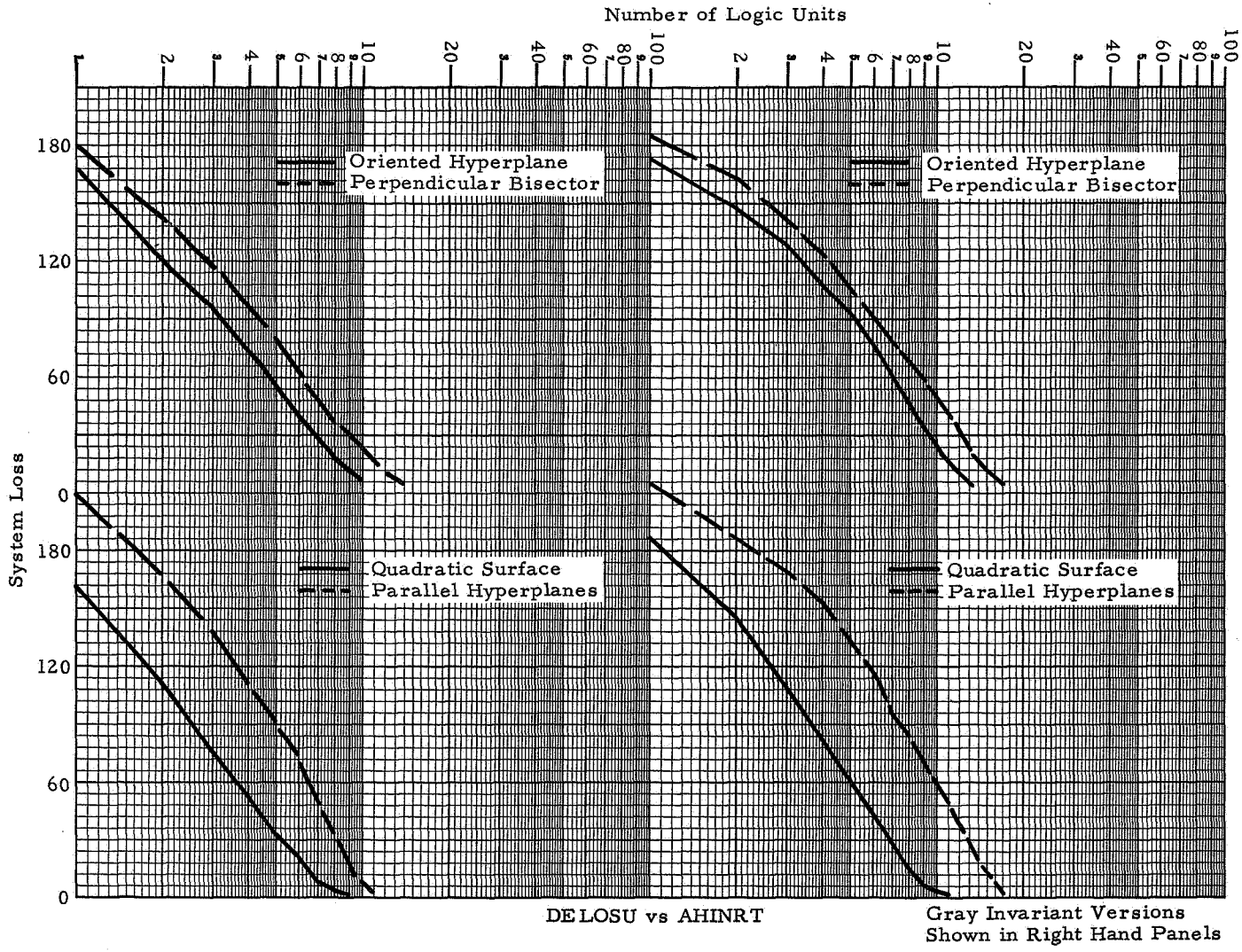


Figure 47. Effects of Technique Complexity — System Loss

the technique complexity comparisons. The result is remarkable in its uniformity. The penalty for simplicity is 20 to 25 percent in efficiency.

Each of the simplified techniques throws away information. In the case of the "Perpendicular Bisector" technique, design time (for equal machine sizes run) is only half that of the "Oriented Hyperplane" technique in the noninvariant forms, and one third in the invariant forms. The net gain is thus a 2/3 to 1/2 reduction in computer time. For the "Parallel Hyperplane" it may be seen in Section 4.7.2 that implementation even of the larger network required is more economical.

Figures 48 through 51 present the data for a more difficult classification of the letters – A, D, E, I, L, and R in one class, and H, N, O, S, T, and U in the other. The data are for individual networks, not "average" ones. Two of the machine designs did not classify all of the sample patterns correctly with 20 logic units.

The comparisons drawn from these data do not evidence the sharp differences of the earlier data. Whether this is due to the individual runs again providing random variations marking the actual effects, or whether the different grouping of patterns produces a different underlying distribution, and hence different actual effects is not known at this time.

These data show only a small penalty is incurred when the simplified techniques are used. Generalization error rates are 20 percent. That they should be higher is not too surprising, as not many people have scored too highly on the D and O discrimination required in this classification (Figures 18 through 27). In this case, however, there seems to be little difference in the generalization capabilities of techniques based on mean vectors and techniques based on covariance matrices. This tends to support the contention that the different grouping of patterns results in different underlying distribution differences.

5.4.6 Random Disturbances

A limited investigation has been conducted on the sensitivity of network designs to the accuracy with which they are

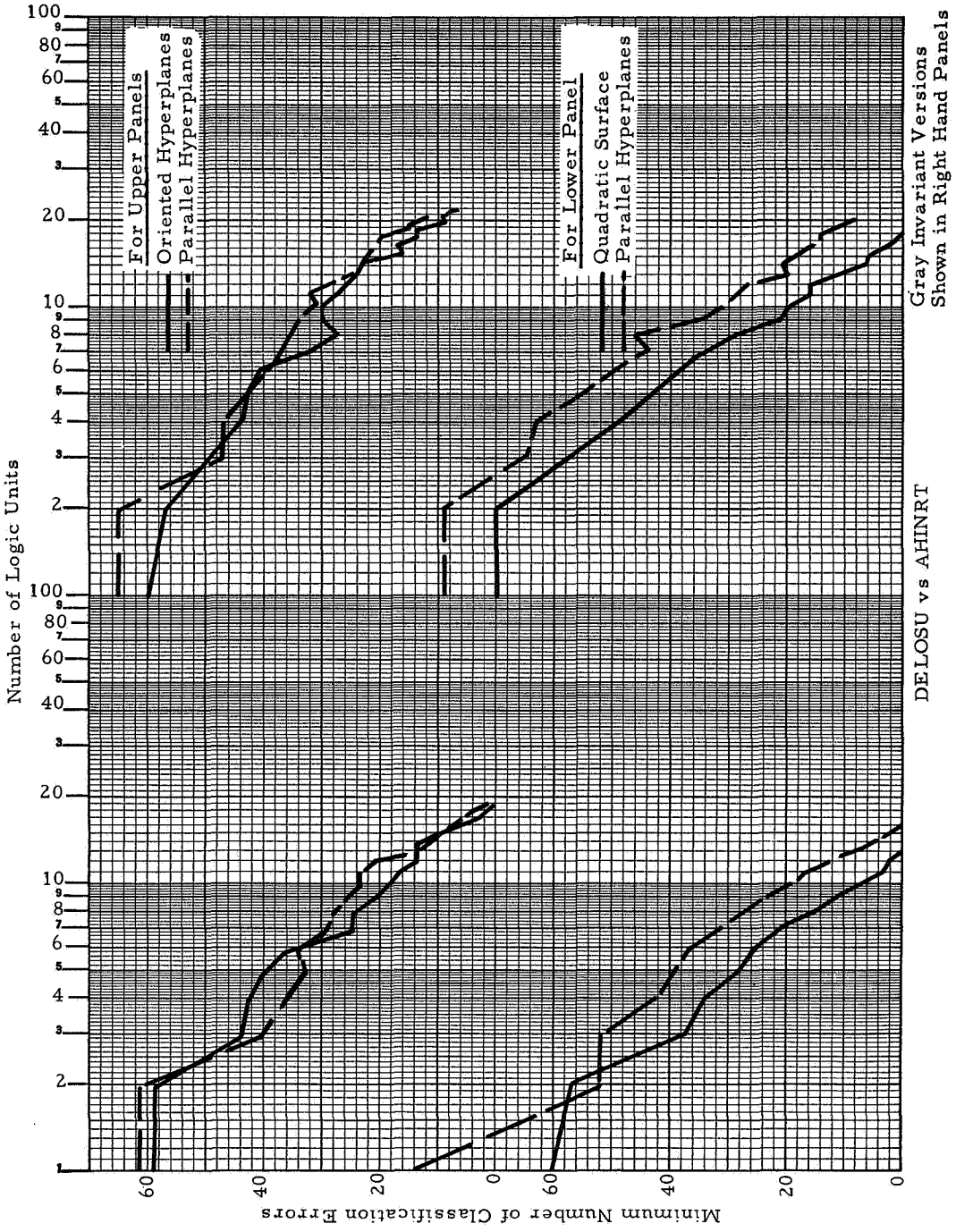


Figure 48. Technique Complexity Comparison — Learning Rate

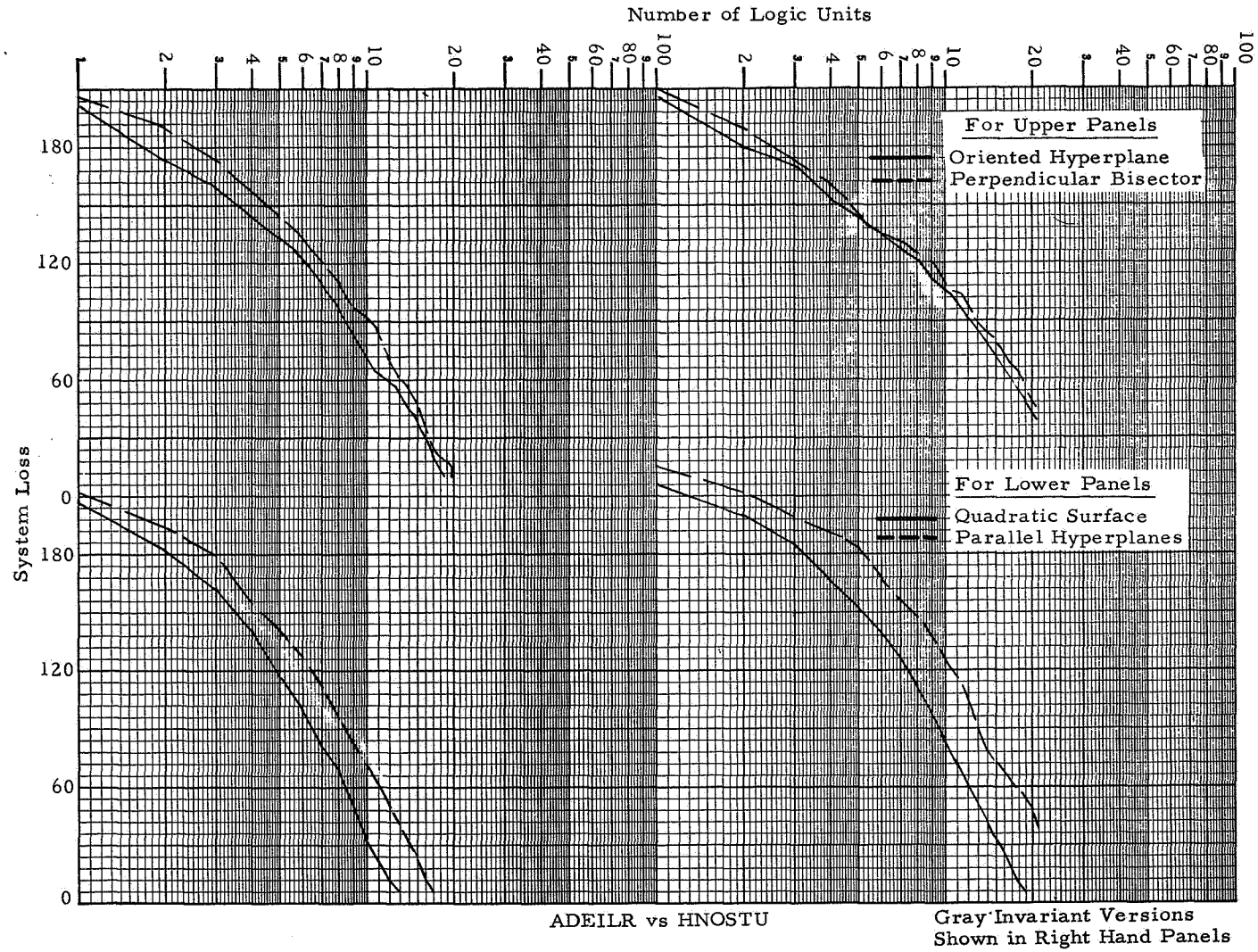


Figure 49. Technique Complexity Comparison — System Loss

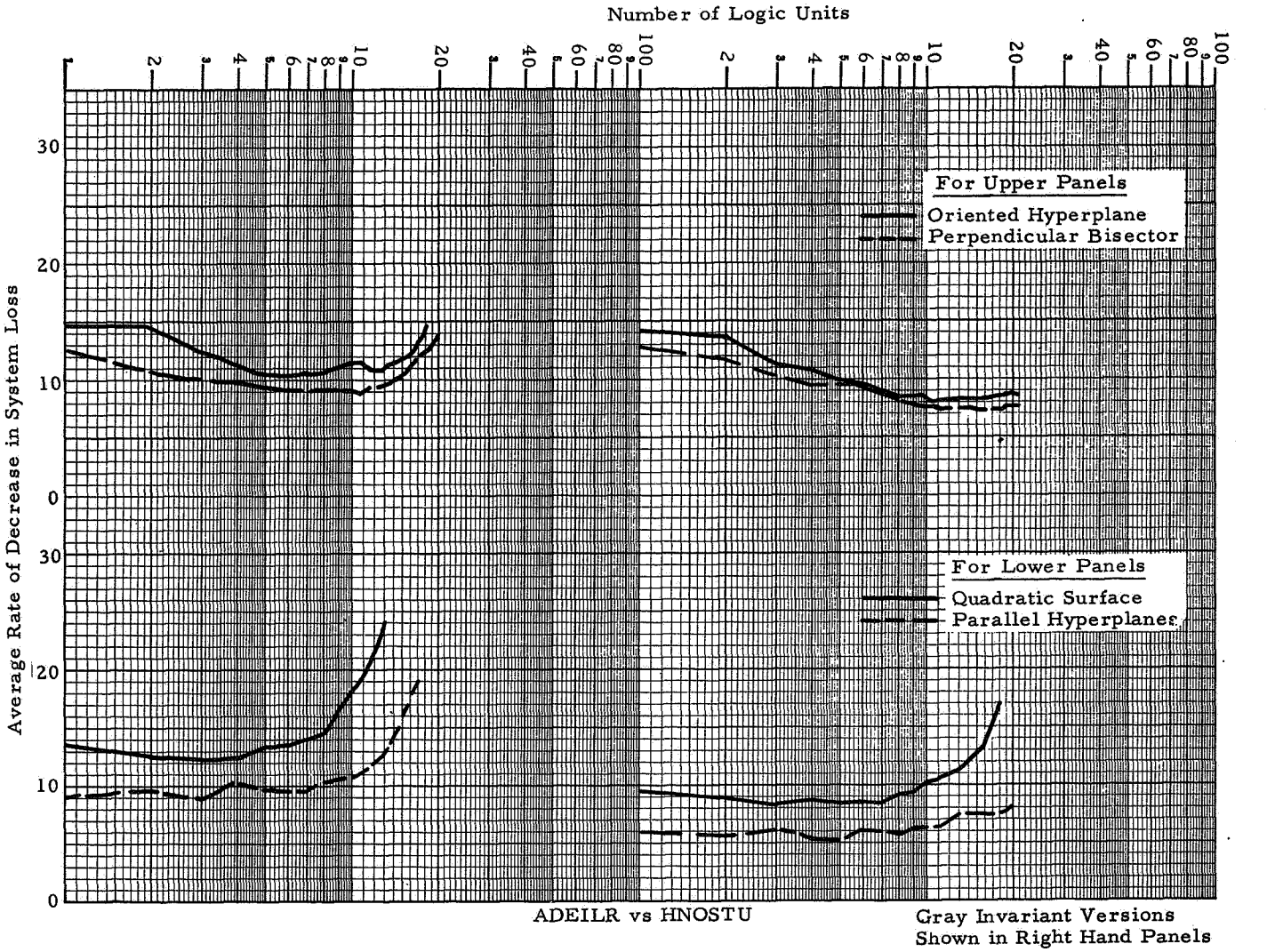


Figure 50. Technique Complexity Comparison — Rate of Decrease in System Loss

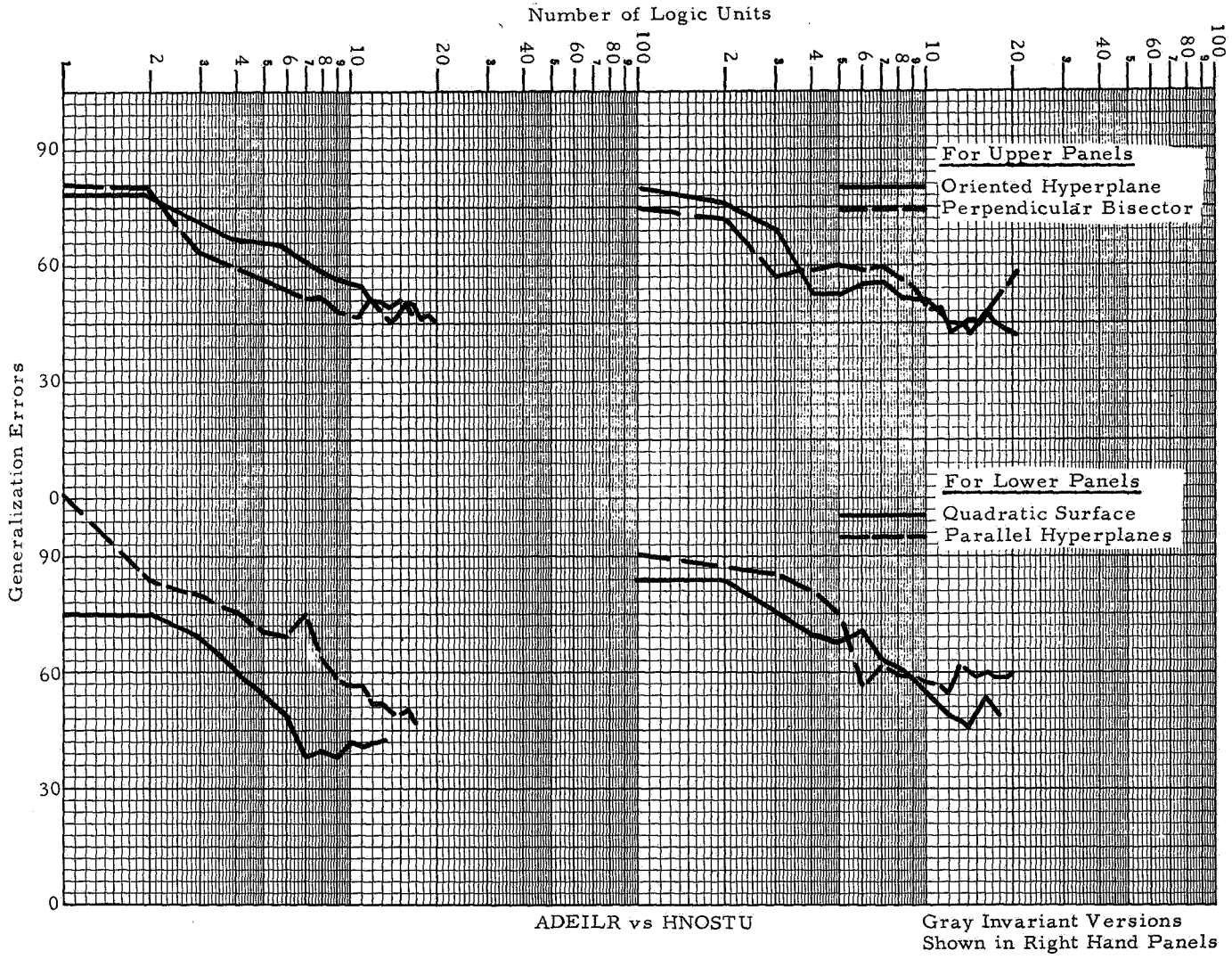


Figure 51. Technique Complexity Comparisons — Generalization Errors

implemented. The results are based on a single network designed using the "Parallel Hyperplane" technique.

Figure 52 presents the performance data of the original network. Two other sets of data are presented, each representing the average results of two networks.

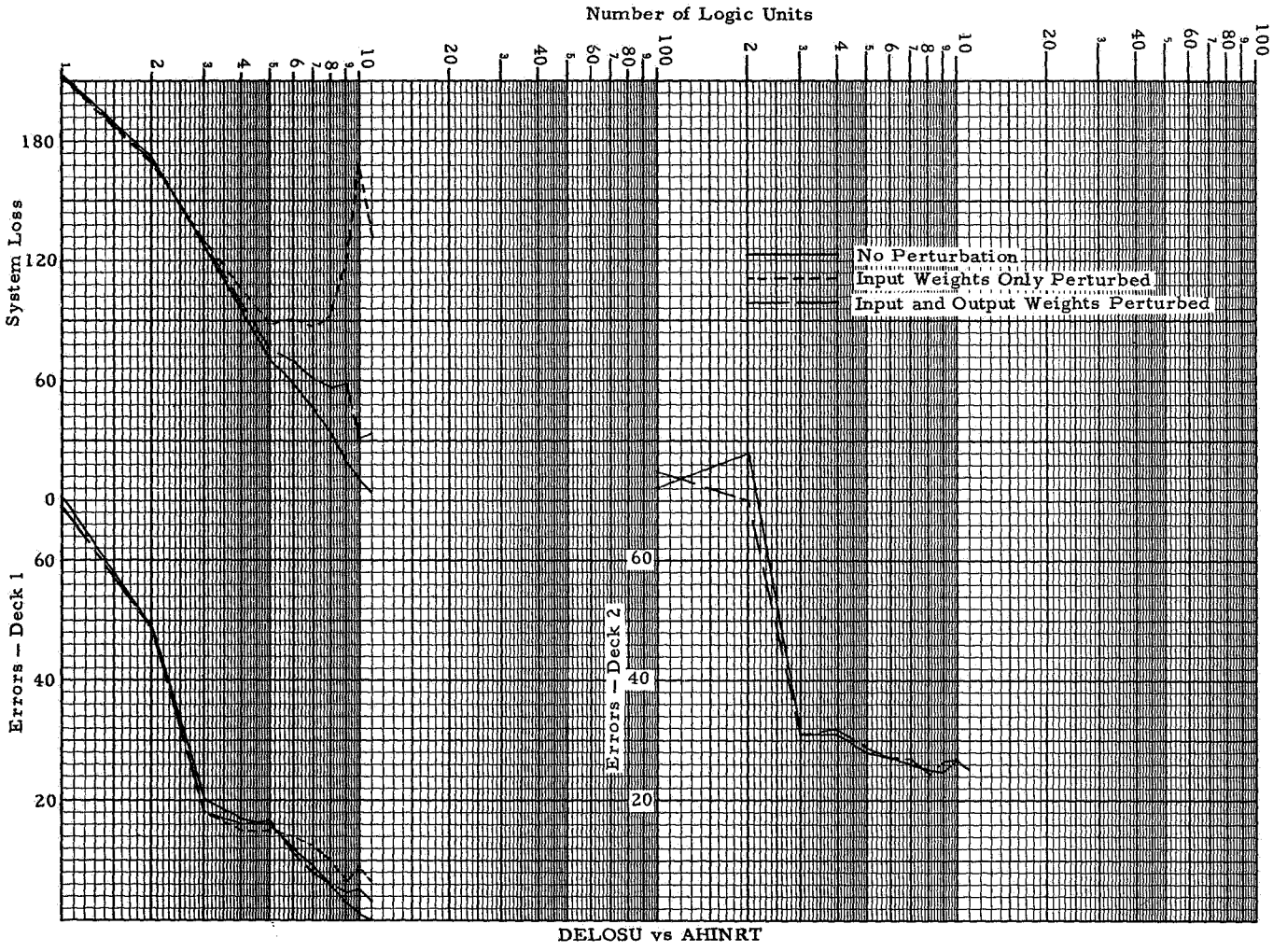
Perturbations are applied to various weights specified in the system. A perturbation is generated as a normally distributed variable with zero mean and standard deviation equal to five percent of the value being perturbed. To obtain one set of data, only the input weights to the logic units were perturbed. As units are added to the network, perturbations applied to earlier units are not changed. To generate the second set of data, the input weights to the logic units are perturbed as above, and in addition, the output weights of the unit are also perturbed. As units are added to the system, the output weights of all earlier units are disturbed anew, since these weights do not remain constant as the network is designed.

The data show that although there is a strong effect on the system loss caused by a disturbance of the linear discriminant function of one or two patterns, these perturbations do not seem to effect the error rates seriously, either on the sample patterns or the generalization patterns.

5.4.7 Standard Perceptrons

In order that a reference could be established for evaluating these techniques, several standard perceptron techniques were used to generate networks for the DELOSU vs AHINRT problem. Three options for specifying logic units, and two options for specifying output weights were used. Six techniques were thus effected.

Logic units could be generated by randomly specifying subspaces and connection weights, by the "Perpendicular Bisector" discriminant analyses (in which unequal pattern weighting is not employed), or by "stealing" them from any network designed by the



c0582

Figure 52. Effects of Random Perturbations – Parallel Hyperplane Technique

"Oriented Hyperplane" or "Perpendicular Bisector" iterative design techniques. These three options are called "Random Units," "Discriminant Analyses Units" and "Prepared Units" respectively. No selection is utilized with the first two options. In these first two options, the network size is limited to 50 units by the capabilities of the verification program; in the latter option, by the size of the network pirated. A "Perpendicular Bisector" run of 15 units was used for the "Prepared Units."

The two output weight options were the "forced learning" method, in which the weight of an output connection is proportional to the difference in the rates at which the unit turns on for patterns of the two classes.

$$w_i = k\,(p_i - r_i)$$

and the "logarithmic weight" method in which the weight is given by

$$w_i = k\left(\ln \frac{p_i}{1 - p_i} - \ln \frac{r_i}{1 - r_i}\right)$$

Two networks were designed for each of four cases by choosing different random starting numbers for subspace selection:

1. Random units-forced learning (RF)
2. Random units-logarithmic weights (RL)
3. Discriminant analysis units-forced learning (DF)
4. Discriminant analysis units-logarithmic weights (DL)

The same pair of random numbers were used in each of the four cases. In the Figures, the choice of random number is coded by a "1" or "2" (as in RF2, for example).

Two networks were obtained for the other two techniques

5. Prepared units-forced learning (PF)
6. Prepared units-logarithmic weights (PL)

in a slightly different fashion. A set of units prepared by the Perpendicular

Bisector technique were used for one set of networks (designated PF and PL), while a set of units prepared by the gray invariant version of the Perpendicular Bisector technique provided the other set of networks (designated IPF and IPL).

The twelve networks were designed on Deck 1 (as were the prepared units) and tested against Deck 2. Figure 53 shows the results for these designs. Results with the forced learning and logarithmic weight options appear to show little variation. "Discriminant Analyses Units" provide a substantial improvement over "Random Units," although it appears that one might expect the asymptotic performances to be similar if sufficient units were added. The "Prepared Unit" option provides a considerable contrast. Even with output weights generated by standard perceptron techniques, the performance is good. The generalization errors rate approaches that of the original design. The value of the "discriminant-analyses/iterative-design" technique in developing the logic units is dramatically demonstrated. Figure 54 illustrates the comparison between the learning rates and the generalization rates for these perceptron networks.

It can be concluded that there is little difference between the forced learning and logarithmic weight techniques. It is also clear that the use of discriminant analysis to provide units leads to more efficiency than random selection (although terminal performance may not be better), and that the feedback in the discriminant-analysis/iterative-design approach provides a considerably better set of units than either of these cases.

5.5 Summary of Results

Only the results of the work reported in Section 5 will be summarized here.

1. A study of a simplified problem, alphabetic character recognition, permitted a much broader investigation as the computer time

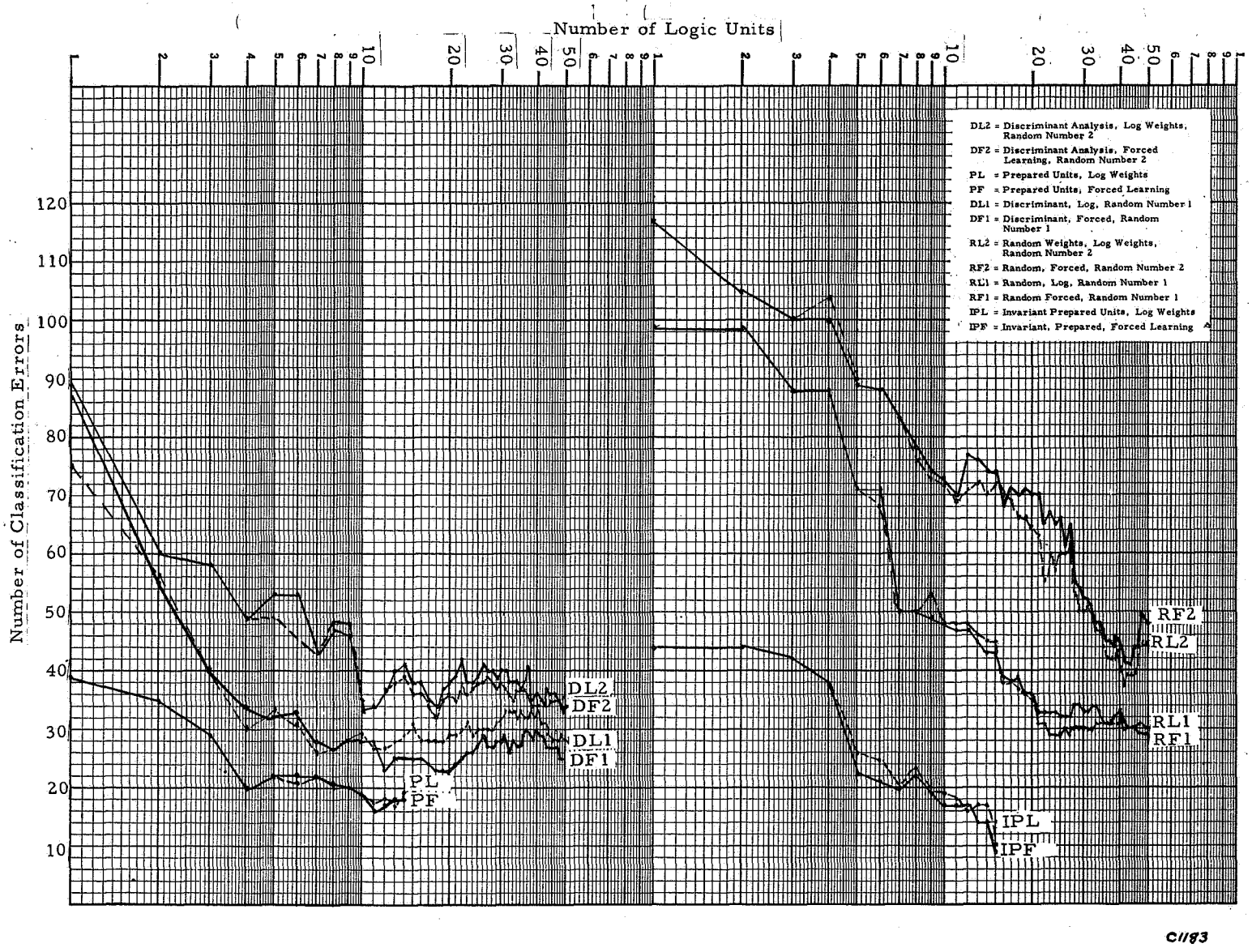


Figure 53. Learning Procedures Applied to Deck I

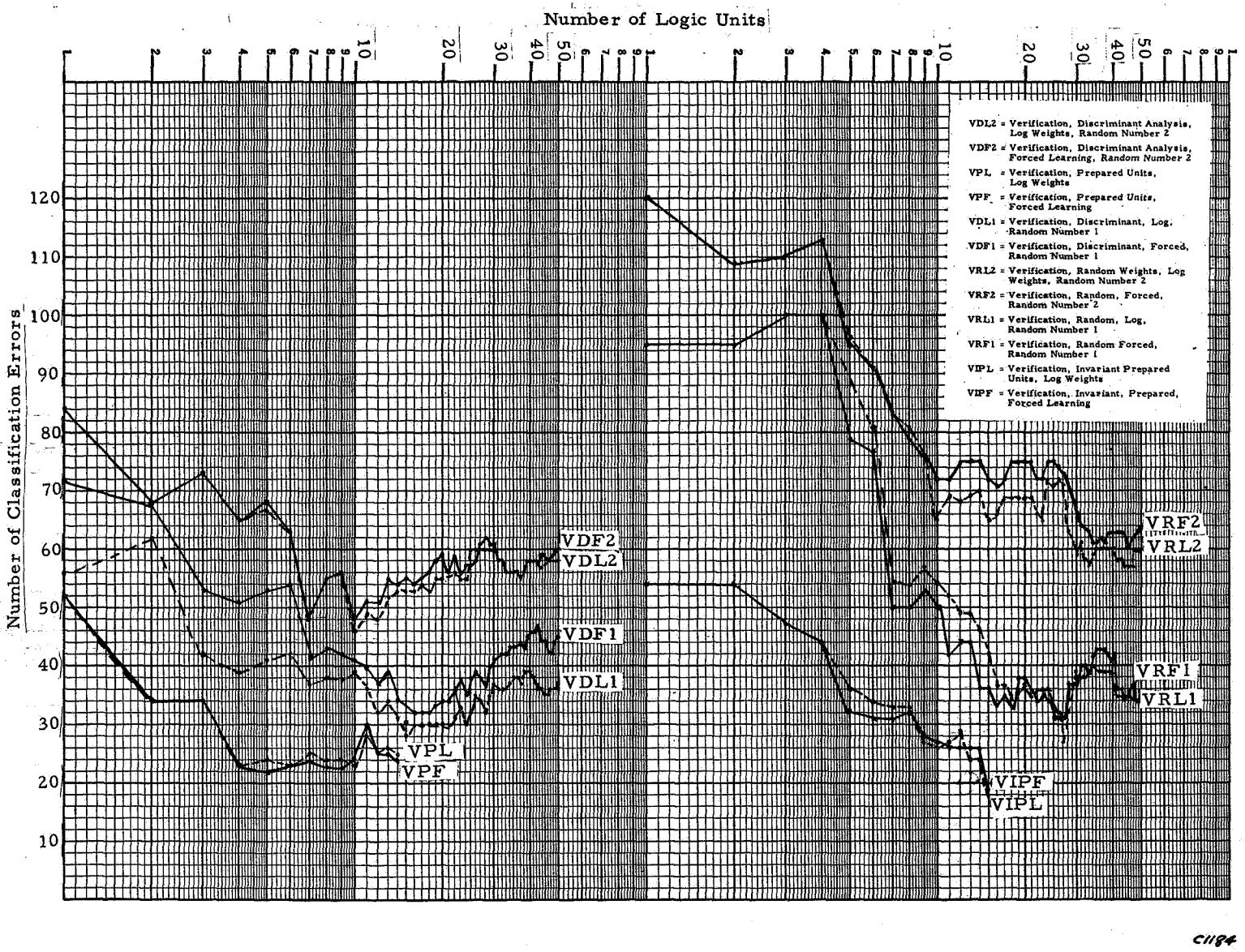


Figure 54. Generalization with Deck II

required for a single network design averaged 1/60 that required with the actual TIROS frames. The investigation was for the most part restricted to problems concerning the design technique itself, rather than its application to a specific problem. Eight variations of the basic design technique were studied, these differing in the means for generating property filters. There are four basic methods for designing property filters, and four variations on these to produce units invariant to changes in the gray scale. Two of the basic methods exploit differences in the means of the distributions of measurement vectors for the pattern classes, one method representing a computational simplification of the other. The other two basic methods exploit differences in the covariance matrices of the distributions, one method permitting much simpler implementation than the other. Several "open" perceptron techniques and a correlation technique were studied to provide a basis for evaluation.

2. All eight of the techniques were very effective in producing small networks giving perfect performance on the sample patterns. In general, 10 to 15 property filters were required for one binary decision on the 240 patterns. This rate of design is too fast, and emphasizes the weakness in using performance on only a limited sample of patterns (the 240 samples represented 12 different letters) as the design criterion. Any false correlations, which are due to nonrepresentativeness of the sample, (for example, a larger proportion of vortex patterns with horizons on the left side of the picture) may solve part of the problem of classifying sample patterns, but do not contribute to the solution of the real problem. In part, these false correlations may be detected by comparing the error rates achieved on the sample patterns used in the design with the error rates on an independent sample. When these false correlations are prevalent, one should increase the sample size, or make stronger use of distributional assumptions. For partial relief, several steps to slow down the design rate are considered in Section 6.0. A slower rate allows more opportunity for useful properties to be found before portions of the problem are considered solved on the basis of extraneous correlations. The presence of the false correlations is also

indicated in a comparison of the results with the property filter generation methods based on differences in the means of the distributions, and those based on differences in the covariance matrices. Although little difference is shown in the rate of design, the ability of the networks to generalize to the independent sample is considerably different.

3. There is considerable variability in the design, due to randomness in the selection of subspaces for the logic units. This variability tends to mask differences which occur in the experiments performed. At least three networks were designed with each method, and the results averaged, to reduce the variability. The variability is no doubt due to the speed of the design, as it is not noticeable in the experiments in Section 6. In part, the variability is due to the high selection ratio used (the ratio of the number of property filters generated to the number incorporated in the network). Since the subspaces are generated randomly, the evaluation of a unit to be generated may be regarded as a random variable. With a high selection ratio, only the tail of its distribution is being used, a situation which induces variability. High selection ratios may be undesirable since units which are selected are usually of high utility for the classification of the sample patterns. In a complex problem, however, it is unlikely that a single property filter could solve a large portion of the real problem. High ratios could thus put a premium on the extraction of extraneous correlations.

4. Despite the difficulties discussed above, the design techniques worked comparatively well on the sample problem. The final network designs work considerably better, both on the sample patterns used in the design process and the independent sample, than did networks three to five times as large, designed using two "open" perceptron techniques. The utility of the current technique appears to be in the design of a set of property filters. In one experiment, the linear decision function of a network designed with the current technique was reassigned using the perceptron techniques. The resulting network classified the independent sample about as well as did the original network. A

correlation method was also used. The entire set of sample patterns were stored, and unknown patterns classified by finding the highest correlation coefficient (the second highest when the sample deck was used for testing). This method showed the least effect of spurious features correlating highly with a pattern class. The error rates on the design sample and the independent sample were virtually identical. The generalization error rates were the lowest achieved. The best networks designed with the current technique (DAID) came very close to the generalization error rates of the correlation scheme.

5. When a parameter is available for such an adjustment (as in the gray invariant version of the parallel hyperplane technique), a property filter activity rate of about 50 percent seems to give the best results. The adjustment is not critical, however.

6. The number of freely variable weights in a network appears to be a critical quantity. Within the range investigated, the effect of the number of input connections per property filter may be accounted for by this quantity. For the linear input property filters, the number of free weights varies linearly with the number of connections. Thus a network of seven logic units, each with seven input connections could work about as well as one with eight units each with six connections. For the quadratic input units, the number of free weights varies quadratically with the number of input connections.

7. The cost of making the property filters invariant to linear changes in the gray scale may also be accounted for by the number of free weights. It can be shown that the constraints which produce the invariance reduce the number of free weights by two per property filter for the linear input units, and by the number of input connections plus one for the quadratic input units. In both cases, the loss is equivalent to the loss of one connection plus one weight. The experimental results confirm these predictions. The value of the gray scale invariance is strikingly demonstrated by the generalization error rates for samples

in which the gray scale has been varied both linearly (as per the invariance) and nonlinearly.

8. The cost of using the simplified techniques appears to be an increase of 30 percent in the number of property filters needed when differences in the mean vectors are being exploited (i. e., the Perpendicular Bisector and Oriented Hyperplane techniques), and an increase of 50 to 75 percent when covariance matrix differences are being used (the Parallel Hyperplanes and Quadratic Surface techniques). In the latter case, the simplified units use less than half as many free weights, so that the total network has fewer free weights in the simplified case.

6. STUDIES ON CLOUD PATTERNS

6.1 Topics

A number of investigations were carried out on actual TIROS frames. With the exception of part of the resolution study, these experiments involved the design of recognition networks. Since each network design required six hours of IBM 7094 time, the studies were limited to topics not suitable for the alphabetic character investigations. A total of seven complete networks were designed. In addition, four partial networks were obtained, each requiring 1.5 hours of computer time. A total of about 30,000 property filters were designed, and about 2,000 of these were retained in the eleven network designs.

Seven topics were studied in the investigations on cloud patterns. These topics and the subsections discussing the results are:

1) Design rate (6.4.1)---due to a peculiarity of the program, instability results when a parameter which controls the rate of design is too small (too fast a rate). Fast design rates also permit the classification due to spurious clues and reduce the chance of finding good properties. How slow must the design rate be to avoid instability?

2) Activity rate (6.4.2)---when a parameter is available for its control, what rate of activity of the property filters is most suitable?

3) Separability of the sample patterns (6.4.3)---can networks be designed to separate the 1,000 sample patterns or not, and how complex a network is required?

4) Generality of design (6.4.4)---can the networks designed on the sample of 1,000 patterns classify an independent sample of patterns? This study is one of the most important phases of this program.

5) Continuation of design (6.4.5)---network design ceases when perfect performance is obtained on the sample patterns. Can generalization performance be improved by combining several networks, thus providing a mechanism for design continuation?

6) Input fields (6.4.5)---does restricting the area from which each property filter may receive input connections (i.e., localizing properties) improve performance?

7) Resolution (6.5)---what resolution is required to classify vortex patterns? This study was conducted in two parts, by optically altering the resolution of a number of photographs and examining the resulting photographs visually; and by altering the resolution of the sample patterns and designing recognition networks.

6.2 The Sample Problem

6.2.1 The Design Technique

The design technique considered in this section is based on the gray invariant version of the Parallel Hyperplane technique described in Section 4.5. Those modifications which were made are designed to minimize the increase in computer time which results when the sample patterns cannot all be held in core memory and to reflect the fact that the sample cloud patterns is not a complete set of possible patterns. The individual logic units produced were not permitted to become absolute indicators of the pattern class by the assignment of infinite weights.

In the discriminant analysis-iterative design technique, it is necessary to have access to the entire file of sample patterns two or three times (depending on the memory available) in the course of designing a logic unit for the network. Since the reading time of magnetic tape files of the digitized Tiros frames is six minutes, the following compromises are used:

1) 150 candidate logic units are designed at one time. 10 of these are selected for inclusion in the network.

2) The design process if folded. In the first pass through the sample pattern file, mean vectors and covariance matrices are estimated for 150 subspaces located in Block A. After the first pass, these data are converted logic unit specifications, and the resulting logic units transferred to Block B. In the next pass through the pattern file, data are collected for a new set of subspaces in Block A, and four numbers accumulated for the units in Block B. These numbers are the sums of the pattern losses for vortex and nonvortex patterns, when the unit is active or not active. After the pass, the units in Block B are evaluated, the best 10 chosen, output weights assigned to these units, and the selected units transferred to Block C. The data in Block A is treated in the same fashion as in the first pass. In subsequent passes, the

losses of the sample patterns are affected by those units selected for the machine (Blocks C and higher). The output weights of the logic units selected are iterated in Blocks C and higher, and the processing in Blocks A and B is as before.

3) Block E is the highest block. The weights of units selected for this block are changed twice from their initial values. In retrospect, it might have been better to store activity tables for the selected units, and perform the iterations between passes, rather than computing the activities anew on each pass. Once the units pass through Block E, the pertinent information is recorded on magnetic tape, and the units are no longer considered in the design run.

In the relaxation process, the weights are not adjusted to exactly minimize the total loss, as was done in the Section 5 experiments on the alphabetic patterns. A lesser correction was used. This was accomplished in the following manner. Four quantities are accumulated for a logic unit. Let

AV = total loss of sample vortex patterns for which the unit is active

AN = total loss of sample nonvortex patterns for which the unit is active

IV = total loss of sample vortex patterns for which the unit is inactive

IN = total loss of sample nonvortex patterns for which the unit is inactive.

The system loss is minimized if the contribution of this logic unit to the decision element threshold is

$$\frac{1}{2} \ln \frac{IN}{IV} \tag{19}$$

and the weight of the unit is

$$\frac{1}{2} \ln \frac{IN}{IV} - \frac{1}{2} \ln \frac{AN}{AV} \tag{20}$$

If one or two of these quantities is zero (other than the AN-IN or AV-IV pairs), then infinite values result. The logic unit would then be an

absolute indicator for certain patterns. This would be acceptable if all patterns of interest were in the file of sample patterns. When the file of classified patterns is a sample, it is undesirable to have absolute indicators. To prevent infinite values, a quantity "β" is added to AV, AN, IV, and IN before computing the values of (19) and (20). An excellent discussion justifying such a change is given by Mosteller and Wallace.[16] In this program, "β" was chosen to be a fraction of the total system loss.

During the debugging operations, two difficulties arose which were traced to the manner in which the output weights of the units are iterated. The quantity "β" was involved in both of these difficulties. At first, the fraction used for "β" was small (.004), and caused the quantities (19) and (20) to be only slightly different from optimum. The first time units were in Block C, these small effects were compounded over ten units, resulting in the total loss for vortex patterns being five-eighths the total for nonvortex patterns. In the following pass, each of the twenty units in Blocks C and D attempted to correct this unbalance, resulting in an excessive swing in the opposite direction (the vortex total was then 8,000 times the nonvortex total). This was fixed by applying a threshold correction after each pass, modifying the pattern losses to assure equal totals, and appropriately adjusting AV, AN, IV, and IN.

The second difficulty arose in a test pass in which the third block of ten units chosen for Block C, although distinct, had approximately the same activity vectors for the sample patterns. Since the weights for all of the units are calculated before any of the pattern losses are affected, each unit attempts to correct the same imbalance. As a result, the total correction applied by the block overshoots the optimum by a considerable amount. In the next pass, each of these units corrects the overshoot, producing an unstable oscillation until these units have passed through all of the blocks. This situation does not seem to arise too often, at least in the earlier phases of the network design. The problem was fixed for this particular case by choosing a larger value

of "β". This has the effect of making each weight adjustment smaller. The tendency to overshoot is thus reduced. "β" was chosen so that the units could easily achieve output weights of the magnitudes observed in three steps. "β" was taken to be seven tenths of the total system loss.

Both of the above difficulties would have been avoided if the activity vectors had been stored, and the adjustment for each unit applied before computing the correction for the next unit.

6.2.2 The Sample Cloud Patterns

Twelve hundred sample cloud patterns were required for the investigation. One thousand of these were used as a design sample to provide the statistical information needed to design pattern recognition networks, and two hundred were used as an independent sample to test the networks so designed for generality. Since the networks have no mechanism for rotational invariance, each rotation of a pattern to any appreciable extent is in essence a new pattern. The 1,200 patterns were obtained by using four rotations of each of 300 basic patterns. Again, as the networks have no mechanism for translational invariance, in a number of instances several basic patterns were derived from one TIROS frame by choosing different edit points. Half of the patterns used were vortex patterns, half were nonvortex. A total of 223 distinct TIROS frames were used, 110 of them representing vortex patterns.

The selection was started with a study of the catalogues of TIROS cloud photography issued by the Weather Bureau. The following catalogues were used:

Catalogue of TIROS V Cloud Photography for June 1962
(Passes 001 through 164)

Catalogue of TIROS V Cloud Photography for July 1962
(Passes 168 through 601)

Catalogue of TIROS V Cloud Photography for August 1962
(Passes 608 through 1055)

Catalogue of TIROS V-VI Cloud Photography for September 1962
(TIROS V - Passes 1054 through 1483)
(TIROS VI - Passes 001 through 183)

In these catalogues the photographic films are listed according to pass number. The last column in the list is entitled LVBSC. The V in the second position stands for VORTEX, and a 3 in this position indicates that there is a pronounced vortex visible on at least one of the photos transmitted during the pass shown in the first column of the list.

Passes showing a 3 in the V column were registered and the associated film rolls were ordered from the TIROS Data Utilization Manager, NASA Goddard Space Flight Center. A sufficient amount of film material was obtained to enable the selection of the required patterns.

The frames on the rolls were then investigated, one by one, and cloud negatives showing the familiar vortex patterns were selected. For each negative showing a vortex pattern, a second complementary negative, showing no such pattern was selected. Care was taken to obtain, whenever possible, the complementary negative from the same orbit, with approximately the same percentage of horizon (more accurately, space) coverage and generally the same overall cloud density and illumination intensity as the vortex frame. Frames showing excessive horizon area were excluded, as were frames showing patterns that could not clearly be classified.

The presence or absence of vortex structures on the selected frames were then confirmed by consulting meteorologists. One hundred of these frames were previously used on Contract NASw-609 and confirmed by Dr. Neiburger, consultant meteorologist from UCLA. They were incorporated with other TIROS V and VI frames for use on this contract. The remaining frames were verified by Mr. H. W. Van Dyke of the Weather Bureau Satellite Station at Pt. Mugu on 22 July 1964.

The 367 selected frames were then requested in digitized form from the Institute for Space Studies in New York. A set of prints of the requested frames were received from the technical

monitor, Mr. J. Silverman. A comparison of the negatives, the prints, and of SC-4020 printouts of the digitized TIROS frames indicated that the negatives from which the frames were ordered had had their panels corrected, the changes not being indicated in the TIROS catalogs. The prints and the digitized tapes were uncorrected, however, so that only 116 of the 275 frames received corresponded to the frames requested.

Additional frames were selected from the TIROS negatives,* and from the Stanford vortex atlas.[22] A total of 505 digitized frames have been received. A significant number of these are duplicate frames, incomplete frames, or very noisy frames. Most of the frames received, and all of the frames utilized in the study have been verified by a printout using the S-C4020.

The frames actually used to provide the sample patterns for designing the networks are listed in Table IX. The frames employed as an independent sample to verify the networks are listed in Table X.

6.2.3 Modification of the Sample Frames

A number of modifications were made to the sample frames selected. As received from the Institute for Space Studies, the patterns are represented by a 240 by 234 array of six bit numbers. The data are packed so that one word contains the brightness levels of six points. The first modification was to reduce the gray scale to four bits, and repack the data to nine points per word. In the original data, virtually no points have gray levels below 8 or above 40. The scale was accordingly changed to five bits by subtracting 8, setting any negative values to zero, and any values above 31 to 31. The low order bit was then dropped to give the four bit scale.

Sample patterns for the study were obtained by selecting a 180 by 180 portion of the array, a process termed editing. The editing was performed in such a way as to minimize the amount of horizon in the edited frame. Seven horizontal and seven vertical edit

TABLE IX

TRAINING SAMPLE

Vortex or Nonvortex	TIROS Number Sequence Number Frame Number	Pass Number	No. of Edit Points	Edit Point Location
CP2N	52 3	17000001		B2
CP2N	52 4	17000002		B1E6
CP2N	5216	28000001		B3
CP2N	5217	28000002		A7D4
CP2N	5218	28000002		E3B6
CP2N	5220	28000002		F6B2
CP2N	5221	28000003		A1E3G5
CP2N	52 1	62000001		D3
CP2N	52 2	62000002		A1D1
CP2N	5318	62000002		A3F6
CP2N	5319	62000002		D6A1
CP2N	5121	144000002		F1A5
CP2N	5122	144000002		A7D2
CP2N	5123	144000002		G6D1
CP2N	5124	144000002		E1A5
CP2N	5126	144000002		A5C4
CP2N	5222	144000002		A1G1
CP2N	5111	145000001		G1
CP2N	5112	145000001		G3
CP2N	5113	145000001		G7
CP2N	5114	145000002		A1D5
CP2N	5115	145000002		A4G1
CP2N	5211	145000002		B1F1
CP2N	5212	145000002		A6G6
CP2N	5214	145000002		B6D4
CP2N	5215	145000002		G2G5
CP2N	5114	160000001		E6
CP2N	5214	160000002		A7B3
CP2N	51 4	666000001		A6
CP2N	51 5	666000001		G4
CP2N	51 6	666000001		G1
CP2N	52 7	667000002		E3G6
CP2N	52 8	667000001		G7
CP2N	52 9	667000001		G1
CP2N	5210	667000001		G4
CP2N	5212	667000002		G6C3
CP2N	5218	667000001		G3
CP2N	5219	667000001		B7
CP2N	5220	667000001		G1
CP2N	5221	667000001		B7
CP2N	5222	667000001		G5
CP2N	5223	667000002		A4D6
CP2N	52 7	681000001		D4
CP2N	52 9	681000001		G7
CP2N	5210	681000001		G6
CP2N	5211	681000001		G3
CP2N	5117	685000002		A1B7
CP2N	5118	685000001		G4
CP2N	51 5	806000001		G1
CP2N	51 6	806000002		C6A4
CP2N	51 8	806000002		A1F3
CP2N	51 9	806000001		G7
CP2N	5110	806000002		A1G7
CP2N	5112	806000001		G6
CP2N	51 1	926000001		G7
CP2N	51 4	926000002		G7D5
CP2N	51 8	926000002		D7G1
CP2N	51 9	926000002		B2F6
CP2N	51 9	951000001		A1
CP2N	5117	998000002		D4F1
CP2V	51 7	56000001		B1
CP2V	63 6	64000001		G6
CP2V	63 9	64000001		D7
CP2V	6312	64000001		D3
CP2V	6313	64000001		A6
CP2V	5125	598000002		C1E3
CP2V	5126	598000002		G7F5
CP2V	5128	598000003		A7C7E7
CP2V	51 1	667000001		G3
CP2V	51 2	667000001		A2
CP2V	51 3	667000001		A3
CP2V	5214	667000001		G7
CP2V	5224	667000001		G1
CP2V	5225	667000001		B6
CP2V	5226	667000001		G6
CP2V	5229	681000001		C7
CP2V	5230	681000001		G1
CP2V	5231	681000001		A7
CP2V	5115	798000003		A3C5F6
CP2V	5119	798000001		A6
CP2V	51 9	799000002		C6A6
CP2V	5111	799000001		C5
CP2V	5113	799000001		A3
CP2V	5111	912000001		A3
CP2V	5115	912000001		A1
CP2V	5216	921000003		A5B6A7
CP2V	5218	921000002		E3F6
CP2V	5219	921000002		G1F5
CP2V	5221	921000001		B6
CP2V	51 4	923000002		C3C6
CP2V	52 8	992000002		A1E1
CP2V	5110	998000001		G7
CP2V	5112	998000001		D3
CP2V	5219	1007000001		F4
CP2V	51 5	1010000002		F1D2
CP2V	51 6	1010000002		A1D2
CP2V	5110	1012000001		D1
CP2V	5111	1012000001		A7
CP2V	5115	1019000002		D7G4
CP2V	51 7	1020000001		F2
CP2V	51 9	1020000001		E4
CP2V	5111	1020000001		A5
CP2V	51 1	1052000002		E3G1
CP2V	51 2	1052000001		A2
CP2V	5115	1053000001		A2
CP2V	5116	1053000001		D5
CP2V	5117	1053000001		G7
CP2V	5122	1053000002		D1F1
CP2V	51 9	1054000001		G7
CP2V	5111	1054000001		D6
CP2V	5116	1054000001		B7
CP2V	5117	1054000001		G7
CP2V	5122	1054000001		G1
CP2V	5123	1054000001		B2
CP2V	5123	1096000001		B7
CP2V	5127	1096000001		A7
CP2V	51 8	1125000001		E1
CP2V	51 9	1125000001		B3
CP2V	51 9	1133000001		E3
CP2V	5110	1133000001		F7

TABLE IX (Continued)

TRAINING SAMPLE

Vortex or Nonvortex	TIROS Number	Sequence Number	Frame Number	Pass Number	No. of Edit Points	Edit Point Location
CP2N	52	19	5219	1005000002		B5F2
CP2N	52	20	5220	1005000001		E3
CP2N	52	23	5223	1005000001		A3
CP2N	51	3	1010000001			G5
CP2N	51	8	1010000001			C1
CP2N	51	9	1010000001			A4
CP2N	51	5	1011000001			F4
CP2N	51	7	1011000002			A3E5
CP2N	51	25	1011000001			A1
CP2N	51	29	1011000001			A1
CP2N	51	17	1012000001			F7
CP2N	51	31	1012000002			F5B2
CP2N	51	2	1054000001			A3
CP2N	51	8	1054000001			F7
CP2N	51	15	1054000001			A3
CP2N	51	27	1054000001			D1
CP2N	51	28	1054000001			A5
CP2N	51	15	1096000001			F2
CP2N	51	19	1096000001			G5
CP2N	51	4	1134000001			A6
CP2N	51	10	1134000001			A7
CP2N	51	18	1134000002			F4G1
CP2N	51	25	1134000001			B7
CP2N	51	31	1134000002			A3D7
CP2N	51	14	1206000001			E6
CP2N	51	17	1206000001			F1
CP2N	52	10	1206000001			G3
CP2V	52	3	1134000001			G2
CP2V	52	4	1134000001			G7
CP2V	51	2	1137000001			G7
CP2V	51	11	1140000001			F7
CP2V	51	13	1140000001			B1
CP2V	51	2	1151000002			E3F3
CP2V	51	5	1163000001			F5
CP2V	51	7	1163000001			F1
CP2V	51	9	1163000001			A7
CP2V	52	3	1181000001			G5
CP2V	52	5	1181000001			G6
CP2V	51	20	1189000001			F7
CP2V	51	21	1189000001			G7
CP2V	51	22	1189000001			G7
CP2V	52	15	1193000001			G1
CP2V	51	28	1203000001			C7
CP2V	51	29	1203000001			G7
CP2V	51	19	1206000002			A7B5
CP2V	52	13	1224000003			G1G4G7
CP2V	52	17	1224000001			G1
CP2V	51	31	1290000002			G1G4
CP2V	52	32	1290000002			G5G7
CP2V	52	27	1363000003			A1C1E1
CP2V	52	28	1363000001			E1
CP2V	52	29	1363000002			E1G1
CP2V	52	6	1366000001			A3
CP2V	52	23	1493000002			A7C7
CP2V	52	25	1493000001			G7
CP2V	52	7	1522000001			A3
CP2V	52	8	1522000002			A3C5
CP2V	52	9	1522000002			E3G3
CP2V	52	14	1534000001			G3
CP2V	52	15	1534000002			F7G7
CP2V	52	23	1549000001			F7

TABLE X

GENERALIZATION SAMPLE

Vortex or Nonvortex	TIROS Number / Sequence Number / Frame Number	Pass Number / No. of Edit Points	Edit Point Location
CP2N	52 3	17000001	D5
CP2N	5216	28000001	G7
CP2N	52 1	62000001	A6
CP2N	5221	144000001	A1
CP2N	5213	145000002	C3F2
CP2N	5114	160000001	B4
CP2N	51 3	666000001	G1
CP2N	52 6	667000001	G5
CP2N	5211	667000001	D3
CP2N	52 8	681000001	G7
CP2N	51 5	806000001	C5
CP2N	51 3	926000001	C7
CP2N	5112	926000001	C5
CP2N	52 8	993000001	A5
CP2N	5114	998000001	F7
CP2N	5220	1005000001	B1
CP2N	5110	1010000001	G7
CP2N	51 5	1011000001	A2
CP2N	5124	1011000001	G1
CP2N	5120	1012000001	G3
CP2N	51 1	1054000002	E4C2
CP2N	5115	1096000001	C1
CP2N	5128	1134000001	F1

Vortex or Nonvortex	TIROS Number / Sequence Number / Frame Number	Pass Number / No. of Edit Points	Edit Point Location
CP2V	6310	64000001	G6
CP2V	5227	667000001	A1
CP2V	5111	799000001	E6
CP2V	51 2	897000002	F5F7
CP2V	5111	912000001	A1
CP2V	5217	921000001	F7
CP2V	5220	921000001	D1
CP2V	52 9	992000001	A5
CP2V	5114	1011000001	A2
CP2V	5111	1019000001	C5
CP2V	5117	1019000001	A1
CP2V	51 2	1052000001	A1
CP2V	5123	1054000001	A1
CP2V	5126	1096000001	A1
CP2V	51 1	1151000001	A3
CP2V	51 4	1151000001	B7
CP2V	51 5	1163000001	E3
CP2V	5213	1193000001	A5
CP2V	5118	1206000001	B1
CP2V	5228	1363000001	C1
CP2V	52 8	1366000001	A3
CP2V	5214	1534000001	E3
CP2V	5125	1549000002	A1C3

positions are available, for a total of 49 edit points. The adjacent horizontal and vertical edit positions are nine points apart, a figure originating from the packing.

Multiple editing, that is, selecting more than one 180 by 180 portion from a Tiros frame, was used to expand the sample of patterns. Of the 110 vortex frames, 75 were singly edited, 30 were doubly edited, and 5 were edited three times. Of the 103 nonvortex frames, 57 were singly edited, 45 were edited twice, and one was edited three times. When multiple editing was performed, an attempt was made to separate the edit points as much as possible without materially increasing the amount of horizon in the edited picture. In only two cases are two edit points separated by only nine spaces. In all other cases, the separation is more than 12.7 spaces. Tables IX and X also list the edit points for each frame.

Each edit point of each frame gives rise to four of the total of 1,200 sample patterns. The 180 by 180 sample pattern obtained by editing is rotated by 90, 180, and 240 degrees to give the additional three sample patterns.

The networks under consideration in this study have no invariance mechanisms for rotation or translation. Therefore, a pattern which is a rotation or a translation of another sample pattern may be considered to be an additional sample, provided that the rigid motion is sufficiently large compared to the period of significant spatial frequencies. The optical resolution studies (Section 6.5.1) indicate that distances of nine or more spaces are indeed significant with respect to these periods.

6.3 Computer Programs

A total of eight computer programs were used in the studies on cloud patterns. One of these, the Cloud Pattern Computer Design Program, was compiled in eight versions. Six of these versions represent parameter variations (all parameters were built into the program) and the

other two represent program changes to restrict the input fields of the logic units. The eight 7094 programs, and a brief functional description of each are given below.

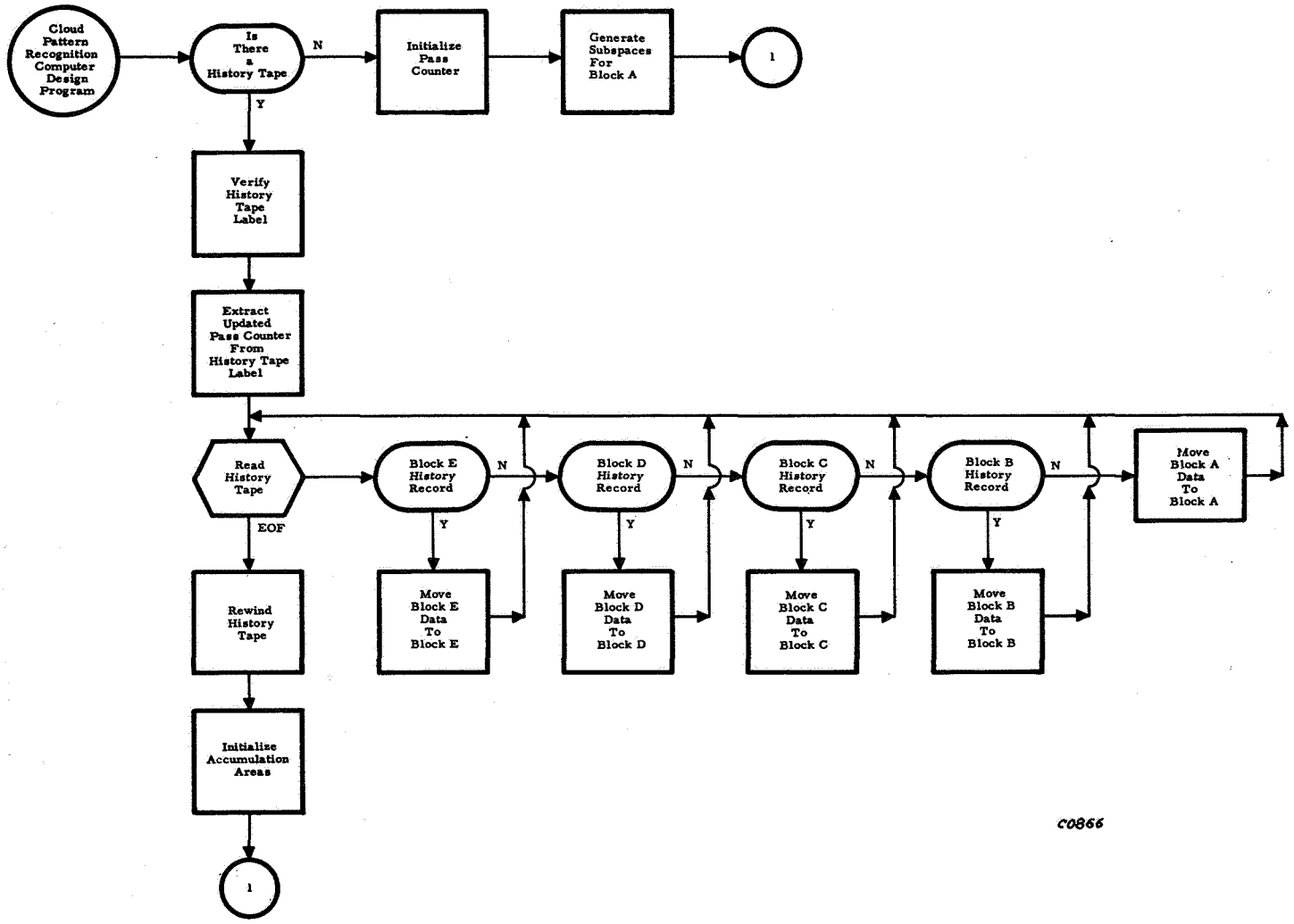
1. CPO1 - Convert and Repack Tiros Data---This program accepts as input each of the digitized Tiros frames as supplied by the Institute for Space Studies. For each frame, the program tests portions of the header to insure that the proper frame has been inputted, and verifies that the proper number of data records are present. If the frame is satisfactory the header is put into a standard format, and the gray scale reduced from 6 to 4 bits, as described in Section 6.2.3. The pattern is then repacked to nine points per machine word, and stored on a pattern tape. A punch card is produced for each pattern on the pattern tape. To produce the 505 pattern output tape, 13.7 minutes of machine time are required.

2. CPO3 - Cloud Pattern Edit and Rotation---This program produces two files of patterns on magnetic tape, a vortex tape and a nonvortex tape. For each pattern to be included in the sample, the card produced by CPO1 is punched to indicate whether the pattern is a vortex or a nonvortex pattern. The number of edit points and the location of these edit points are also punched into these cards. The program accepts the whole set of these cards and stores them. When the pattern tape contains more than one pattern with the same header (a frequent occurrence), an additional card must be supplied to indicate which of the duplicate patterns is desired. The program accepts patterns from the pattern tape and checks the header information against the table of desired patterns. If a match is not obtained, a new pattern is read in; if a match is obtained, the pattern is edited as described in Section 6.2.3, according to the first edit position. The resulting 180 by 180 pattern is written onto the appropriate output tape. The pattern is then rotated and repacked for rotations of 90, 180, and 240 degrees, and the three patterns thus obtained are written onto the same tape. The process is repeated for all additional edit locations. For the 505 patterns on the input tape

and 500 patterns to be written on each of the output tapes, the running time for this program is 16 minutes.

3. CPO5 - Cloud Pattern Computer Design---This is the major program of the study. The design algorithm used is discussed in Section 6.2.1. A flow chart of this program is given in Figure 55. In general, the operation of the program is as follows: Input connections are generated for the logic units in Block A. A pattern is input from the vortex file. The activities of the logic units in Blocks C, D, and E are determined, and the pattern's loss modified accordingly. For each unit in Blocks B through D, the pattern loss is added to the active vortex or nonactive vortex total for that unit. For each unit in Block A, the contribution of the pattern to the estimate of the vortex covariance matrix for that unit is determined. This is repeated for all of the patterns in the vortex file. The vortex covariance matrices for Block A are stored on a scratch tape. The nonvortex file is then processed in an equivalent manner. When all of the sample patterns have been processed, the logic units in Block E are stored in a permanent file, corrections for the output weights of logic units in Blocks D and C are computed, and these units transferred to Blocks E and D respectively. The units in Block B are evaluated, and the best 10 are transferred to Block C together with an output weight for each unit. The covariance matrices for each of the units in Block A are processed to produce weights for the input connections of that unit. The units are then transferred to Block B. The writing of a history tape completes the pass. The history tape permits the program to be interrupted after completion of any pass. Steps are taken to terminate the design process when there are no errors on the sample patterns. Three additional passes are required to move the units in Blocks C, D, and E into the permanent file. The running time of a single pass is 13.7 minutes and 5.3 minutes for a terminal pass. The design of an entire network takes about six hours of machine time.

4. CPO7 - S-C4020 (240 by 240) Cloud Pattern Plot (using six bit intensity points)---This program was supplied by NASA, and was



(Continued)

Figure 55. Program Flow Chart

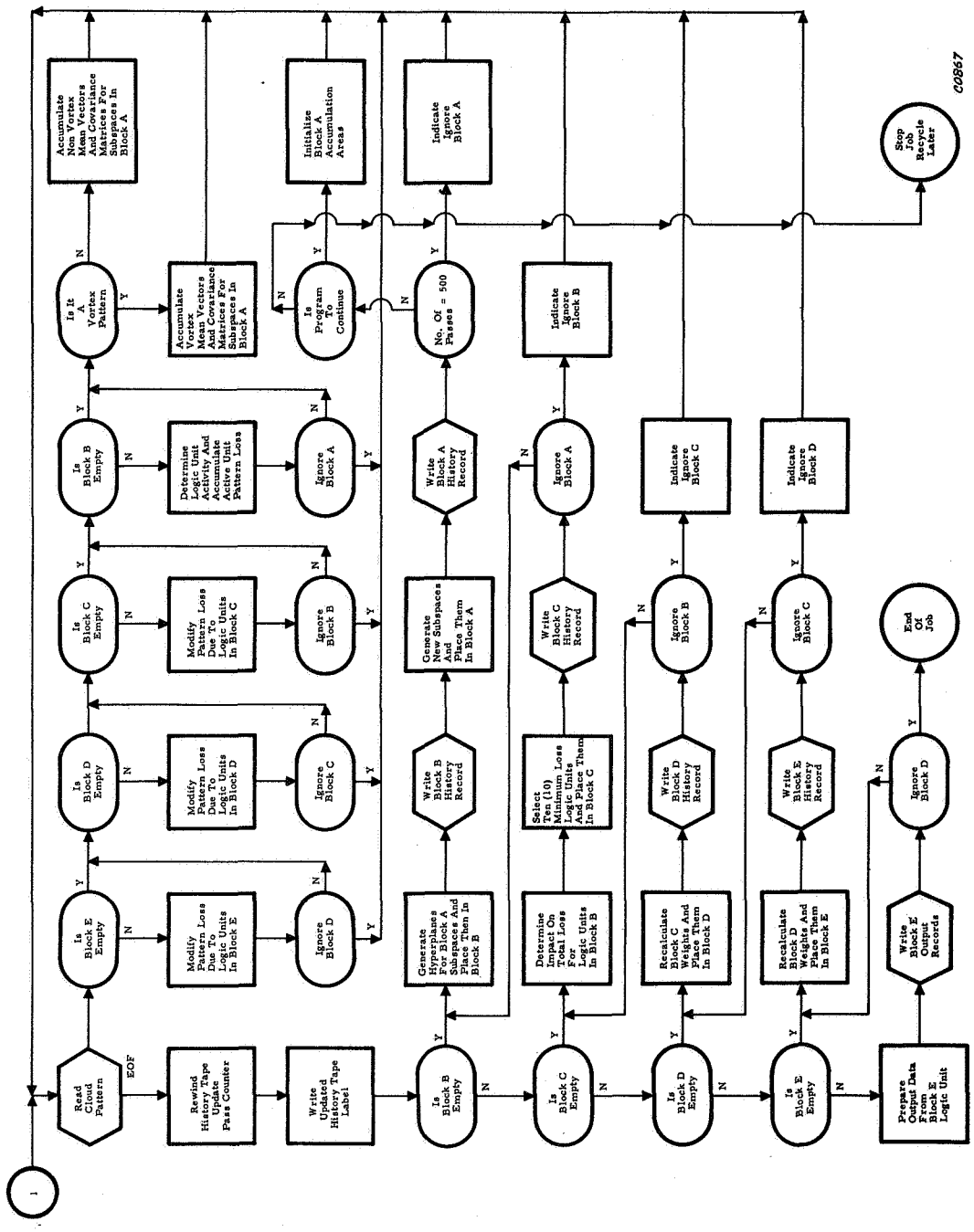


Figure 55 (Continued). Program Flow Chart

modified to be compatible with the Douglas monitor. It accepts the patterns as supplied by the Institute for Space Studies, and produces a tape enabling an S-C4020 printout of the TIROS frame. Sixteen gray levels are used in the printout. The running time of this program is approximately one (1) minute per frame.

5. CPO9 - S-C4020 (240 by 240) Cloud Pattern Plot (using four-bit intensity points)---This program is a modification of CPO7 to accept the pattern tape produced by CPO1. The running time is ~1 minute per frame.

6. CP11 - S-C4020 (180 by 180) Cloud Pattern Plot (using four-bit intensity points)---This program is a modification of CPO9 to accept the pattern tapes produced by CPO3. The running time is also ~1 minute per frame.

7. G205X - Tape Consolidation---This program accepts all of the permanent file tapes produced in the course of designing a network, and generates a single tape from them. Multiple tapes are produced when the design of a network is interrupted. This program can also insert missing data obtained from punched cards, when necessary. The running time is six minutes.

8. G205 - Generalization Program---This program tests the generality of a network design. The network design is stored, and the patterns of the design deck are accepted one at a time. The activity of each logic unit is determined and stored. When all patterns are processed, the program computes the performance of partial network designs in increments of ten units. For each partial network, a threshold for the decision element which gives minimum classification errors is determined. The network design is printed out, as is the number of errors and the discriminant value (input to the response unit) for each partial network. The generalization sample of patterns is then inputted and processed in a like manner. The thresholds determined on the design sample are used to count errors on the independent sample. The

program permits the discriminant values determined from one network to be used as starting values for testing another network. This is the equivalent of combining more than one network design into a single recognition machine. The running time for this program is ~10 minutes per network.

6.4 Results

A total of seven complete networks and four partial recognition networks were designed. The complete networks, and in some cases, combinations of networks, were tested against the 1000 patterns used in their design and against an independent sample of 200 patterns. In these tests best performances were determined for partial networks — the performance being optimized with respect to the decision element threshold.

6.4.1 Rate of Design

In the first successful series of network designs, three partial networks were designed in which a parameter β was varied. β controls the rate at which the networks are designed, and is discussed in Section 6.2.1. Runs obtained during the debugging phase had indicated that when β was as small as .007, instability could occur in the design, while a value for β of 700 (the value was this high through a program error) lead to virtually no changes in the pattern losses. The values for β which were used for the three networks were .07, .2, and .7. Seven passes through the program were accomplished for each case. In these seven passes 30 logic units were placed in the permanent file, and an additional 20 units had affected the pattern losses, but had not completed the full cycle of weight adjustments.

Figure 56 shows the total system losses and the numbers of classification errors as a function of the pass number for the three designs. The numbers of logic units affecting the pattern losses is ten times the pass number minus twenty. The error count is relative to a decision element threshold giving minimum system loss. The value of .07 for β was selected, since it provided the fastest rate of design

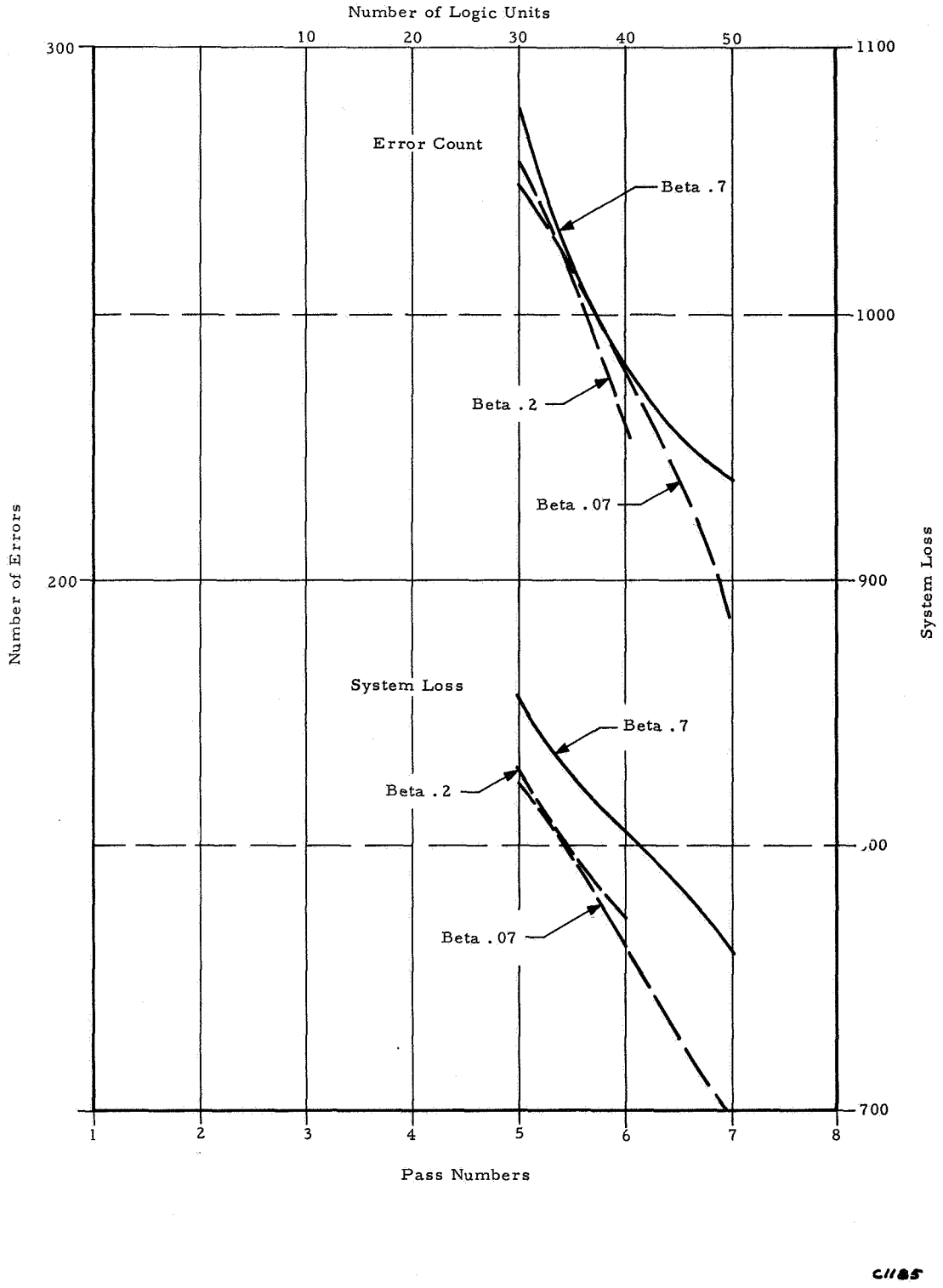


Figure 56. Rate of Design (Beta)

and since no sign of instability was evidenced. With this choice of β, instability was not a problem in subsequent studies.

6.4.2 Rate of Activity

For the type of logic unit selected (the gray invariant version of the parallel hyperplane unit, see Section 4.5), the half-angle of the cone determines the rate of activity of the logic unit. The studies on alphabetic characters indicate that the design rate is not too sensitive to this parameter, and that an activity rate of one-half for the negative class gives good results. The cone angle giving this rate depends on the number of input connections to a logic unit. The parameter appears in the program as COSPHI, the cosine of the half angle. A value of .248 gives a theoretical activity rate of one-half for the nonvortex patterns.

The partial networks of the preceding section used a value of .248 for COSPHI. Two additional seven-pass designs were accomplished, corresponding to values of .174 and .342 for COSPHI. The system losses and numbers of classification errors for the three partial networks with β = .07 are shown in Figure 57 as a function of the pass number. As before, the number of logic units affecting the pattern losses is equal to ten times the pass number minus twenty. Based on these data, the value of .248 was used for COSPHI in the remaining network designs.

6.4.3 Separability of the Patterns

In an earlier study,[1] results indicated that an "open" loop design technique, such as a "forced learning" perceptron, would not provide recognition networks capable of a high level of performance on the design sample of patterns. It is a matter of some interest, then, whether a tightly "closed" technique, such as the current one, could provide perfect or nearly perfect performance on the design sample.

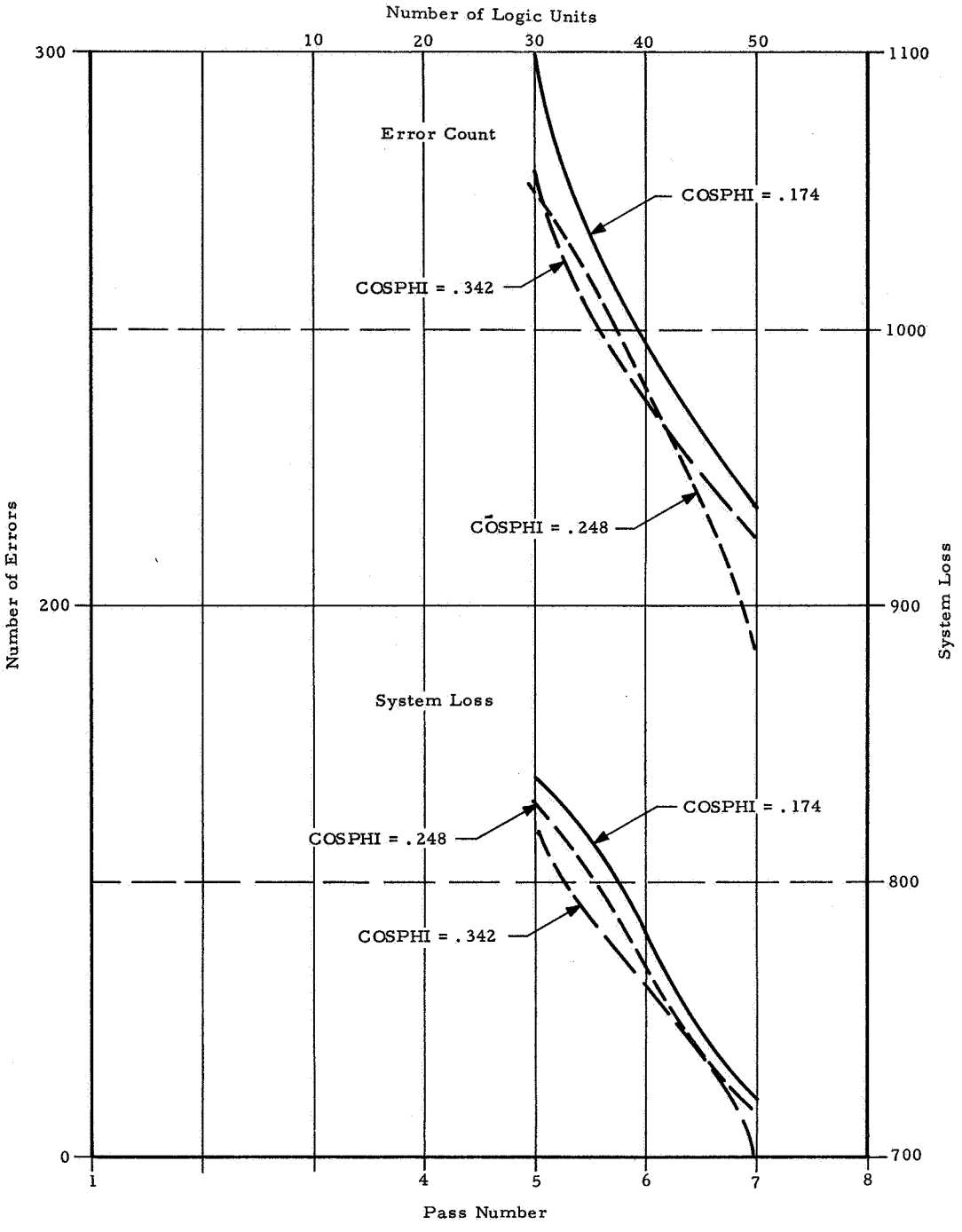


Figure 57. Rate of Activity (COSPHI)

The convergence proof of Section 4.6, and the error correction theorem for perceptrons[21] show that means are available for adjusting the weights of the inputs to the decision element so as to give perfect performance on the sample patterns, provided that the property filters were such as to permit linear separability of the property vectors. The problem of designing for high performance on the design patterns becomes one of providing a set of property filters which permits linear separability. The perceptron approach of choosing random property filters is likely to require enormous numbers of units to obtain separability. The current technique provides a means for designing logic units specifically to solve the remaining portions of the sample problem, once the major portion of the sample patterns have been correctly classified using a set of statistically derived property filters. The results of the studies using the sample cloud patterns and the results of the studies discussed in Section 5 indicate that this current technique is capable of providing a small set of property filters permitting linear separability, even in complex problems.

Seven complete networks were designed, each providing perfect performance on the sample patterns. The number of units in these networks ranged from 220 to 290. (It is to be noted that perfect performance is actually achieved when the number of logic units affecting the pattern losses is 10 less than these numbers.) Figures 58 through 64 present the system losses and classification errors for the partial networks of these seven systems. The uniformity of the rate of design for these seven machines is remarkable. In several cases, the number of classification errors increases after reaching zero, due to subsequent adjustments of the input weights of the decision element. To obtain perfect performance with these networks, further iterations of these weights may be performed, or the design process can be terminated immediately after zero errors are achieved.

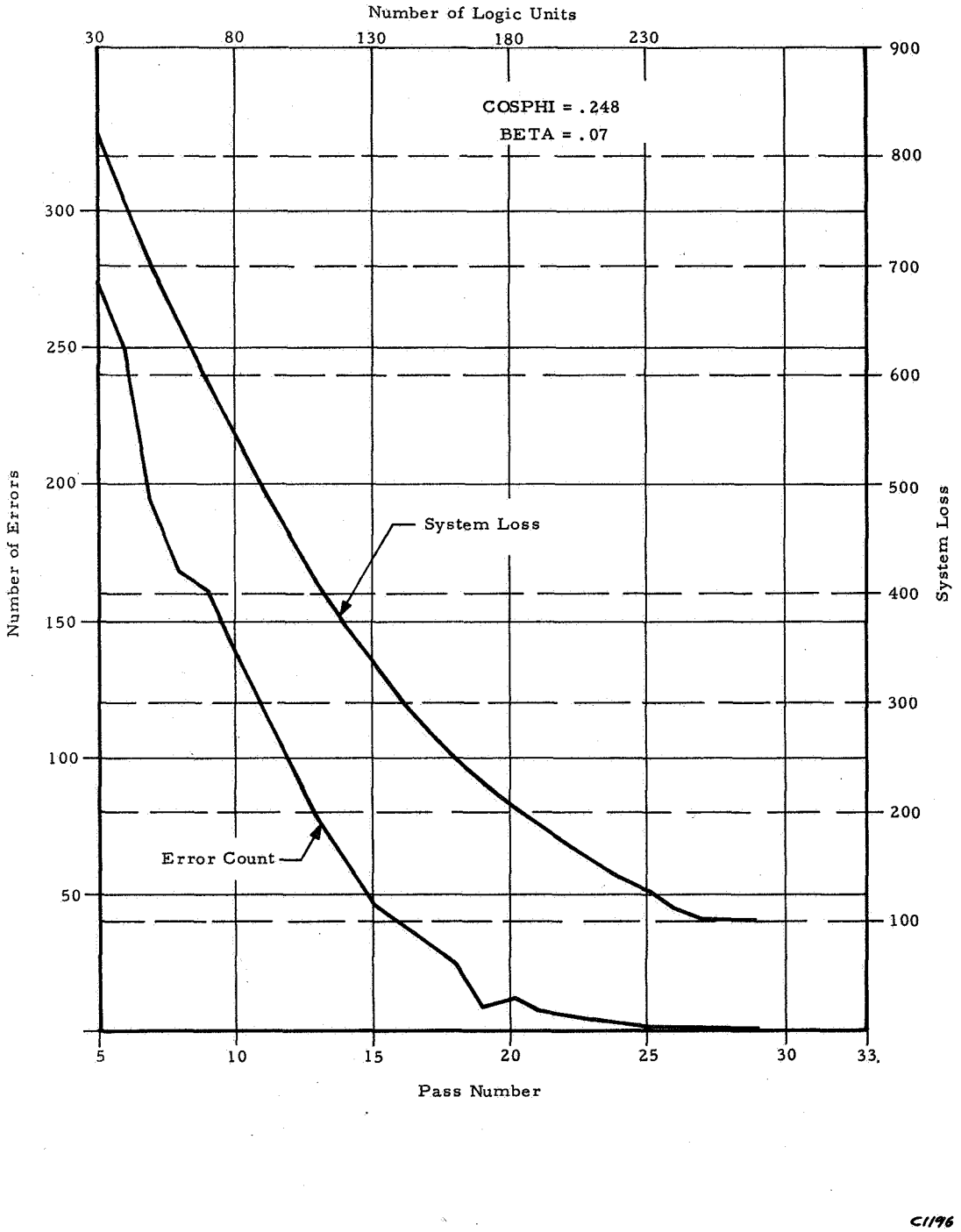


Figure 58. Machine Design #1

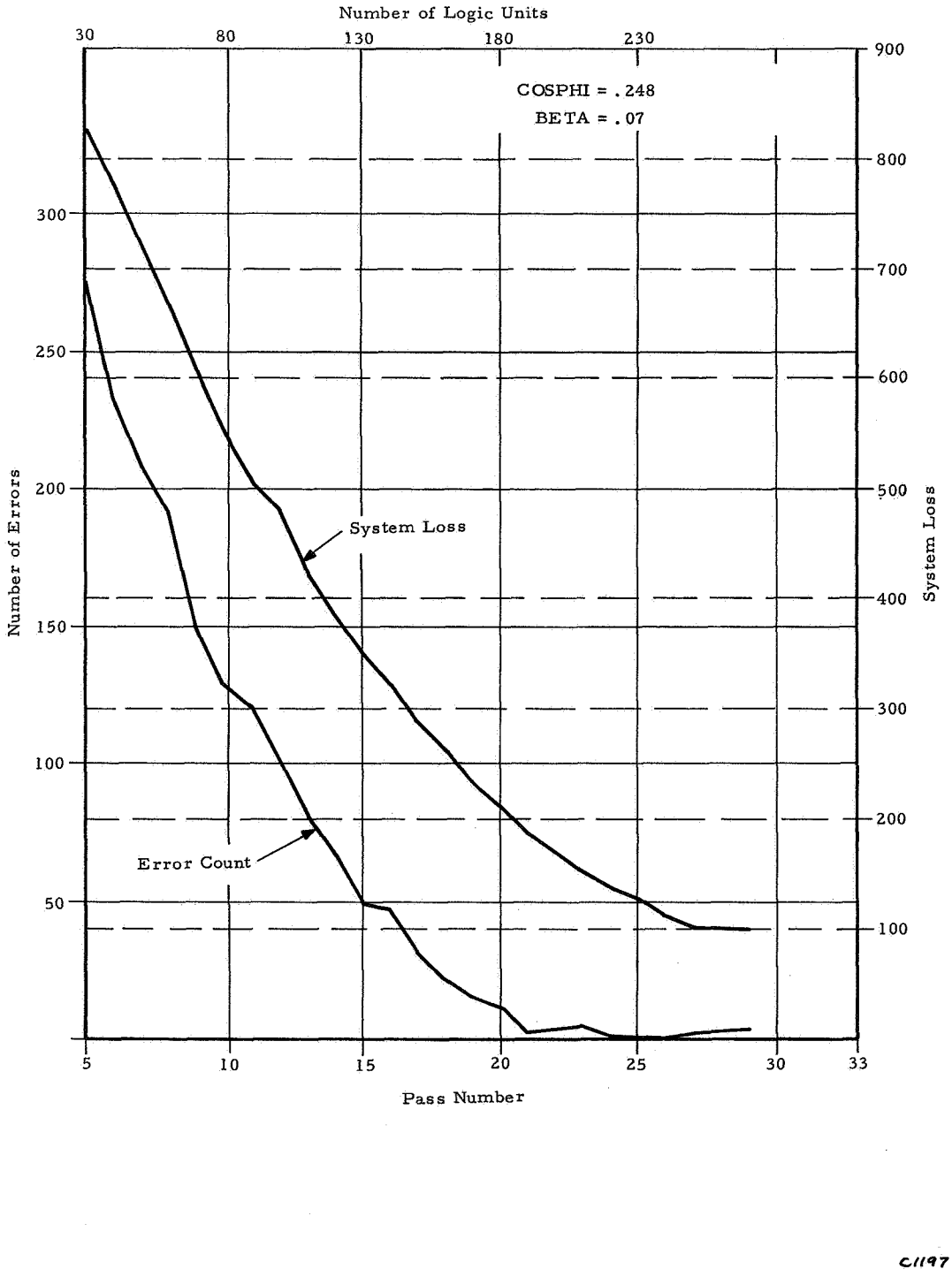


Figure 59. Machine Design #2

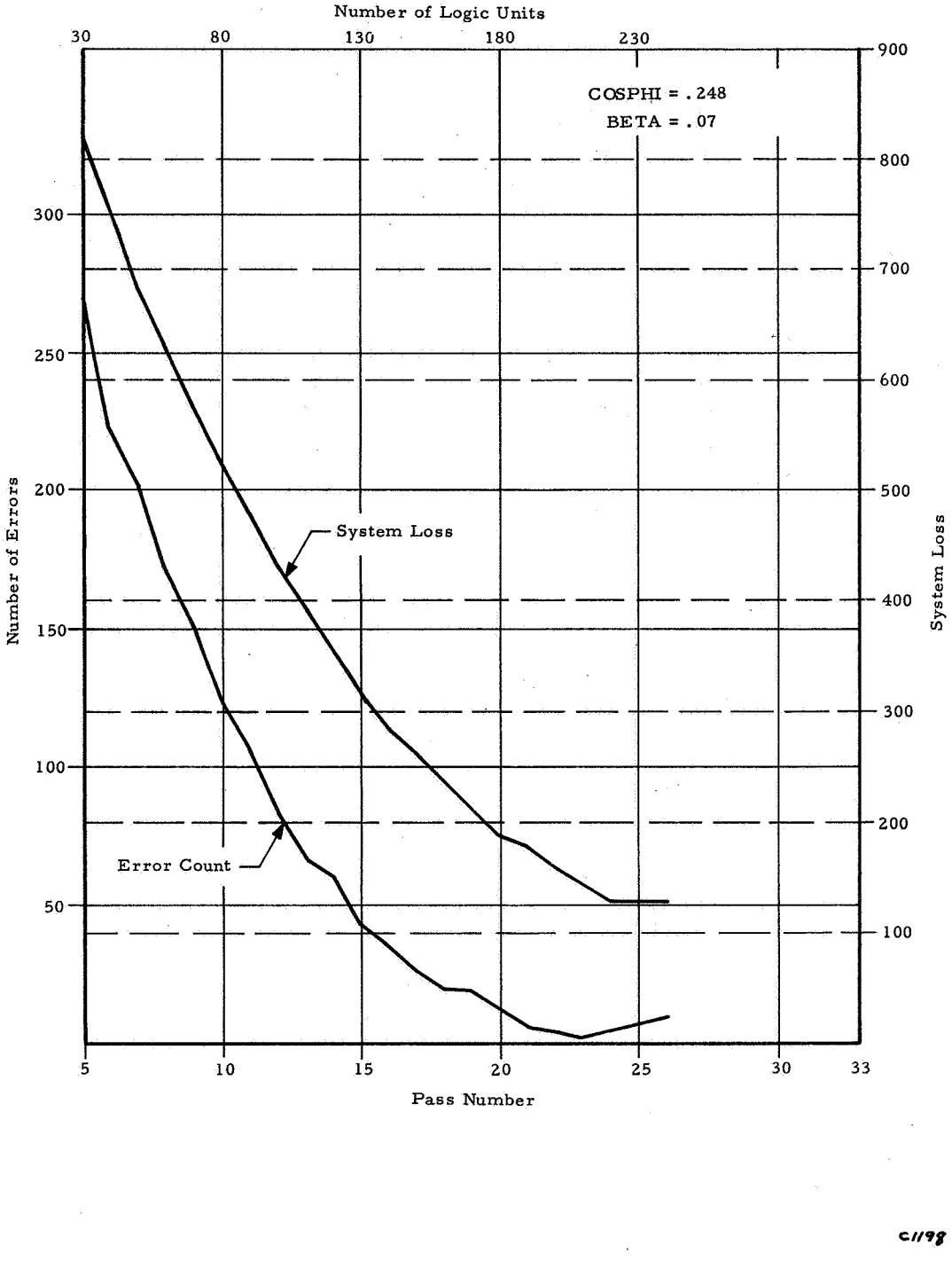


Figure 60. Machine Design #3

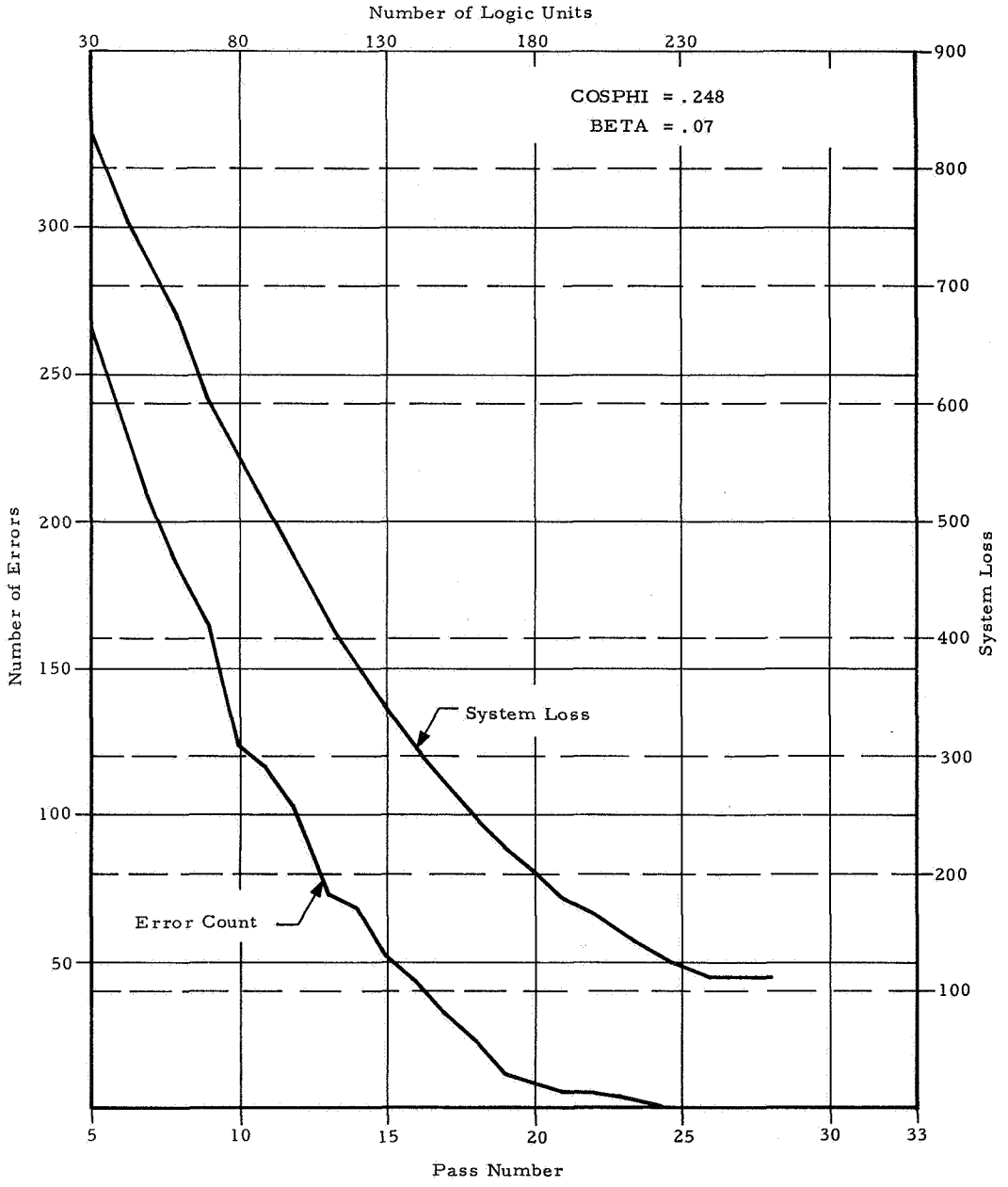


Figure 61. Machine Design #4 — Reduced Resolution Patterns (90 x 90)

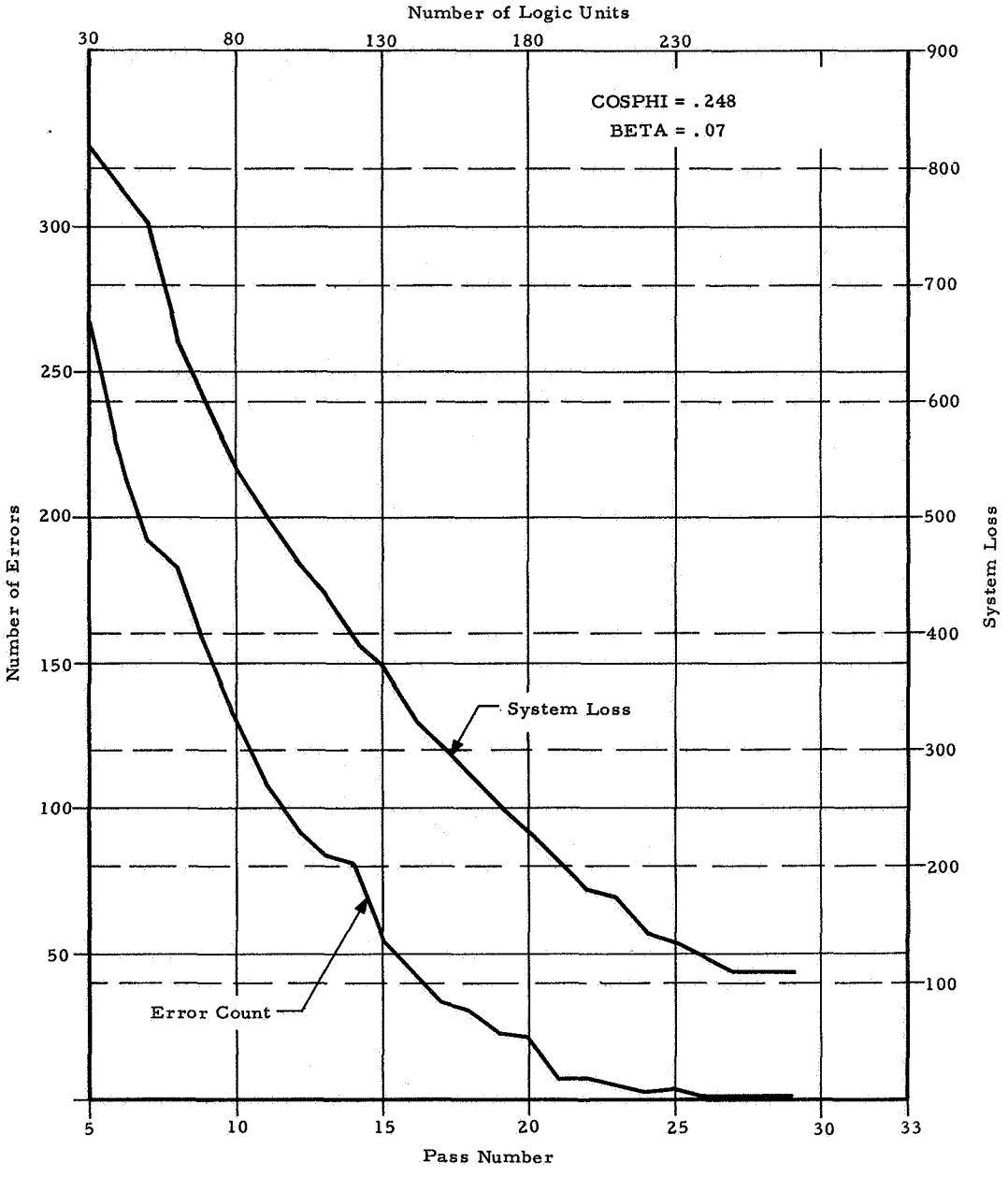


Figure 62. Machine Design #5 – Reduced Resolution Patterns (60 x 60)

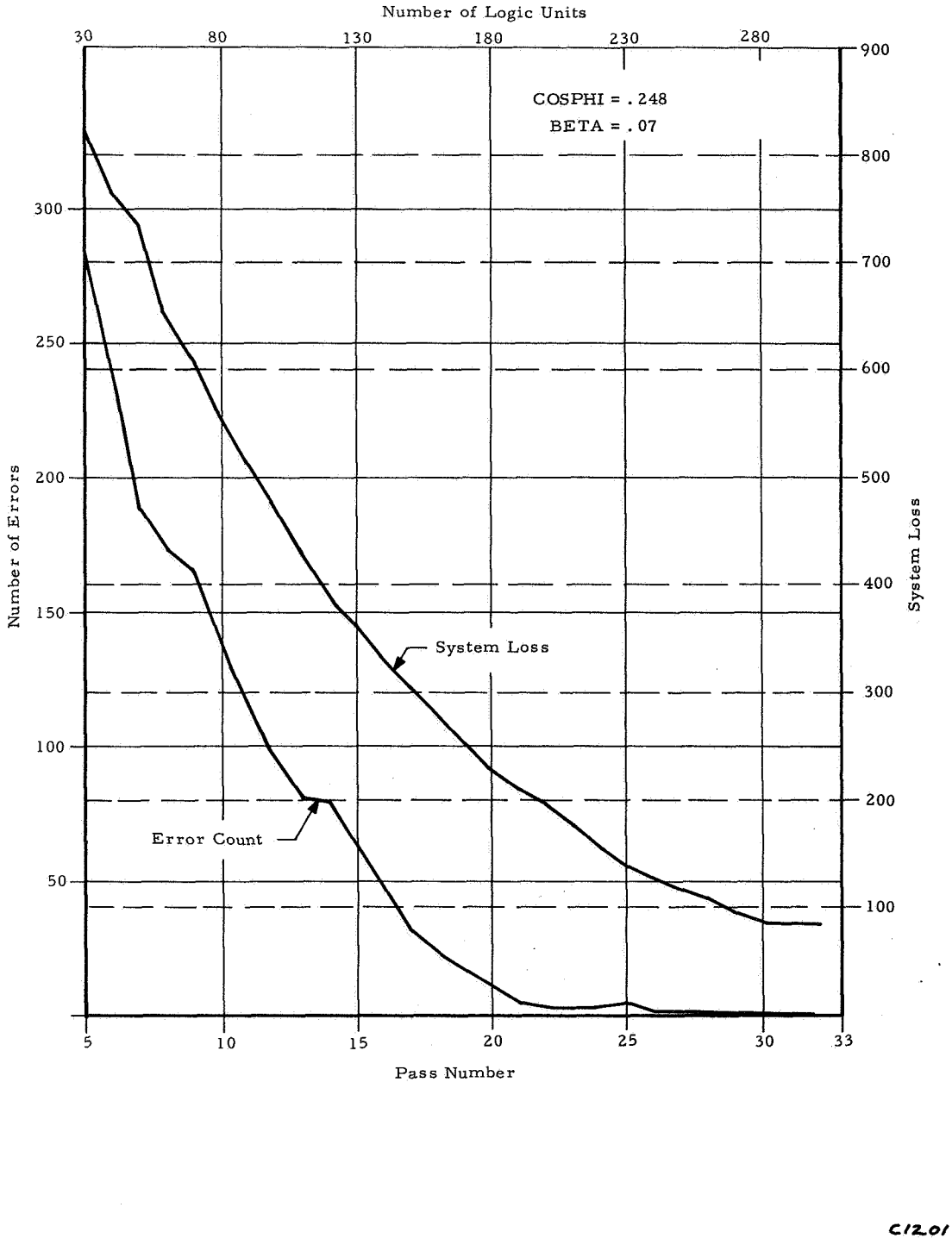


Figure 63. Machine Design #6 — Logic Unit Input Field Limited (90 x 90)

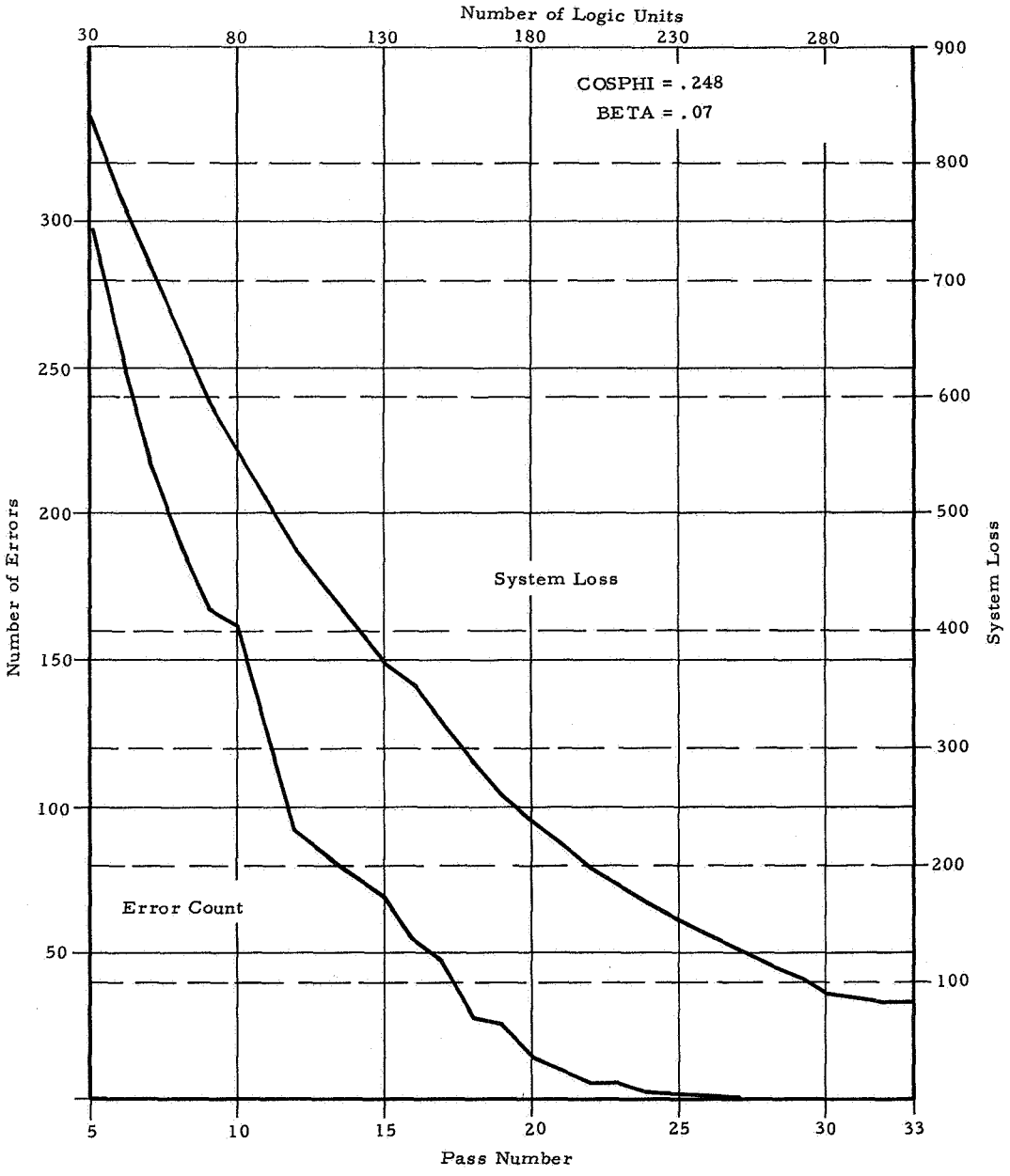


Figure 64. Machine Design #7 — Logic Unit Input Field Limited (45 x 45)

6.4.4 Generality of Design

The attainment of a high level of performance on the design sample of patterns is of great importance to the study of the application of a design technique. The results of the preceding section show that for this problem, perfect performance may be obtained on the design sample.

Of equal importance, however, is the attainment of a high level of performance on an independent sample of patterns. Two factors may lead to different "generalization" performance than "recognition" performance. One of these factors, the representativeness of the design sample (and for that matter, the independent sample), or the accuracy of the distributional information available in the design sample, is of lesser interest in this study. The second factor is how well the design technique utilizies the distributional information available in the design sample, and is of great importance in this study. In general, however, it is not possible to separate the effects of these two factors in an experiment which uses only one design technique, such as this one. Section 5 is perhaps more indicative in this respect. An example may serve to illustrate these factors.

First, we present a design technique which is guaranteed to provide perfect performance on the design sample, but in general will give very poor results on an independent sample. Let P_i be the vector (32,400-dimensional) representing the i-th sample pattern. Let Z be a vector which is <u>not</u> orthogonal to any vector $P_i - P_j$. Since there are only a finite number of sample patterns, such a vector Z can always be found. The components of Z will serve as weights for the input connections for all of the property filters. The property filters will differ only in their thresholds. The property filters thus partition the space with a set of parallel hyperplanes. The thresholds are selected in such a way that no cell contains both vortex and nonvortex patterns. Since Z is not orthogonal to any $P_i - P_j$, this is possible, and the maximum

number of property filters which may be required to separate the 1000 sample patterns is 999. The specification of a linear discriminant for these property filters is trivial. Figure 65 illustrates the separation of the space and the reason that generalization may be quite poor. In the illustration a more appropriate choice of Z might lead to only one property filter for perfect results on the sample patterns and better generalization results.

The example above shows that perfect performance on the sample patterns is not in itself an adequate design criterion. Even if the design sample of patterns is very representative of the actual distribution of patterns, the technique above is capable of providing poor designs.

In the example, an additional design criterion of minimum network complexity would give good results. Such a criterion, however, places great demands on the accuracy of the design sample. In a problem as complex as the vortex recognition one, it is unlikely that any individual property filter can solve a very significant portion of the problem. If the design sample is not sufficiently representative, then a high premium on minimum complexity (such as in the present design technique) may result in a network based on sample anomalies. For example, if a large percentage of the sample vortex frames contain a horizon, and the same were not true of the sample nonvortex frames, then the presence of a horizon in the frame will be considered an important property of vortex frames. The restriction on network complexity reduces redundancy in classifying the sample patterns, redundancy which may include the properties which are of real significance to the generalization problem.

The generalization results are shown in Figures 66 through 72. Partial network designs are considered — that is, data are plotted for the first ten logic units considered as a network, the first twenty units considered as a network, and so on. Since these data are not available from the machine design run, the 1000 design patterns are processed to determine thresholds for the decision element which gives

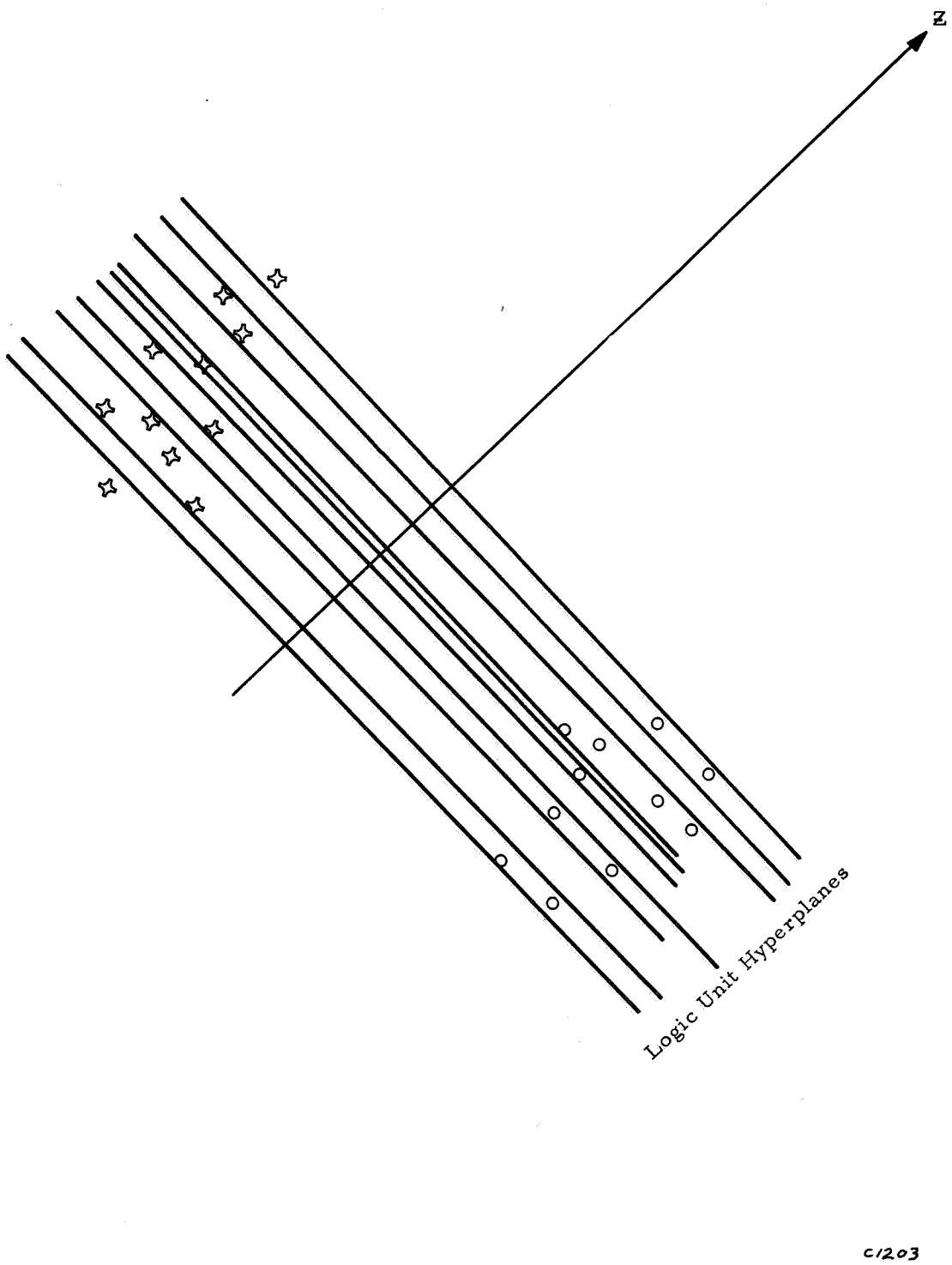


Figure 65. A Special Design Technique

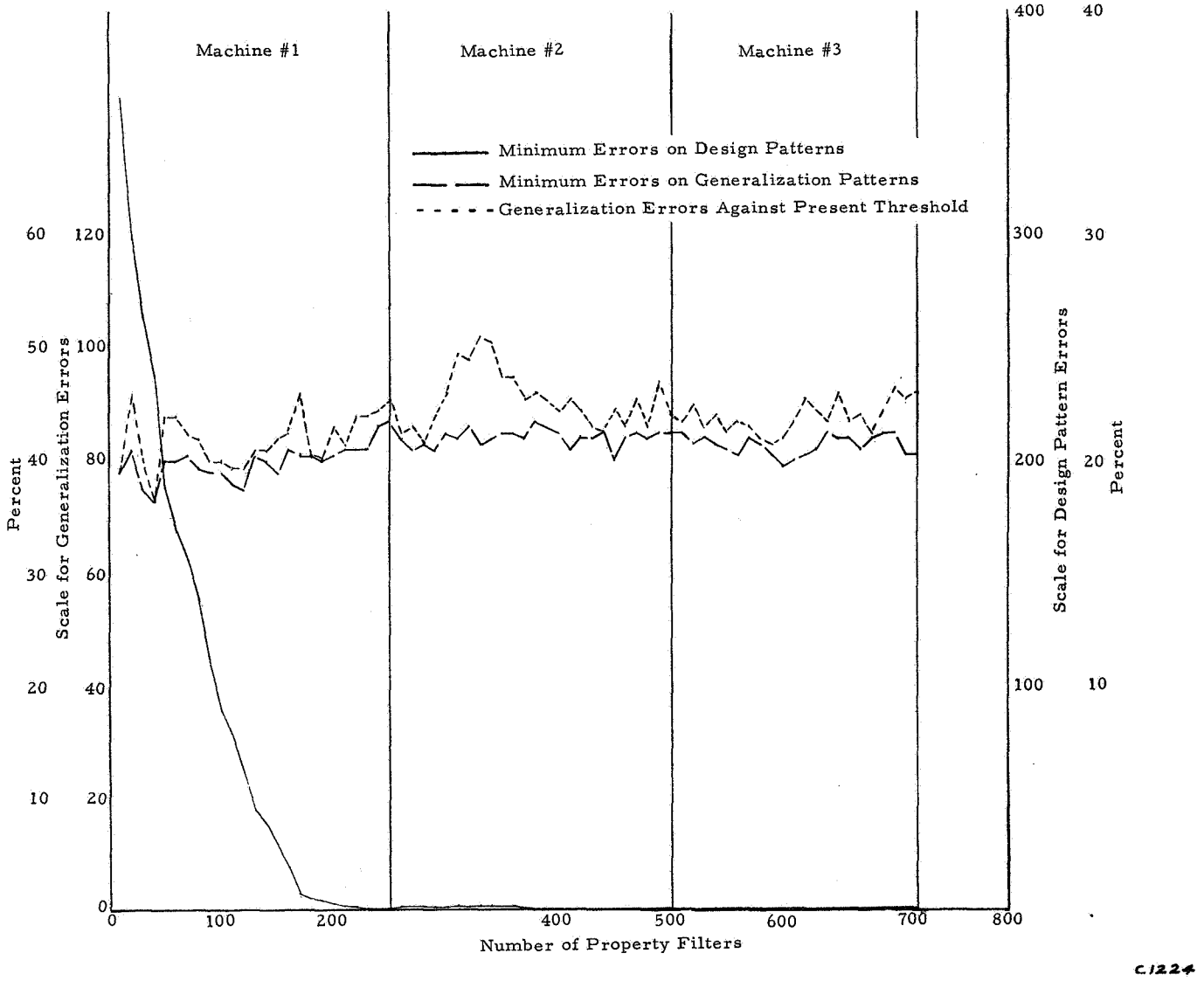


Figure 66. Network Performance — Machine #1 Plus #2 Plus #3

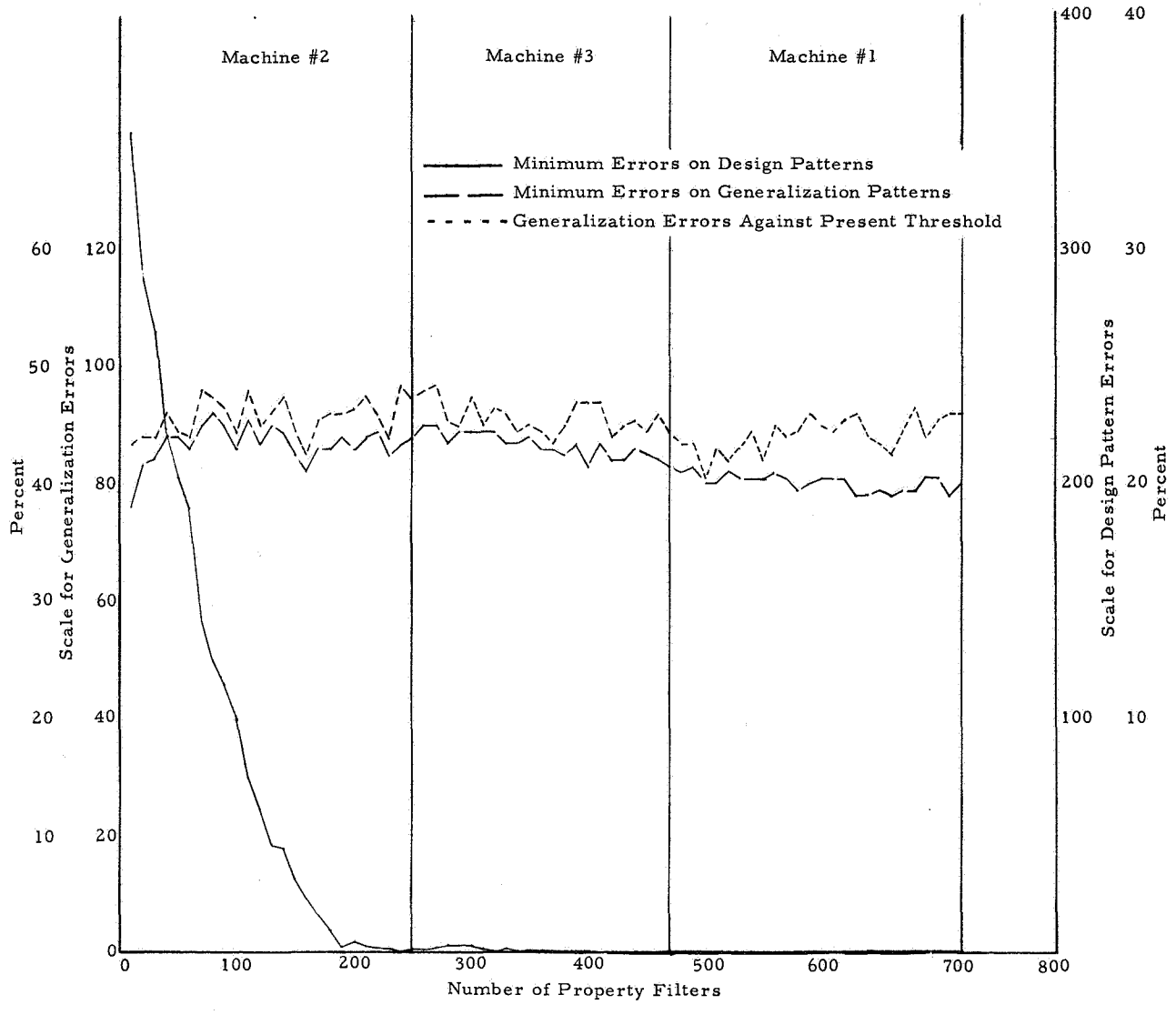


Figure 67. Network Performance — Machine #2 Plus #3 Plus #1

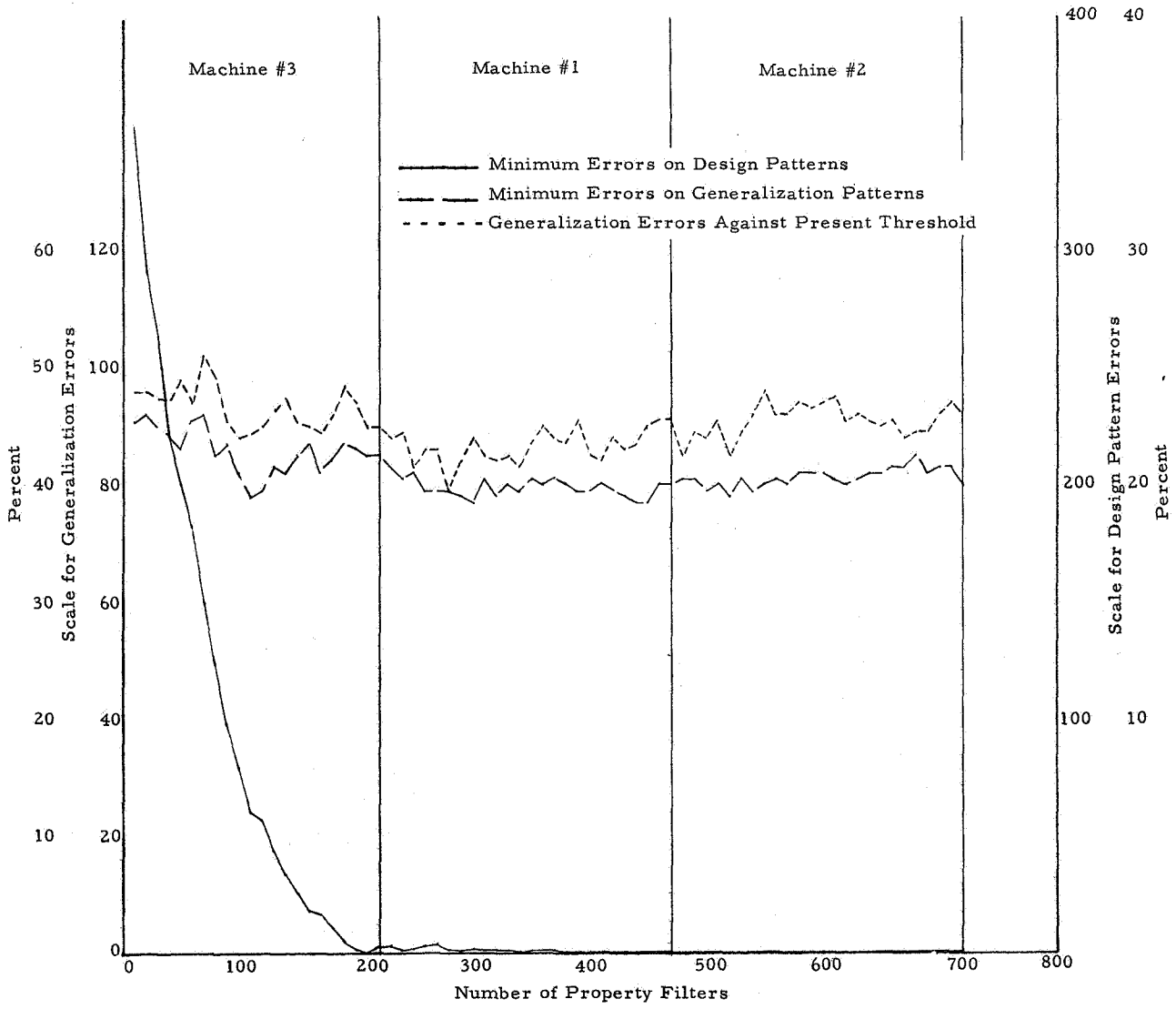


Figure 68. Network Performance — Machine #3 Plus #1 Plus #2

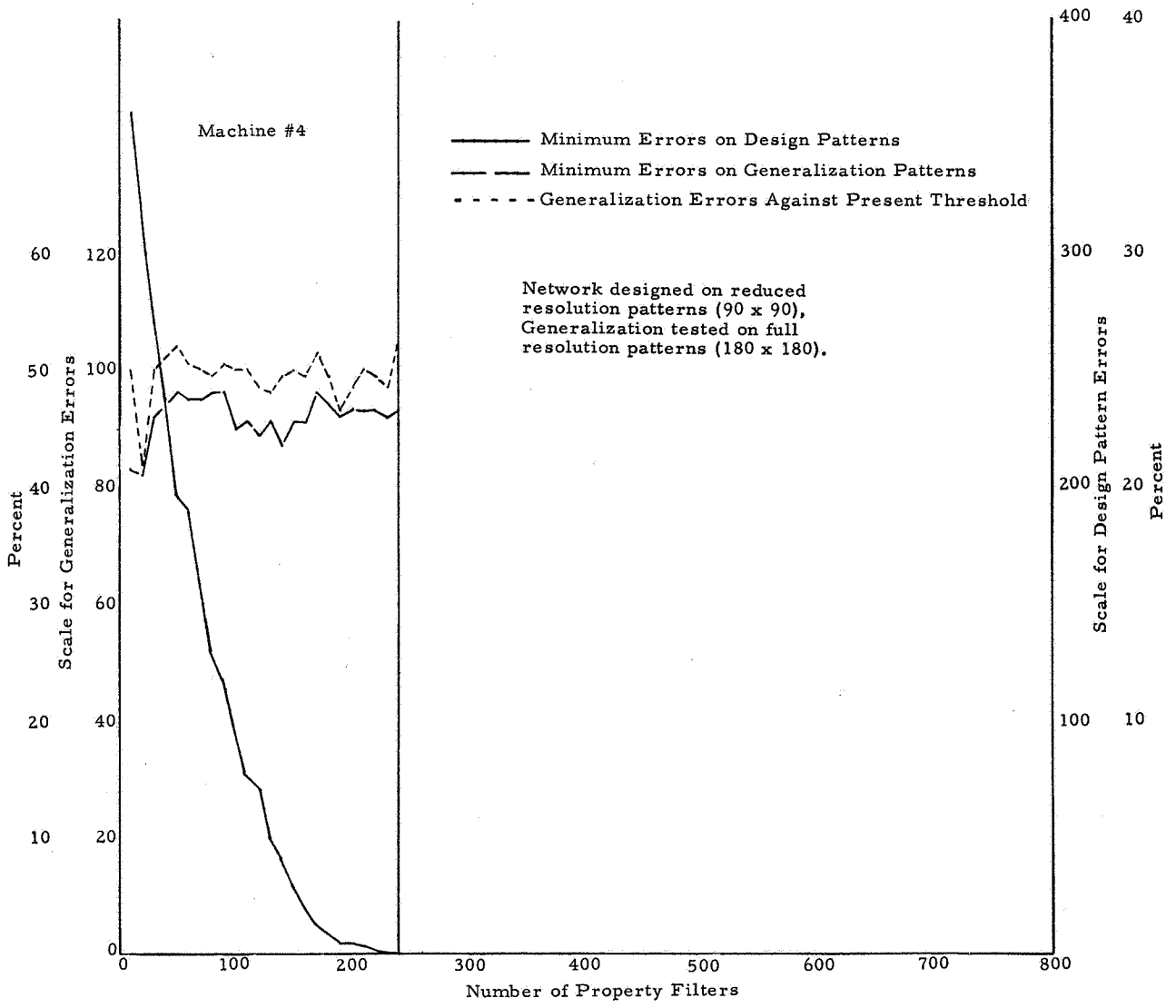


Figure 69. Network Performance — Machine #4

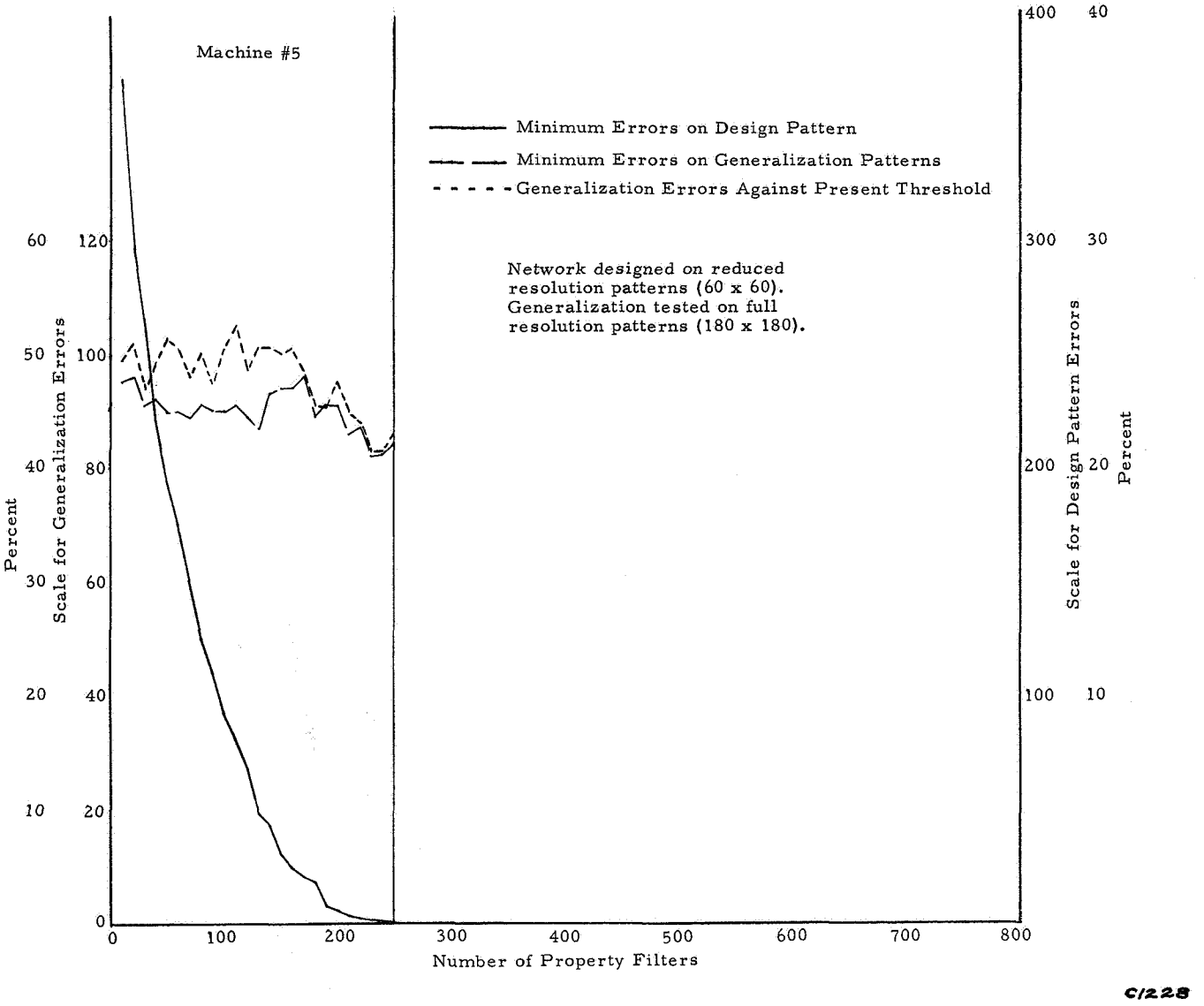


Figure 70. Network Performance — Machine #5

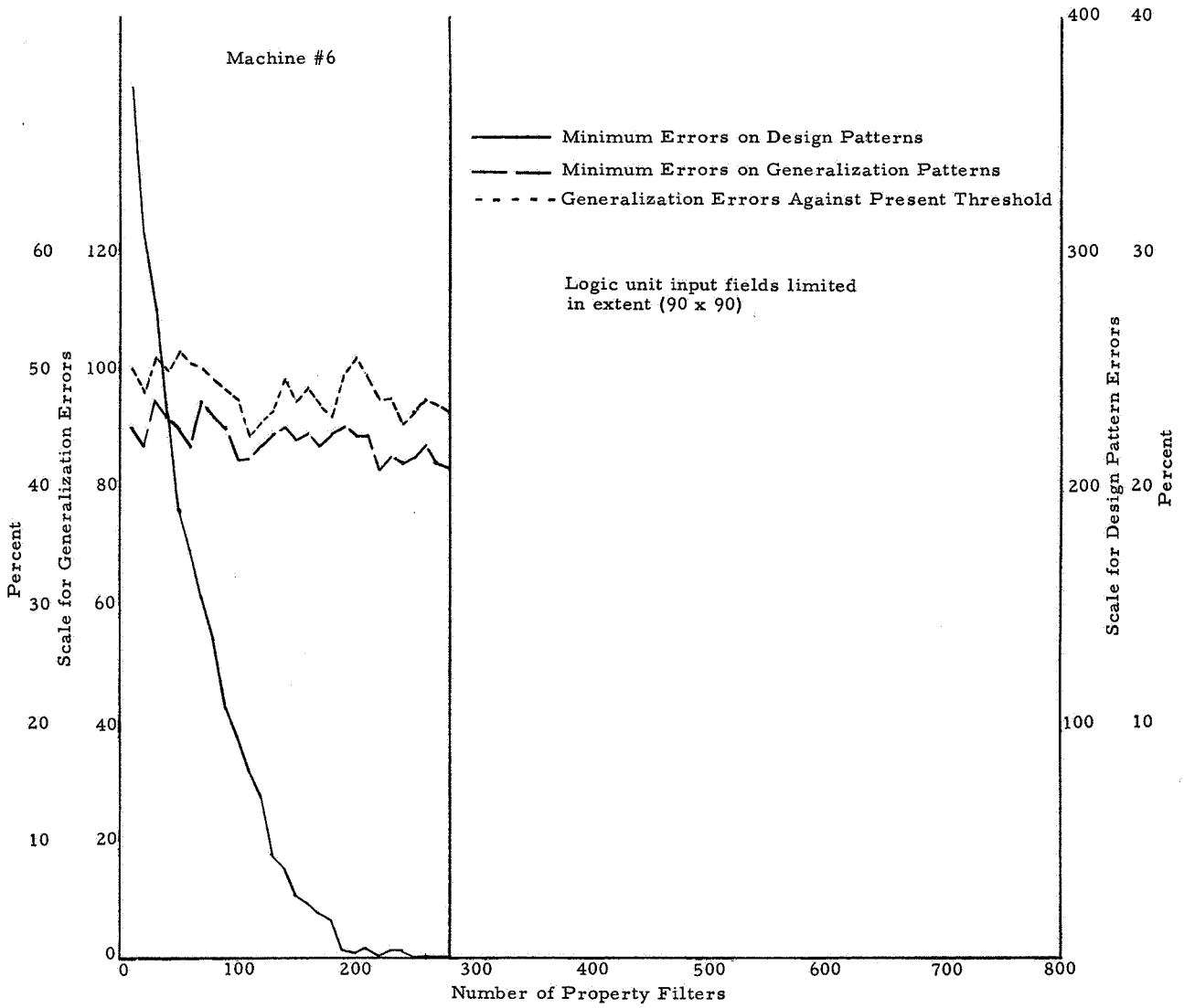


Figure 71. Network Performance — Machine #6

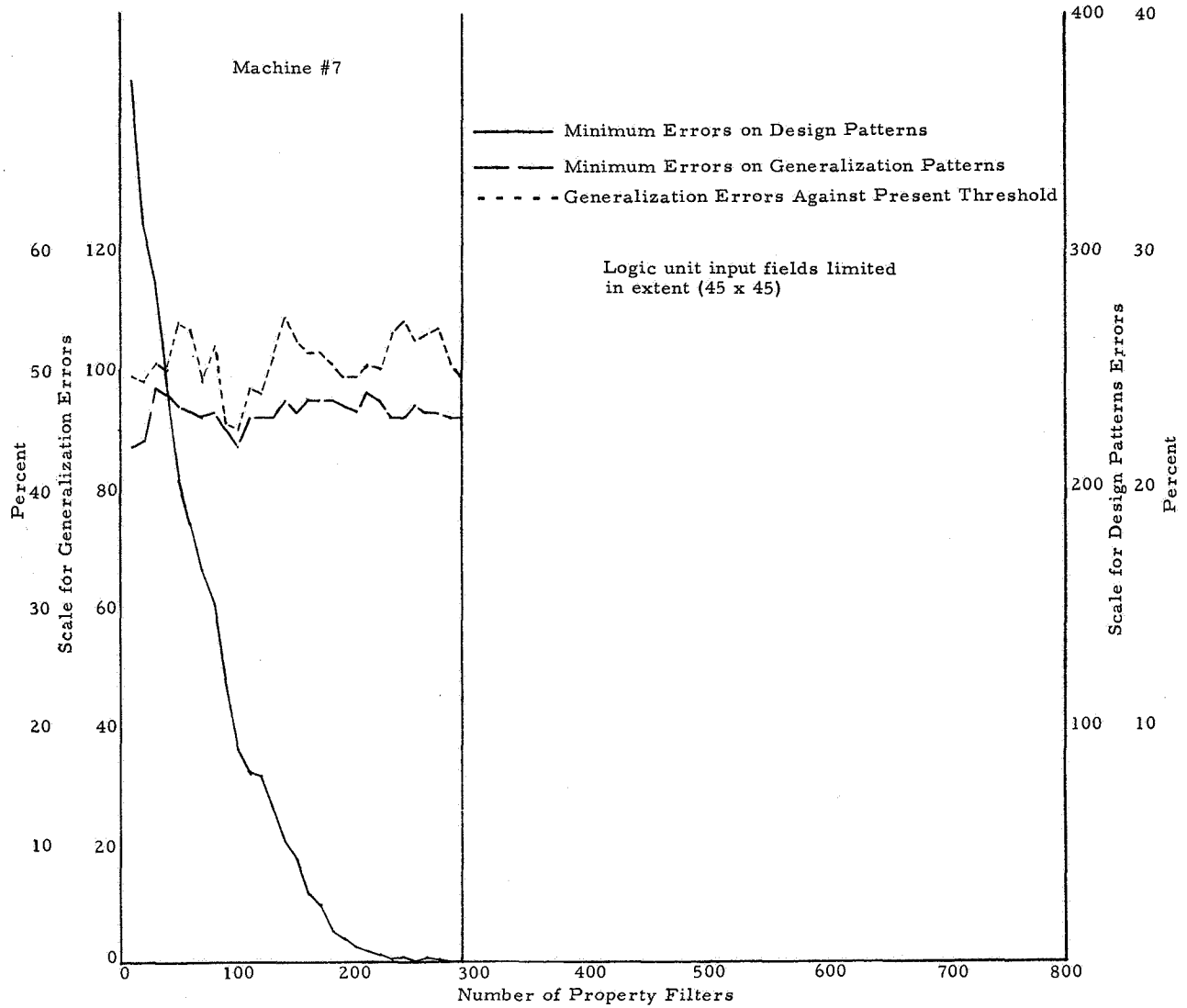


Figure 72. Network Performance — Machine #7

the least number of incorrect decisions on the design patterns. The thresholds for the partial networks are recorded for use on the generalization patterns. The solid curves in Figures 66 through 72 present the minimum numbers of errors for the partial networks. These curves are in slight disagreement with the data in Figures 58 through 64 for two reasons. First, the errors are counted against different thresholds. The earlier curves used a threshold to give a minimum value for the loss function, the later curves to give minimum error count. The second difference is in the number of property filters affecting the discriminant. In the earlier curves, the first data are produced in Pass 5 when the first block of ten property filters are placed in the permanent file. At that point there are 20 additional units which have affected the discriminant values, although these property filters have not been assigned their permanent weights.

A vertical line indicates a complete network. The curves in Figures 66, 67, and 68 continue beyond the designs of Networks 1, 2, and 3, due to a compounding of networks discussed in the following section.

The performance of the partial networks was then tested against the generalization patterns, using the thresholds derived above. These data appear as the short-dash curves in Figures 66 through 72. Thresholds for the decision element were also derived which would minimize the number of errors on the generalization samples. The numbers of errors using these thresholds are shown by the long-dash curves in these figures. It is often the case that when a recognition network is constructed, the decision element threshold is left adjustable to compensate for variations and drift in the network. The thresholds are then set to give the desired performance on a standard set of patterns. The two curves shown thus represent some indication of the performance to be expected, depending upon how lucky or unlucky the choice of thresholds was.

Networks No. 1, 2, and 3 are designed under the same conditions — a different starting number for the random number generator provides different sets of input connections for the candidate property filters. Thus the three designs represent the variations which occur in network design. Networks 4 and 5 are designed using restricted size input fields for the property filters and are discussed in Section 6.4.6. Networks 6 and 7 are designed on lower resolution samples, but are tested against standard resolution generalization patterns. They are discussed in Section 6.5.2. Except as noted, the remainder of this section is concerned with Networks 1, 2, and 3, as shown in the first parts of Figures 66, 67, and 68.

For each of the three networks, at some point in the network design, generalization performance provides at least 61% correct recognition — the best value overall being 63-1/2%. Network 1 appears to provide the best overall performance. Almost all of the seven networks show a significant local minimum near the middle of the design. Most of them show a lower minimum near the beginning or end of the design, usually the beginning. The mid-design minimum occurs when the error rate on the design patterns is 5 or 6% (in one case, 2.5%).

Network 1 shows its best values near the beginning of the design, with an error rate of 36.5% for both thresholds. At the mid-design minimum, the error rates are 37.5 and 39.5%. The highest error rate observed is 46%. All of these figures are the best obtained with any network. The network contains 250 property filters.

Network 2 also achieves its best performance near the beginning, where the error rates are 38 and 43.5%. At mid-design, they are 41 and 42.5%. The highest error rate is 48%. This network also has 250 property filters.

Network 3 is the smallest of the seven networks, using only 220 property filters. It shows only the mid-design minima at which point the error rates are 39 and 44.5%. At one point in the

design, performance against the preset threshold is not better than chance.

All three networks show generally poor generalization results, but results which are nonetheless consistently better than chance.

6.4.5 Continuation of Design

To determine whether or not the poor generalization performance could be due to early termination of design, the first three networks were tried in various combinations. The results are shown in Figures 66, 67, and 68.

No combination of networks results in better performance than Network 1 by itself does, but the variability throughout the latter part of the combination is considerably reduced. Thus, when Network 1 is added to Network 3 in Figure 68, the minimum error rate is nearly constant at 40% through the second half of the design.

When Networks 1 or 3 are added to another network or combination, a small improvement is noted over the final performance level of the preceding portion of the system. When Network 2 is added to other designs, performance levels decrease somewhat. The reason for this is unknown.

The conclusion from these results is that early termination of design is not the cause of the poor generalization results. The results must be due either to the nonrepresentativeness of the design sample of patterns or to a tendency of the design techniques to design by templating for perfect performance on the design samples. The results on the alphabetic characters indicate that the technique does not lead to templating. It seems likely, then, that the performance levels reached are due to nonrepresentativeness of the sample.

6.4.6 Input Field Restriction

For the general design program used to design the first five networks, input connections for the candidate property filters

are obtained by randomly selecting (without replacement) ten points from the entire 180-by-180 input field. The rationale for this is that candidate property filters with an assortment of input field sizes would be generated. Since only one out of 15 candidates is selected, the weeding-out process would select property filters with the most appropriate input fields.

An examination of a limited number of property filters indicated that their input fields are quite spread. Since it is unlikely that all ten connections would come from a small area, the fields of these units were re-examined after first discarding the connections with weights whose magnitude was smaller than "1", that is, the less important connections. The remaining connections still covered a wide area, but tended to occur in closely spaced pairs, one member of the pair having a positive weight, one having a negative weight.

Two networks were designed to determine whether or not the property units should have smaller input fields. In the design of Network No. 6, the input connections for each property filter were restricted to a 90-by-90 subfield, the location of the subfield being selected randomly and independently for each candidate unit. The same procedure was followed for Network No. 7, except that 45-by-45 point subfields were selected. Once a subfield was established, the ten input connections for the property filter were selected randomly (without replacement) from the subfield.

Figures 63 and 64 present the design data for these two networks; Figures 71 and 72 show the performance on the design and generalization patterns. For the 90-by-90 subfields, 280 property filters were required for separation of the sample patterns. For the 45-by-45 case, 290 property filters were needed. The average number of property filters in the five networks, with unrestricted fields, was 242. The added units seem to result from the inability to remove the last few errors in the design deck. Since this is usually accomplished (see Section 5.4.3) in terms of the peculiarities of individual patterns rather than properties which aid in generalization, this increase does not seem too important.

The generalization results obtained with the 90-by-90 restrictions are somewhat poorer than those obtained with no restrictions on the input fields. Generalization performance with the 45-by-45 network is noticeably poorer than the no restriction case. From this it is concluded that there is little benefit from seeking more localized properties, and indeed there is some deterioration in the results achieved. It is possible that if the property filters had fewer input connections (and were thus unable to measure several independent properties), the input field restrictions might be beneficial.

6.5 Resolution Studies

The determination of the resolution necessary for an automatic recognition device is a matter of some importance. If the resolution used is too high, the input device will be more complex than necessary, the number of property filters will be unnecessarily large, and the wiring complexity or program running time will be unnecessarily high. On the other hand, if the resolution used is too low, the rate of misclassified patterns will be higher than it needs to be.

Two studies on the resolution requirements were conducted. In one of these, the requirements of individuals not trained as meteorologists were investigated. The resolution of a number of frames was modified photographically, and the resulting prints analyzed by a number of Astropower personnel. In the other study, recognition networks were designed using sample patterns whose resolution had been altered on a computer. Two such cases were tried, one in which the resolution was one-half the original level, and one in which it was one-third the original level.

6.5.1 Photographic Studies

Accuracy in differentiating vortex from nonvortex cloud patterns depends on the resolution of the print under examination, on the type of print, on the type of examiner, and finally, on the "forcefulness" of the patterns themselves. The resolution investigation is, first,

an attempt to separate patterns into two major categories. The first of these categories would contain patterns which are never accurately classified, regardless of resolution. The second category would contain patterns which do, eventually, get correctly classified. The existence of such categories implies, of course, that the examiners are investigating patterns previously classified by a reliable source (e.g., a meteorologist), and that they are not as good at this job as he is.

The resolution investigation is, next, an attempt to push back the level of resolution which was sufficient for placing a pattern into the second category. This is the major part of the investigation.

The method used for producing prints will be called the "screening process," since it involves overlaying a negative with a screen. This process transforms a negative into a grid of alternating black and white spaces. The number of such spaces determines the resolution. Thus, a resolution of 100 lines indicates the presence of a horizontal sequence of 100 squares, 50 black and 50 white. Although the total space occupied by a black and white pair of squares remains fixed throughout a print for a given resolution, the relative portion of this space covered by the black square varies in a manner which reflects the contrast on the original negative. Thus, the reproduction of the pattern of the original photo is accomplished by the local ratio of black to white space. Clearly then, finer resolutions are more adequate in preserving patterns, since they allow the contrast from the original to be reflected by the black-white ratio of a larger number of square-pairs for the same amount of space. There may be patterns which require a very high resolution in order to be recognized, while others are evident in very crude reproductions. When it is a cloud pattern that is under consideration, the obvious structure to be preserved throughout transformations of the original picture is the presence of a vortex, if there be one. Thus the resolution problem, relative to cloud patterns, is the determination of the minimum resolution compatible with the preservation of recognizable vortex structures.

The resolution investigation involved three tests. On each occasion a group of people were shown prints at various resolutions. For the first two tests, the prints were exhibited one resolution at a time, working from coarsest to finest levels. Each print was identifiable by a code number. Thus, the code 30-150 meant picture 30 at resolution 150. The examiners were given a sheet of paper on which they recorded their decision regarding any particular print — N for nonvortex, V for vortex. For the third test, the previous two-number code was replaced by a single number (for instance, 30-150 might become 75), and the prints were exhibited randomly, i.e., by mixing resolutions rather than handling levels one at a time. This change of procedure was motivated by the desire for independent decisions on each print. For instance, in Tests I and II, the coding allowed pairs such as 30-150 and 30-329 to occur. Someone deciding for the print 30-329 might remember his decision for print 30-150 through the recall effect of the 30. This interdependence might be negligible, since the prints get finer as the test proceeds, and an examiner is unlikely to allow a previous decision from a poor model to modify the decision occasioned by the more detailed later print. However, the method employed for the third test removed such interdependence, negligible or not, and was thus preferable.*

The first test involved six different prints, four vortex, two nonvortex, at nine resolutions each:

40, 60, 88, 100, 111, 122, 133, 150, 329.

These were examined by six people.

The second test involved sixty prints at four resolutions each:

95, 122, 133, 329.

Forty-five were vortex, fifteen nonvortex.

Finally, for the third test, fifteen people examined fifteen prints at each of the following resolutions:

133, 150, 165, 180, 200, 220, 329.

*The entire print processing procedure was carried out by the Dean Hesketh photographers.

The examiners were "amateurs," i.e., had no special meteorological training, nor, for that matter, any extensive acquaintance with cloud patterns. The prints from the first test were glossy, needlessly complicating the examination. This irrelevant complication was removed from tests two and three by the use of matte (nonglossy) prints.

The first test was constructed for the determination of those resolutions which were "good" and "near good" as opposed to those which were too coarse to be of any use. The latter turned out to be levels 40, 60 and 88. Correct classifications were almost unanimous for the 150 and 329 levels, while the remaining resolutions were correctly classified about half the time. For this small sample, then, the "dividing line," if there were one, lay somewhere between 100 and 150. Hence, the second test utilized resolutions of 95, 122, 133 and 329; the first three levels aimed at pinning down the "dividing line," the fourth was a "safety area." If correct classification were poor for this very fine resolution, one couldn't ascribe this failure to lack of detail; one would have to conclude that the pattern was simply too complicated for correct analysis by an untrained eye. However, if classification at this level were largely successful, then one could search the lower levels to determine where such adequate classification began.

The test results divided patterns into three classes. There were those which were never successfully classified; the number of such patterns was far from negligible. There were others whose classification was correct even for the coarsest resolution. Here the pattern was so strong that even coarser resolutions might have sufficed. There weren't too many of these. Finally, the largest class (though not too much larger than the first class) contained those patterns which were correctly classified at level 329, but not earlier. There were no nonvortex patterns in this category. The reason for this is fairly obvious. A poorly detailed picture has the formlessness associated with nonvortex patterns, hence all low resolution prints tend to be classified as nonvortex.

For the third test, levels 133 and 329 were carried over to fix upper and lower bounds. The fifteen patterns chosen were taken from the sixty used in Test II; these fifteen were drawn from each of the three classes distinguished in the latter test. Of course, the "very easy" and "very difficult" patterns could only be expected to remain so, since the extreme bounds chosen for this final examination were taken from the previous test. However, the gradations between these limits could be used to decide a minimal resolution for the category of those patterns incorrectly classified at 133 but correctly classified at 329. As it turns out, "most" such patterns were correctly classified at the 150 level. This tempts one to conclude that all patterns correctly classified at level 329 will be correctly classified as of level 150. However, due to the smallness of the sample class, this would be an unwarranted conclusion. A larger sample might reveal strong differences in those patterns well classified at the 300 level.

The results of Test II and Test III are given in Table XI. The notation for the data is to be interpreted as follows:

ηV means that the last η levels were classified V, while the immediately preceding level was called N. A similar statement holds for ηN. For instance, pattern "4" in Test II is classified:

once as 4V	- all levels were classified V.
once as 3V	- levels 122, 133, and 329 were classified V.
eight times as V	- eight examiners answered V for level 329, but N on level 133.
eight times as 4N	- eight examiners classified the pattern as N on all four levels.

From the present investigation the following meager conclusions may be drawn:

(1) There exist a good many "hard" patterns — an example is pattern 150 (Figure 73). No resolution level seems to lead to correct classification by all of the examiners.

TABLE XI

CLASSIFICATION RESULTS IN RESOLUTION STUDY

Test II

(Levels 95, 122, 133, 329)

4V	1					1	14	7	20	1	18	2	1	3		1	7
3V						1	2			1			2	1			
2V											1		1	1			
V	3	2	3	5	3		1		8		12	6	2				1
N																1	
2N																1	
3N																2	
4N	8	9	9	15	5	1	5		8		3	1	4	12	13		5

Test III

(Levels 133, 150, 165, 180, 200, 220, 329)

7V	1			1			8	7	15		15	6	1	5			7
6V		3		1	3					7		3	14	1			
5V					2												
4V	2			1	1												
3V							1			1				1			
2V	2				1					2							
V	4	2	5	1	11	2	2		3		1			6			2
N																1	
2N														1			
3N																	
4N																	
5N																2	
6N																	
7N	4	4	3	4	2		3				1			1	15	11	3
	150	337	309	31	16	323	24	48	4	46	27	30	224	80	98	24	

Print Number

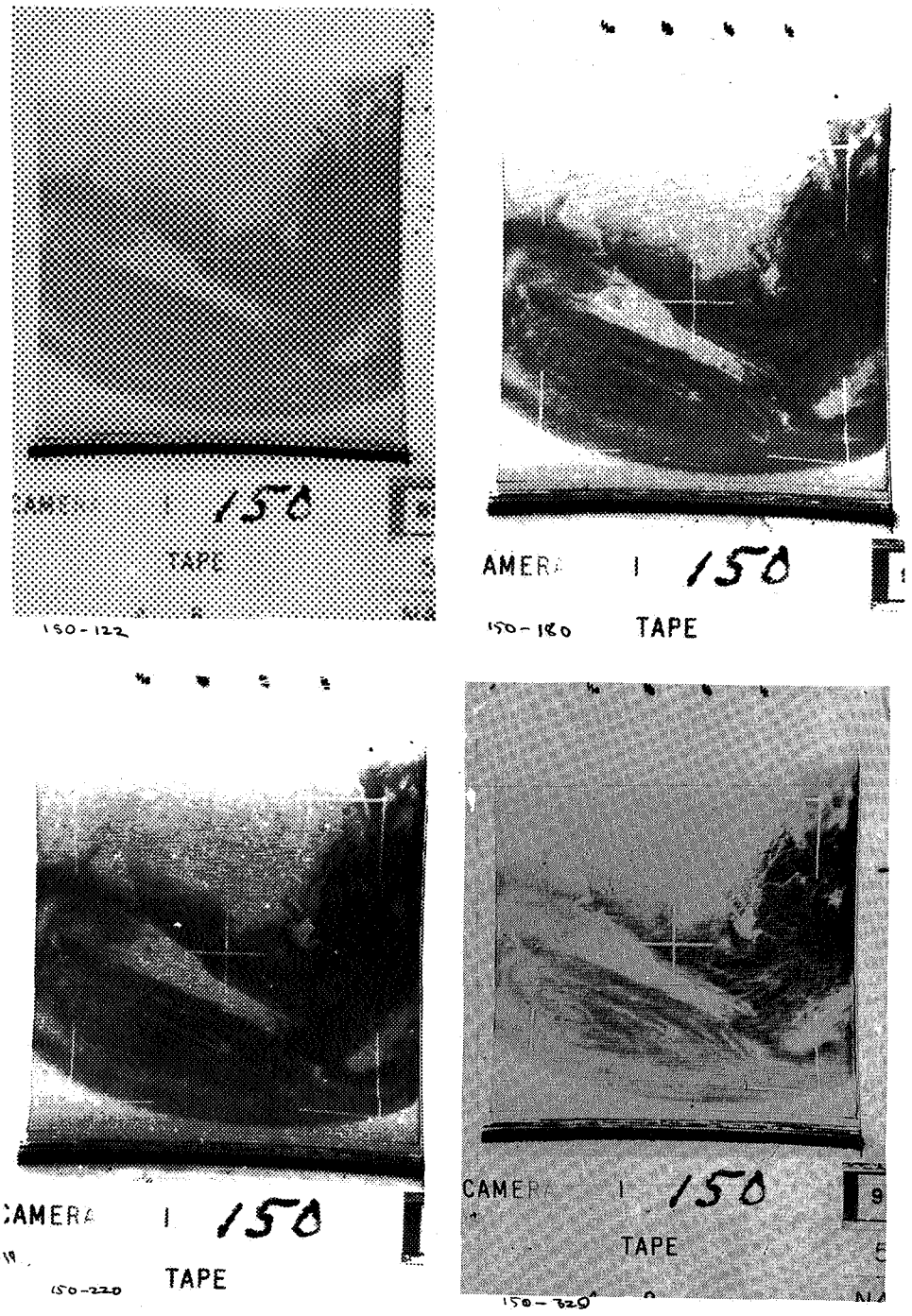


Figure 73. Pattern 150 at Resolutions of 122, 180, 220 and 320

(2) There exists a fairly large collection of "easy" patterns — examples being patterns 48 and 80 (Figure 74 and 75). The "easiness" is deceptive, since a good many of these are nonvortex. However, 48 is a strong vortex pattern.

(3) Most patterns which are successfully classified at level 300 are correctly classified at level 150. An example is pattern 30 (Figure 76).

If this process were to be considered further, it might be worthwile to allow more than two classifications for patterns. A third classification would be "U" for uncertainty — in this case the examiner would be directly singling out difficult patterns, and the V and N could be modified with W and S, signifying "weak" and "strong." Nonetheless, the results were sufficiently indicative to provide a range of resolutions for the simulated designs of the recognition system. The simulation experiment is discussed in the following section.

6.5.2 Computer Studies

The optical studies indicate that generally the resolution cannot be reduced from the 240 lines per frame without interfering with the essence of many of the patterns for visual recognition. With the editing that is performed, this resolution level provides the 180-by-180 patterns which were used in the main investigation.

It was hoped that by designing networks on a lower resolution sample of frames, an indication would be available from the generalization results as to whether an automatic recognition system would require as high a resolution level as people do. The generally poor generalization results tend to mask any such results. The photographic studies indicate that the fraction of frames which are recognizable at lower resolution levels is sufficient to accommodate the recognition levels achieved.

A second purpose of the resolution study was to get an indication of the nature of the properties being extracted. If the

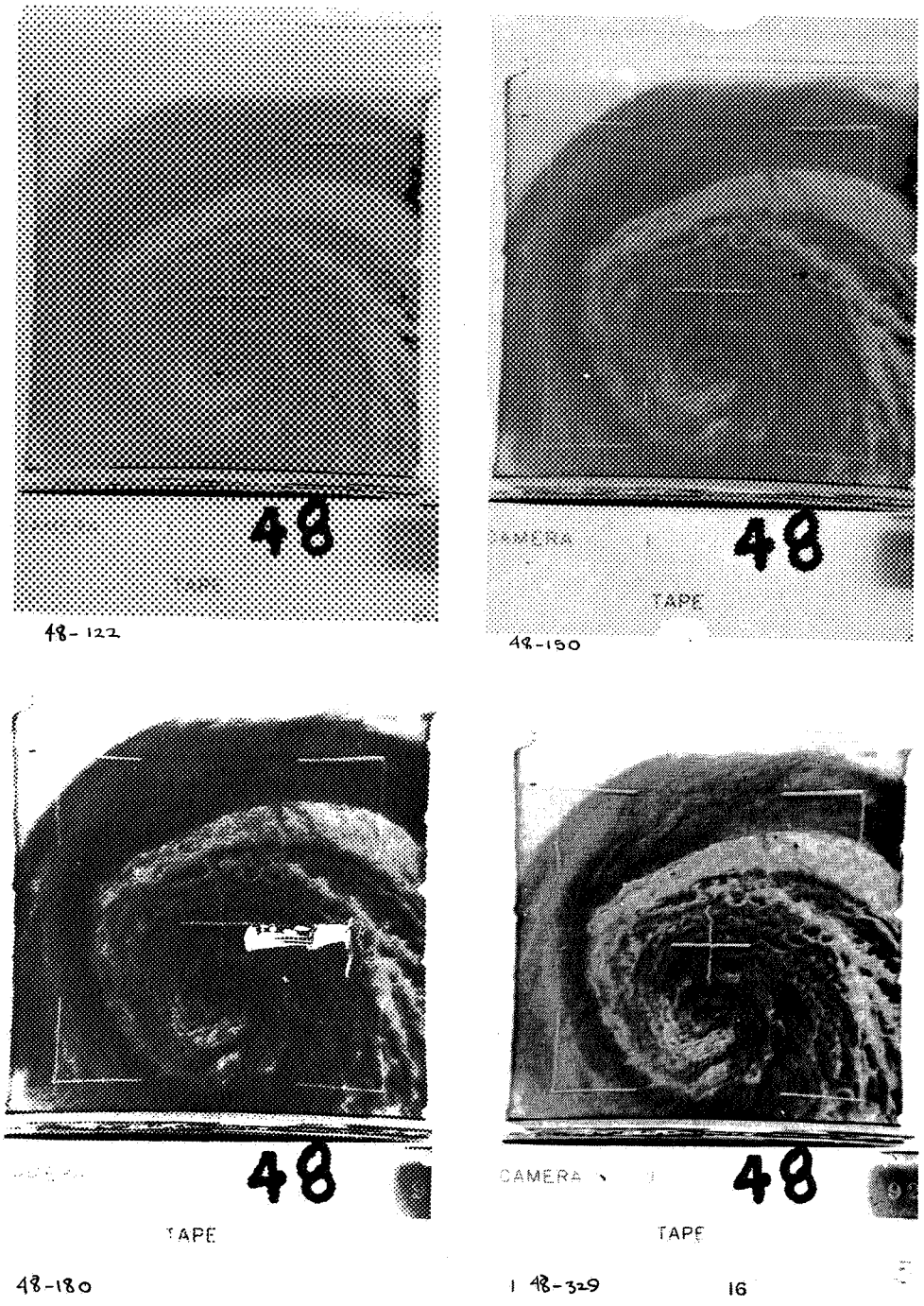


Figure 74. Pattern 48 at Resolutions of 122, 150, 180 and 320

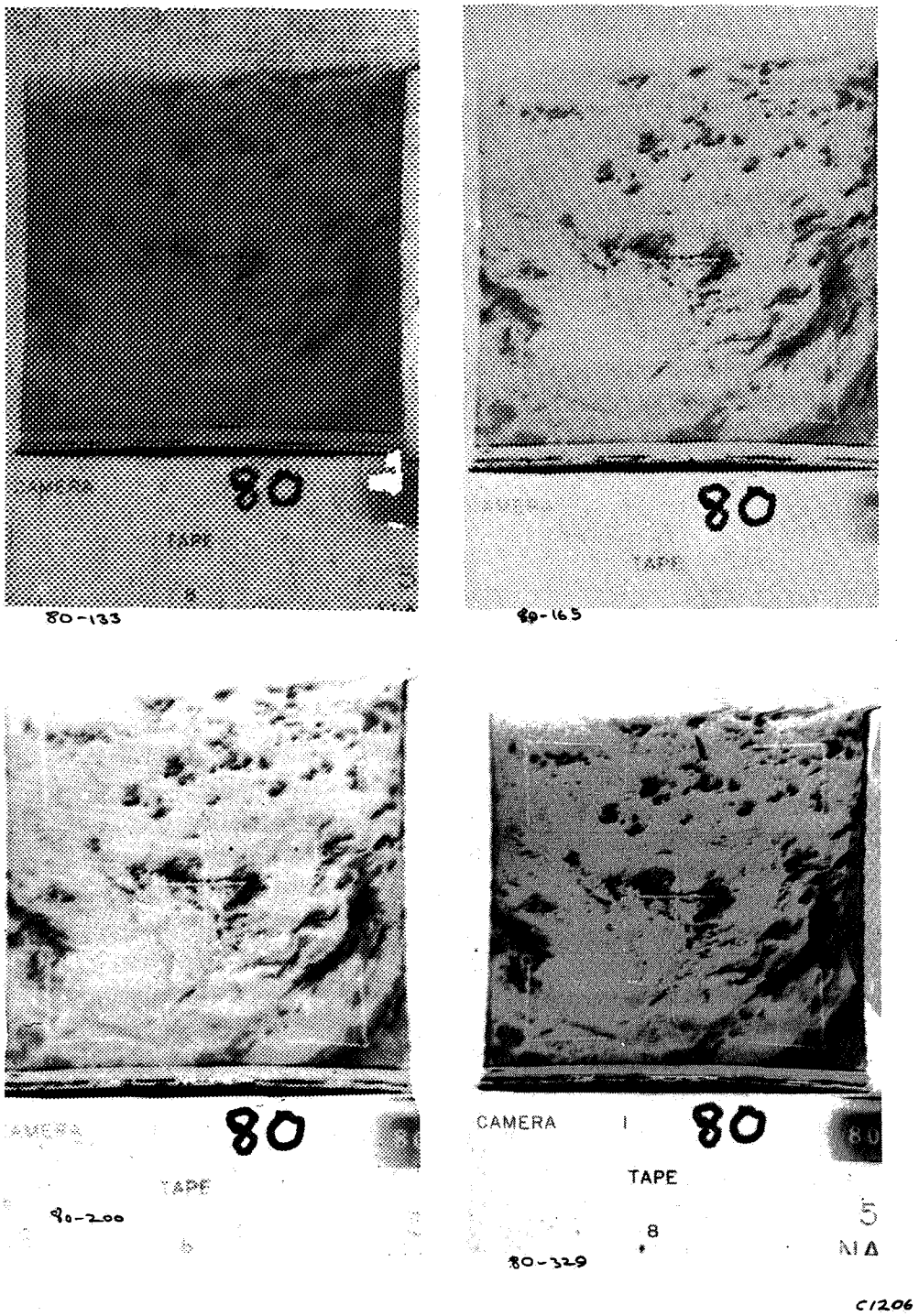


Figure 75. Pattern 80 at Resolutions of 133, 165, 200 and 329

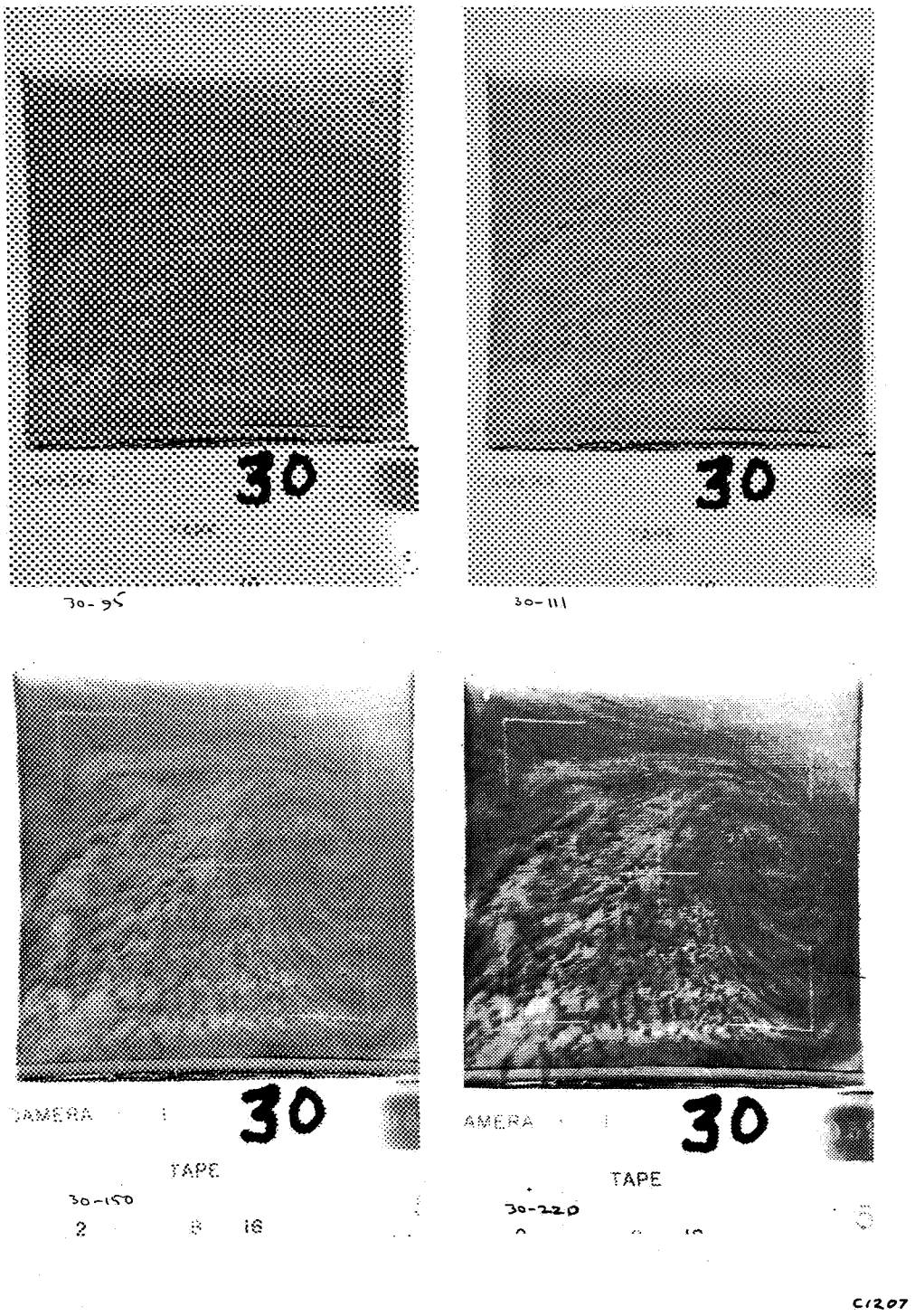


Figure 76. Pattern 30 at Resolutions of 95, 111, 150 and 220

number of logic units required for perfect performance on the design patterns increases as the resolution is lowered, it may be some indication that the design technique has been forced to seek different properties. Since the reduction in resolution tends to mask detail in many of the vortex patterns, an increase in the number of property filters might indicate that the design technique was dealing with the proper level of detail. Such extraneous features as the location of the horizon or the location of the fiducial marks are not masked by the resolution reduction.

Two reduced resolution files of sample patterns were generated by taking four or nine neighboring points, averaging their gray levels, and reading the average value back into all four or nine locations. The resultant frames are 180-by-180, but the gray level changes only every second or third point. This method was used, rather than actually reducing the picture to a 90-by-90 or 60-by-60 frame, so that the resultant networks could be tested against the generalization sample at full resolution.

One network was designed on the 90-by-90 patterns, and one network was designed on the 60-by-60 patterns. The design data are shown in Figures 61 and 62. The former network required 240 property filters, the latter 250. Since the average for the three networks designed on the 180-by-180 patterns is 240 logic units, the increase does not seem to be significant. Generalization performance for the two networks was tested only against the full resolution generalization patterns. The results are shown in Figures 69 and 70. The network designed on the 90-by-90 patterns shows a marked deterioration in generalization performance, a deterioration which is somewhat greater than the variability between the three networks designed on the full resolution patterns. This encouraging sign is offset by the fact that the generalization performance achieved with the 60-by-60 design shows a substantial recovery. In each case the same sets of input connections were supplied to the candidate property filters as were supplied in the design of Network No. 3. This network provided the second best generalization performance.

6.6 Summary of Results

This section summarizes the results obtained in experiments on TIROS frames.

(1) Four files of digitized TIROS patterns were generated. The first and third files are frames containing vortex patterns; the frames in the second and fourth files contain nonvortex cloud cover. The first two files contain 500 patterns each, and were used in network design; the last two files of 100 patterns each were used for testing the networks. The 1200 patterns are derived from 223 actual TIROS frames, by considering different translations and rotations of these pictures. Difficulties were encountered in obtaining and verifying these patterns. The TIROS frames were edited to a 180-by-180 size, and the gray scale reduced to four bits.

(2) Seven complete recognition networks were designed, and four fractional networks were designed. A total of 48 hours of IBM 7094 time was used to design these networks and two hours were used to test their generality. These figures are dwarfed by the amount of time used in program debugging.

(3) The partial network designs were obtained to adjust two parameters of the design technique. One of these parameters controls the expected fraction of nonvortex patterns which turn each property filter on (activity rate), the other parameter controls the rate of design and the stability of the weight adjusting procedure. Three partial networks were available for selection of each parameter. Instability did not occur in these networks (although it had with different values in the debugging runs). Values which gave the fastest design rates were selected for these parameters. In retrospect, this criterion does not seem optimum. The parameter value selected by this method gives an expected activity rate for nonvortex patterns of one-half of the property filters.

(4) Approximately 250 property filters are required to provide perfect separation of the 1000 patterns in the design files. The actual figures range from 220 to 290. Designs with more than 250 units occurred only when constraints were placed on the property filters. Steps to terminate the design process are taken when perfect performance on the design patterns is obtained. This termination procedure is such that it produces designs with ten property filters more than required for the zero error rate. These units, and the additional adjustments made to the output weights, appear to reduce performance levels rather than improve them. The error rates on the design sample drop rapidly until they reach 1%, at which point a sharp break is noted in the rate of reduction. Approximately 180 property filters are required to achieve the 1% error rate.

(5) Two generalization error rates were computed, corresponding to different thresholds for the decision element. One of these thresholds gives a minimum error rate on the design sample of patterns, the other gives a minimum rate on the generalization patterns. These error rates are computed for each increment of ten property filters to the network. The error curves generally show two significant local minima. Usually these minima occur near the beginning of the design and in mid-design. Occasionally, the minimum near the beginning is replaced by one near the end of the design. The mid-design minimum occurs when the error rate on the design patterns is about 5 or 6%. The mid-design minimum is usually the larger of the two.

(6) The first three networks were designed under standard conditions. Different random numbers provide different input connection combinations and hence different networks. These networks provide the best generalization performances of the seven networks, giving error rates which are not particularly low. Network 1 has 250 property filters. The first minimum gives a 36.5% generalization error rate with both thresholds — the lowest rate observed. The mid-design minimum gives the rates of 37.5 and 39.5%. The highest error rate is 46%. Network

2 has 250 property filters, early error rates of 38 and 43.5%, and mid-design minima of 41 and 42.5%. The maximum error rate is 48%. Network 3, with 220 property filters, exhibits only mid-design minima which are 39 and 44%. At one point in the design an error rate of 51% is observed. Thus each network is capable of making less than two errors in each five decisions at some point in its design. Although these performances are not high, the networks do achieve better than chance performance quite consistently.

(7) When the first three networks are considered in combination, the generalization error rates are not better than those for the best single network. The error rate curves are smoother, however. Failure to improve the single network performance is indicative of a nonrepresentative design sample rather than too fast a design rate.

(8) Networks 4 and 5 were designed on reduced resolution design patterns, using 90-by-90 and 60-by-60 rasters, respectively. The generalization performances were tested on full resolution patterns, however. Network 4, with 240 property filters has early minima of 41 and 42%, and mid-design minima of 43.5 and 46.5%, these latter occurring at different points in the design. Performance no better than chance is observed twelve times in the design. Network 5 has 250 units and shows pronounced minima only near the end of the design. These minima are 41 and 41.5%. Performance no better than chance is observed ten times in the design. These figures represent significant deteriorations from the first three networks. Combined with optical studies, which indicated that full resolution is necessary for many patterns, this result is indicative that the property filters in the first three networks, at least in part, are dealing with the proper level of detail.

(9) Networks 6 and 7 were designed with restricted area subfields for the property filters, No. 6 with 90-by-90 subfields, No. 7 with 45-by-45 subfields. They required 280 and 290 property filters. The rate of design was similar to the other networks until a one or two

per cent error rate on the design patterns was achieved. Unlike the first four networks, the generalization error rate did not increase during the remaining network design. The minimum error rates for Network 6 are 41.5 and 44.5%. Performance is no better than chance seven times. For Network 7, the minimum error rates are 43.5 and 45%. The design fails to better chance performance 19 times. These results are significantly poorer than those with the first three networks. The conclusion is that local properties are not desirable for this problem when ten input connections are used for each property filter.

(10) Optical studies were performed on TIROS frames. The resolution of 60 frames was altered to several levels using a screening process. The prints obtained were then examined by 15 to 20 people, in order of increasing resolution. The resolution level at which their decisions became consistent with their final judgment was noted. For a remarkable number of frames, the panel disagreed in their final judgment. When the resolution was too low, the frame was almost always judged nonvortex. Consequently, when the final judgment was nonvortex, all preceding judgments were also. About half of the vortex decisions were reached at the highest resolution level (240 lines per frame). The other half were made at the coarsest resolution levels (about 70 lines per frame). Very few final judgments were made at intermediate levels.

REFERENCES

1. "Cloud Pattern Interpretation," Final Report, Astropower, Inc. Report 129-F. NASA Contract NASw609 (August 1963).
2. Katz, Y. and W. Doyle, "Automatic Pattern Recognition of Meteorological Satellite Cloud Photography," RAND Corp. Memorandum RM-3412-NASA, Santa Monica, December 1964.
3. Holford, W. L., "Conflex Experiments" presented at the Automatic Target Recognition Meeting held at Philco Corp., Newport Beach, California (1965).
4. Gerdes, J. W. "Automatic Target Recognition from Aerial Photography," presented at the Automatic Target Recognition Meeting, Newport Beach, California, (1965).
5. Joseph, R. D., Kelly, P. M. and Viglione, S. S., "An Optical Decision Filter" Proc. of the IEEE, (August 1963).
6. Randall, N., "Progress in Image Screening," presented at the Automatic Target Recognition Meeting held at Philco Corp., Newport Beach, California (March 1965).
7. Fischer, R. A., "The Use of Multiple Measurements in Taxomic Problems," Ann. Eugen., 7, pp. 179-188 (1936).
8. Wald, A., "On a Statistical Problem Arising in the Classification of an Individual into One of Two Groups," Trans, Amer. Math. Soc., 54, pp. 426-482 (1944).
9. Daly, J. A., Joseph, R. D., Ramsey, D. M., "An Iterative Design Technique for Pattern Classification Logic," presented at WESCON (August 1963).
10. Kanal, L. N., and Nambiar, K. K., "On the Application of Discriminant Analysis to Identification in Aerial Photography," presented at 7th Mil-E-Con, Washington, D.C. (1963).
11. Sebestyen, G. S., Decision-Making Processes in Pattern Recognition, Macmillan Co., New York (1962).
12. Daly, J. A., Joseph, R. D., and Nelson, E. E., "Time Varying Threshold Logic," Biophysics and Cybernetics Systems, Maxfield, Callahan and Fogel, Eds., Spartan Books, Washington, D. C., (1965).
13. Rosenblatt, F., "The Perceptron: A Theory of Statistical Separability in Cognitive Systems," Cornell Aeronautical Lab. Report No. VG-1196-G-1 (January 1958).

14. "Research in Systems Theory, Devices, and Physical Phenomena for Microsystems Electronics," Stanford Electronics Labs Rpt AL-TDR-64-81 SU SEL-64-037 prepared under Contract AF33(616)-7726 (June 1964), pp. 84-93.

15. Nilsson, N. J., "The Stanford Research Institute Minos II Learning Machine Facility," presented at the Automatic Target Recognition Meeting, Newport Beach, California (1965).

16. Mosteller, F., and Wallace, D. L., "Notes on an Authorship Problem," Proc. of a Harvard Symposium on Digital Computers and Their Applications, Harvard Univ. Press, Cambridge, (1962).

17. Mosteller, F., and Wallace, D. L., Inference and Disputed Authorship: The Federalist, Addison-Wesley, (1964).

18. Rosenblatt, F., "On the Convergence of Reinforcement Procedures in Simple Perceptrons," Cornell Aeronautical Lab. Report No. VG-1196-G-4, (February 1960).

19. Fein, F., "The Artificial Intelligentsia," IEEE Spectrum, Vol. 1, No. 2, (February 1964).

20. Highleyman, W. H., "Linear Decision Functions, with Applications to Pattern Recognition," Proc. IRE, Vol. 50, No. 6, (June 1962).

21. Rosenblatt, F., Principles of Neurodynamics, Spartan Book, Washington, D. C. (1962).

22. Wiegman, E. J., Hadfield, R. G., and Serebreny, S. M., "Atlas of Cloud Vortex Patterns Observed in Satellite Photograph," Final Report, SRI Project No. 4516, Stanford Research Institute, Menlo Park, California, April 1964.