[35] Tony Leal

Final PR-1032-77-418
April 29, 1977

ACALP ALGORITHM SELECTION AND APPLICATION

Moshe Ben-Bassat
Antonio Leal

Prepared For:

TRW Systems Group
One Space Park
Redondo Beach, California 90298

PERCEPTRONICS

6271 VARIEL AVENUE • WOODLAND HILLS • CALIFORNIA 91367 • PHONE (213) 884-7470

Final PR-1032-77-418
April 29, 1977

ACALP ALGORITHM SELECTION AND APPLICATION

Moshe Ben-Bassat
Antonio Leal

Prepared For:

TRW Systems Group
One Space Park
Redondo Beach, California 90298

PERCEPTRONICS

6271 VARIEL AVENUE • WOODLAND HILLS • CALIFORNIA 91364 • PHONE (213) 884-7470

TABLE OF CONTENTS

1. INTRODUCTION

1.1 Overview

This report describes a pattern recognition and learning system which can be applied to several Ballistic Missile Defense (BMD) control problems, as well as many problems in other disciplines. The basic problem is concerned with the recognition of objects as either belonging to one of a given set of classes or possibly as a new unforeseen type. In the latter case, the properties of the new type need to be learned by the computer. This classification and learning must be performed in real time and under time constraints. For the sake of clarity and to motivate the algorithm development, it is described as applied to the recognition of a re-entry vehicle threat cloud which is approaching a friendly target.

1.2 The Problem

It is assumed that objects in a threat cloud composed of RV's (Reentry Vehicles) and non-RV's are to be classified and distinguished as to type. The known classes include several types of RV's, decoys (DC), and tank fragments (TF), and will be denoted henceforth by $C_1, C_2, \ldots, C_M$. It is also possible for the cloud to contain objects which are not members of any of the known classes.

The classification process is dynamic. The cloud is observed continuously and the collected data is analyzed at discrete points of time, $t_1, t_2, \ldots, t_E$, where t_E denotes the latest time for a defense action. (See Figure 1-1.) Final classification of any of the objects can be made if sufficient evidence has been accumulated. If an object is identified as not belonging to one of the known classes and temporal considerations allow, then the characteristics of this new type are learned so that similar objects may be more rapidly and accurately classified. A decision is also

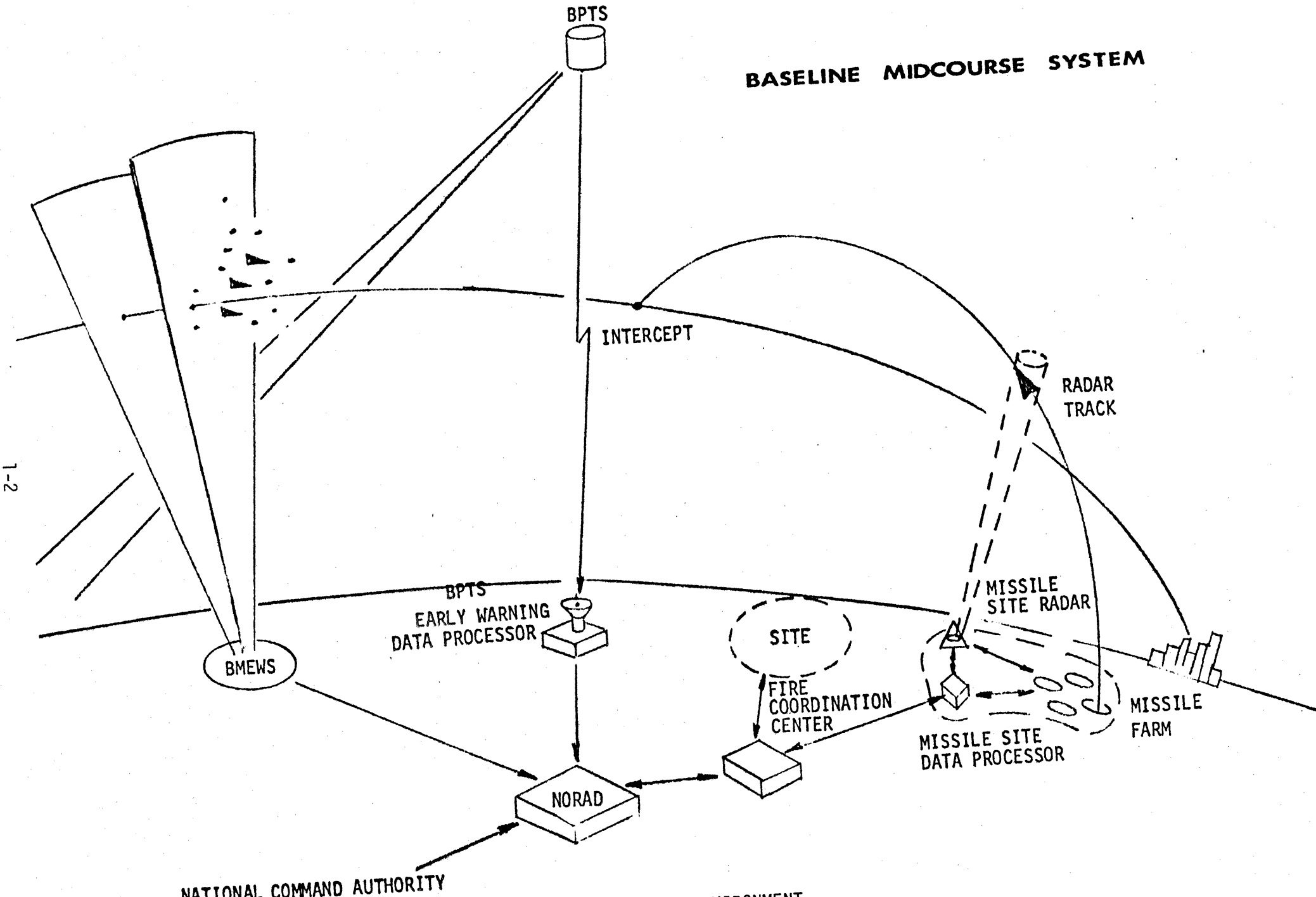


FIGURE 1-1. PROBLEM ENVIRONMENT

made whether the new type is more likely to be an RV or a decoy. At time t_E all the objects need to be classified into either a known class or a new one. A "no decision" is not allowed.

Information features which may be used for classification include: trajectory motion, radiant intensity and scintillation, reflected sunlight, temperature history, tumbling frequencies, etc. The pattern recognition model assumes that the preprocessing required to extract the classificatory features has been made and that at a given point of time, each object is represented by a pattern of its features $(x_1, x_2, \ldots, x_n)$. This pattern can represent a single observation or an average of continuous observations over a period of time. No assumption is made with regard to the distribution of these features.

Object classification is mainly based on its feature pattern. However, for boundary cases, classification can be assisted by any holistic information that may be available. For instance, intelligence reports may provide information that a given RV cloud contains at most three RV's. In this case, if three RV's have already been identified, then a boundary case is more likely to be a non-RV. Other holistic information can be derived from physical characteristics of the whole cloud, for instance, a cloud of volume (weight) V cannot carry more than x RV's. Clearly, the reliability of holistic information plays a prominent role in determining the rules which are derived from it.

The main characteristics of this object classification problem can be summarized as follows:

1. Classification of many objects in real time.

2. Classification as early as possible, when the data is of adequate quality for a decision rather than after a predetermined amount of data has been collected.

3. Recognition of new classes.

4. Learning new class characteristics in real time.

1.3 The Classification Process

This section describes the top level structure of the pattern recognition and learning algorithm. Following sections describe the subfunctions of the algorithm in greater detail.

The three-level system described below and shown in Figure 1-2, processes and classifies one object at a time as each sensor data pattern is presented to it. The pattern submitted at each point of time may either be independent or an average of all the patterns previously observed for this object. Outside of this hierarchial structure is a more general information processing loop that uses any holistic information that may be available to aid the single object classification.

Individual object classification is made in a hierarchial process composed of three main levels:

Level 1 Immediate classification by deterministic rules;
Level 2 Classification by the nearest neighbor rule;
Level 3 Final Classification of Rejected Objects.

During Level I processing, predefined deterministic rules are evaluated in order to classify objects with an obvious and certain identity. For example, if the absolute temperature of an object is below a given threshold, it may be possible to classify it immediately as a non-RV. If this level does not lead to a clear cut decision, the information collected is retained for the next process levels. Notice that if all the classes are eliminated by these rules except for one known class, it does not imply

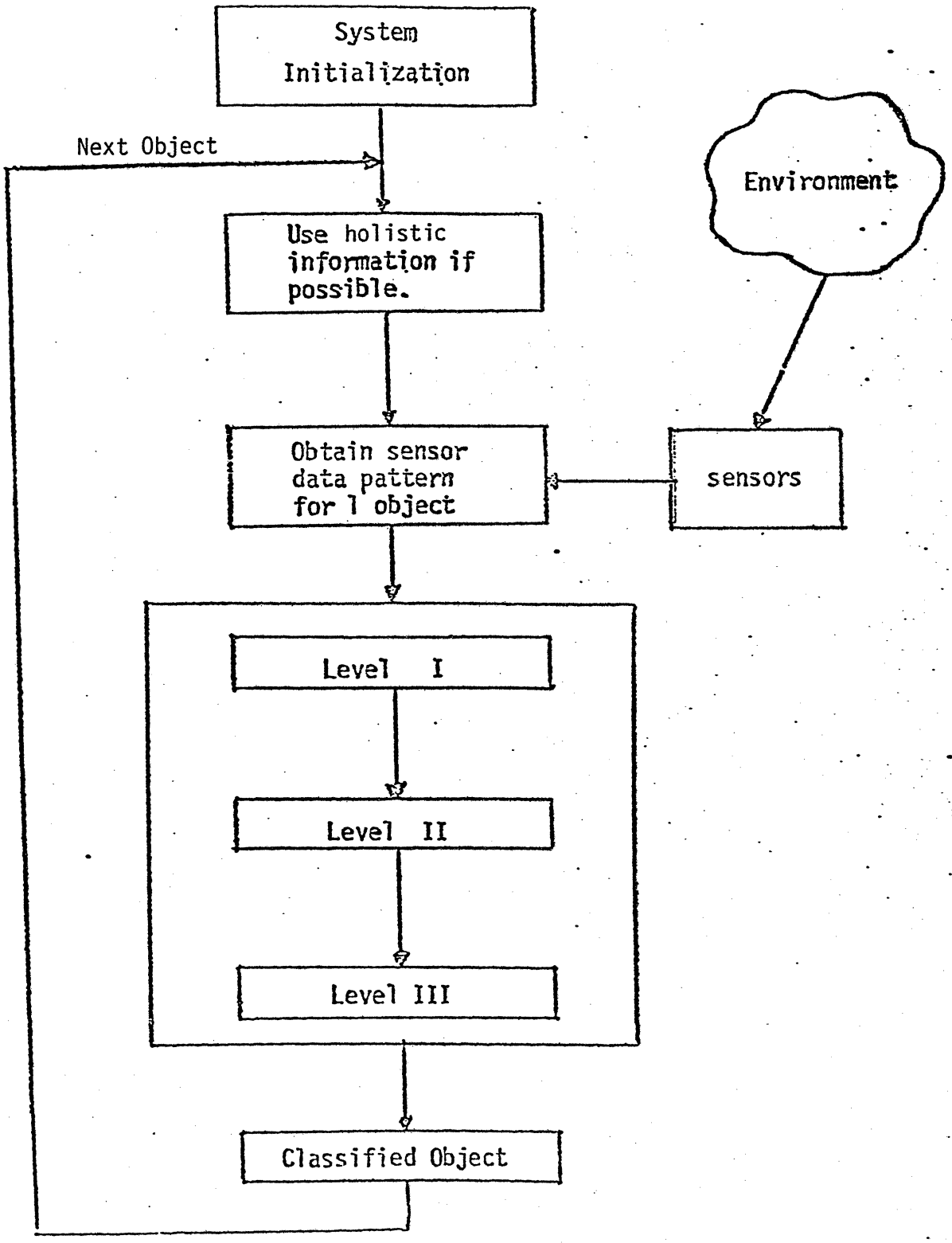


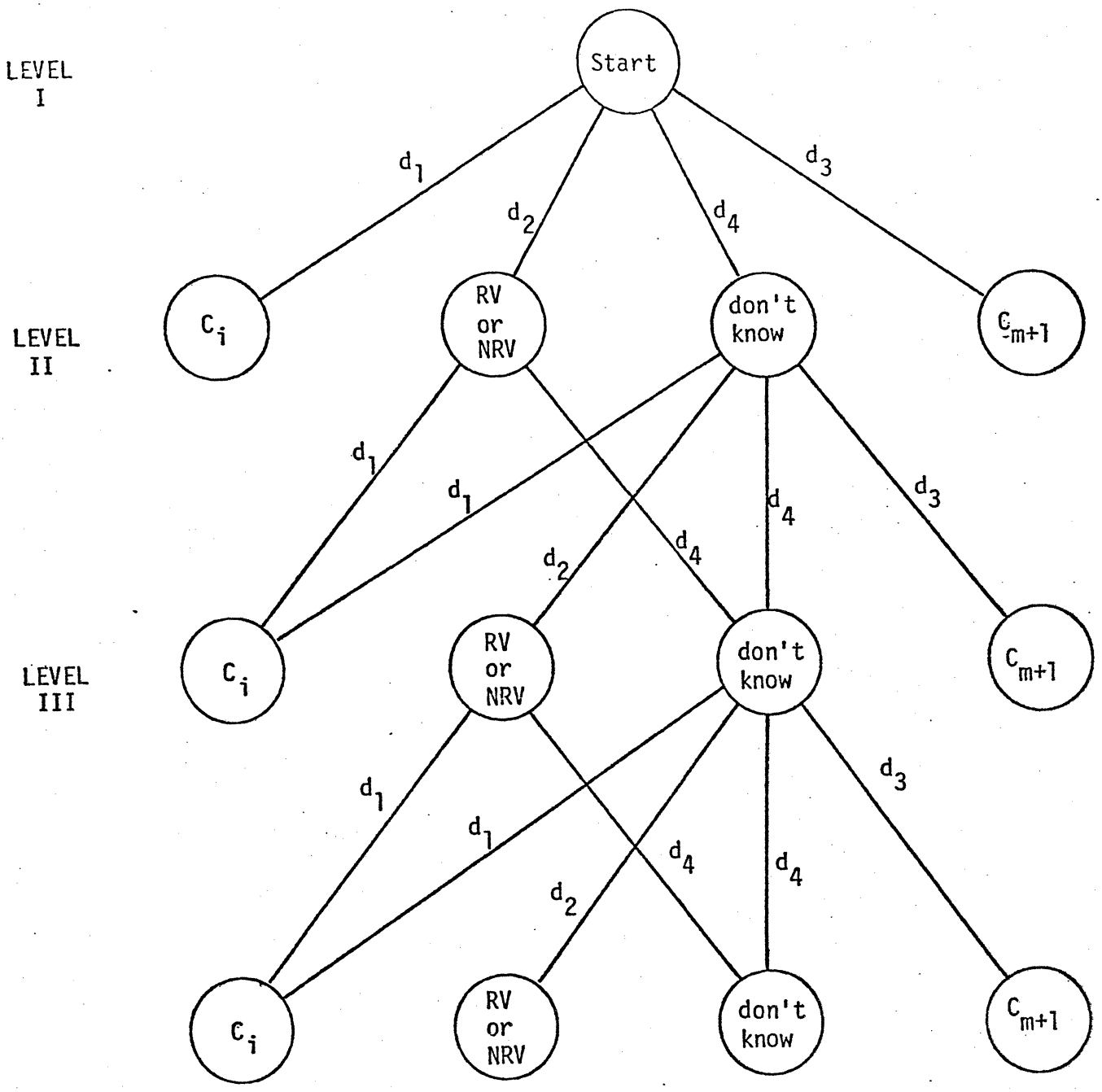
FIGURE 1-2. SYSTEM OVERVIEW

that the object belongs to this single remaining class since it is possible that the object could be of a new type.

In Level II, the object is either finally classified by the k, k′ nearest neighbor rule or rejected and submitted for the third classification level. Again, the information accumulated up to this point is retained for the next level.

Finally, the first task at Level III is to classify the object as an RV or non-RV. The second task is to identify whether this object is one of the old types or is a new type. Both tasks are carried out via a clustering approach.

Figure 1-3 specifies the possible decision outcomes at each level of the algorithm. The following sections will describe the operation of each level individually and their interrelationships.



d_1 - classification as one of the M known classes (C_i)

d_2 - classification as an RV or a non-RV (RV or NRV)

d_3 - a new type (C_{m+1})

d_4 - don't know

FIGURE 1-3. CLASSIFICATION DECISION DIAGRAM

2. SYSTEM DESCRIPTION

2.1 Deterministic Rules (Level I)

Some objects can easily be identified as RV's or non-RV's, or as a certain type of RV or non-RV. In these cases, it is possible to simplify the pattern recognition process by using deterministic rules. A deterministic rule might state that if feature x_1 is not in the range [a,b], then the object cannot belong to class C_1. In our example, the corresponding rule might state that an object colder than 270°K or hotter than 400°K is not an RV. Such rules facilitate and accelerate the classification of obvious cases by using detailed knowledge about the specific problem. Because deterministic rules are inflexible, they must be carefully prescribed to avoid sensitivity to "spoofing". The rule given above could be spoofed by a heated RV, for example, at a non-trivial performance penalty. For this reason, only the most certain deterministic rules should be used. A more effective deterministic rule would classify stationary objects as non-RV's and be unspoofable. Because objects which are not classified by the deterministic rules will be classified at a later stage, only the clearest cases should be classified at this point. Figure 2-1 presents the structure of Level I process.

If, at the end of Level I only one admissible class remains, that is, all other classes have been eliminated, the system proceeds straight to Level III. Otherwise, it proceeds to Level II.

2.2 Nearest Neighbor Classification (Level II)

Any classification method which can incorporate a rejection option is suitable for Level II. With the rejection option, whenever a criterion for incorrect classification (e.g., probability of error) is not satisfied, classification is deferred. This includes parametric or nonparametric

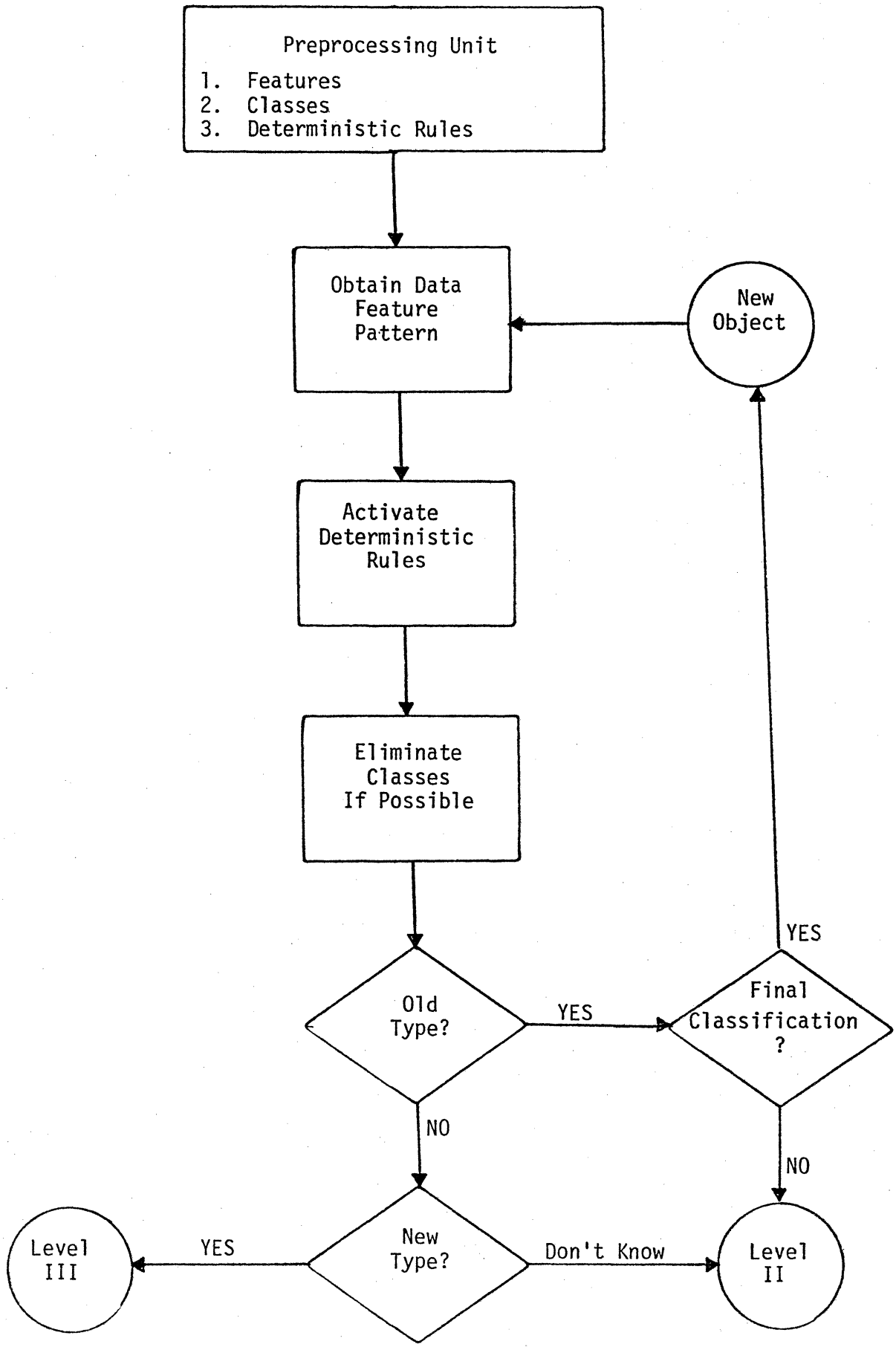


FIGURE 2-1. LEVEL I CLASSIFICATION PROCESS

discriminant functions methods, nearest neighbor methods, or any other ad hoc method. In order not to confine the algorithm to specific assumptions on the nature of data, the (k, k′) nearest neighbor method is adopted.

In the simplest form of the nearest neighbor (NN) method, when a new pattern arrives, its distance from each of the preclassified training patterns is calculated, and it is assigned to the class of its nearest neighbor class. Improved versions of this approach are obtained by considering the distances from the k-nearest neighbors and using a majority rule. The basic structure of the k-NN method is as follows:

Input

NC = number of classes
NT = number of training patterns
NF = number of features (dimension of patterns)
T = $(X^1, X^2, \ldots, X^{NT})$ set of training patterns
L = $(\ell_1, \ell_2, \ldots, \ell_{NT})$ labels of training patterns
X = unknown pattern
d = distance function
k = majority parameter

Procedure

(1) Compute $d(X^j, X)$ for $j = 1, 2, \ldots, NT$

(2) Identify $T_k = (j_1, j_2, \ldots, j_k)$ indices of the k nearest neighbors

$L_k = (\ell_{j_1}, \ell_{j_2}, \ldots, \ell_{j_k})$ labels of the k-NN

(3) Count N_i the occurrence of class i in L_k

(4) Assign X to class c* where $N_{c*} = \max (N_1 \ldots N_m)$

2.2.1 Performance Considerations. Nearest neighbor techniques are attractive due to the asymptotic relationship between the expected probability of error of the NN classifier (P_e) and the optimal Bayes classifier (P_e^*). For a large set of training patterns and k = 1, the following inequality holds (Cover and Hart, 1967):

$$P_e \leq 2P_e^* - \frac{NC}{NC-1} P_e^{*2}$$

Roughly speaking, the significance of this inequality is that P_e is at most twice P_e^* for a large set of training patterns. For k > 1, tighter bounds are obtained.

The main disadvantage of the nearest neighbor methods is that the training patterns are stored in the classifier and used in the classification phase which implies computational difficulties for a large training set. On the other hand, for a limited training set, the attractive asymptotic properties of NN rules may not hold. Hence, the relationship between the size of the training set and the level of performance of NN rules is crucial. Kanal (1974) reviews recent literature on this subject. Algorithms for efficient search of the k nearest neighbors are described by Fukunaga and Narenda (1975); Yunck (1976); and Friedman, Basket, and Shustek (1977). A survey of these techniques is given by Bentley (1975).

2.2.2 Distance Functions. Any metric can be used as a distance function. The distance function most commonly used belongs to the Minkowsky family of metrices:

$$d_p(X^j, X) = \left| \sum_{t=1}^{NF} | x_t^j - x_t |^p \right|^{\frac{1}{p}}$$

When the feature scales are not of the same magnitude and correlated features exist, a weighted distance function is used. For that purpose the covariancce matrix is usually used and a typical distance function is:

$$d(X^j, X) = (X^j - X)\ \Sigma^{-1}\ (X^j - X)$$

where Σ is the feature covariance matrix over the whole set of training patterns. Currently the latter distance function is being used.

2.2.3 The Rejection Option. When the high N_i values are very close to each other, classifying X to c^* involves high risk of misclassification. To avoid this, the rejection option is used by which high potential of misclassification is converted into rejection, according to the following rule:

Let k' denote a threshold majority level. Then, if $N_{c^*} > k'$ assign X to c^*, otherwise reject X. The value of k' is determined experimentally based on our tolerance level for misclassification.

It has been proven theoretically and experimentally that this modified NN rule, which will be referred to as "(k, k') -NN rule", does improve the performance level of the classifier. Exact relations between the reject rate and the error rate can be found in Hellman (1970) and Devijver (1976).

2.2.4 Distance-Weighted k-NN Rule. A variant of the K-NN method is based on the argument that it appears more reasonable "to weight the evidence of a neighbor close to an unclassified observation more heavily than the evidence of another neighbor which is at a greater distance from the unclassified observation" (Dudani, 1976). Pursuing this idea, the following distance weighted k-NN procedure is used:

(1) Determine the k nearest neighbors $T_k = \{j_1, \ldots, j_k\}$ and their corresponding labels $L_k = \{\ell_{j_1}, \ldots, \ell_{j_k}\}$ where X^{j_1} is the closest one while X^{j_k} is the farthest. Let $d_1 \leq d_2 \leq \ldots \leq d_k$ be their corresponding distances from X.

(2) For each one of the k-NN compute:

$$W_j = \frac{d_k - d_j}{d_k - d_1} \quad d_k \neq d_1$$

$$1 \quad , d_k = d_1$$

(3) Count N_i the occurrence of class i in L_k.

(4) For every class i, sum up the weights of the N_i neighbors from that class.

(5) Assign X to the class with the highest total weight.

Experiments with the Distance Weighted k-NN rule reveal better performance (Dudani, 1976). Theoretical proof has not yet been reported.

A possible rejection criterion for this method is the following: Reject the pattern from classification if the highest total weight is less than a predetermined threshold. This criterion is proposed here for the first time and has not yet been examined.

2.2.5 The Edited k-NN Method. Wilson (1972) proposed the following method for cleaning the training set of "poor" patterns. A poor pattern is one that lies in a region where most of the other training patterns belong to a class different from the class of this particular pattern.

For each i, use $\{X^1, \ldots, X^{i-1}, X^{i+1}, \ldots, X^{NT}\}$ as the training set and classify X^i by the k-NN method. If X^i is correctly classified, proceed to the next i. If X^i is misclassified, delete it from the training set and then proceed to the next i with the reduced set.

In addition to reducing the storage required for future classification of unknown patterns, the Edited k-NN method has better asymptotic performance

than the k-NN rule (Wilson, 1972; Wagner, 1973). Although, a recent paper by Pendro and Wagner (1977) points at an error in Wilson's proof, it appears that, except for pathological cases, better performance should be anticipated with this method.

Tomek (1976) raises the following interesting question: Let T' denote the training set obtained from T by applying Wilson's method for editing, "It is natural to ask what would similar editing of T'(leading to T'', T''', etc.) do to the design set. Should we expect progressively better and better classification or will editing of experiments indicate that the described method indeed improves the performance of k-NN classification considerably ..." (Tomek, 1976). However, Tomek was unable to prove anything about the extension of Wilson's result to repeated editing.

2.2.6 Implementation. Consider the two-dimensional situation illustrated in Figure 2-2. For this case, all the k nearest neighbors of X belong to C_2 and hence, the (k, k') NN rule, if applied, would assign X to C_2. However, it is clear in this case that a better decision would be to consider X as a new class type. To cope with situations when the set of classes is not exhaustive, the following modification to the NN rule is used:

(1) Set a boundary threshold D_k on each class k, k = 1,2, ..., m

(2) Compute $d(X, X^j)$ for j = 1,2, ..., NT

(3) If $d(X, X^j) > D_k$ where $X^j \varepsilon C_k$, do not count X^j as a candidate for the set of k nearest neighbors. Otherwise record X^j.

(4) If k nearest neighbors cannot be found, reject X

(5) If k nearest neighbors can be found but $N_{c^*} \leq k'$, reject X

(6) If k nearest neighbors can be found and $N_{c^*} > k'$, assign X to c*

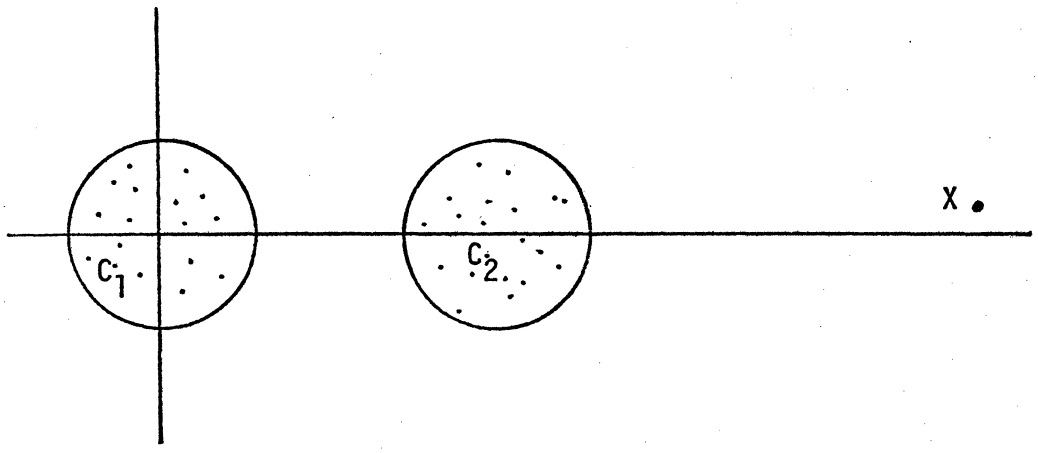


FIGURE 2-2. TWO DIMENSIONAL CASE

The method for setting the boundaries D_k should be determined ad hoc based on the knowledge of the class structures and the misclassification tolerance level. An adequate Monte-Carlo procedure works as follows:

(1) For each pair of training pattern X^j, X^t in C_k compute $d(X^t, X^j)$

(2) Determine the value D_k for which β percent (e.g., $\beta = 95\%$) of the pairs satisfy $d(X^t, X^j) \leq D_k$

The principles of this procedure are closely related to the principles of complete-link and graph-theoretic clustering methods. Similar procedures can be developed following the single-link or k-means principles. Presently, the (k, k′) NN rule is used with the modification just described. This method is available to the user with various options for distance functions and the option to apply the edited nearest neighbor preprocessing. Patterns rejected by this (k, k′) NN method are submitted to Level III.

2.3 Classification By Clustering (Level III)

2.3.1 The Clustering Algorithm. The classification method used in Level III is clustering-oriented. It is assumed that the training patterns have passed an imaginary clustering process which has generated M clusters corresponding to the M known classes and assigned each of the training patterns to the cluster it belongs to. When a new pattern arrives, the clustering algorithm is "reactivated" and decides whether to assign this pattern to one of the old classes or to classify it as a new type. Many clustering algorithms can be found appropriate. The algorithm used in the current system is described below. It is essentially the same proposed by MacQueen (1967) and Sebestyen and Edie (1966).

Notation

d - distance function

$K(i)$ - number of clusters at stage i

For cluster k at stage i denote:

$n_k(i)$ - number of members

$C_k(i)$ - centroid of the cluster (average of all the members)

$\Sigma_k(i)$ - covariance matrix

$A_k(i)$ - membership threshold

$$A_k(i) = f_A \cdot \max_j \{d(X^j, C_k(i)) \mid X^j \text{ in cluster } k\}$$

$B_k(i)$ - non-membership threshold

$$B_k(i) = f_B \cdot A_k(i) \qquad f_B > 1$$

f_A, f_B - predetermined factors

Initialization

Let X^1 be the only member of cluster 1

$C_1(1) = X^1$

$\Sigma_1(1)$ = predetermined covariance matrix or the identity matrix

$n_1(1) = 1$

$K(1) = 1$

Step i

(1) For k = 1 to K compute

$$d\,[X^i, C_k(i)] = [X^i - C_k(i)]^T \; \Sigma_k^{-1}\,(i)\,[X^i - C_k(i)]$$

(2) Find

$$d\,[X^i, C_\ell(i)] = \min_k \{d\,[X^i, C_k\,(i)]\}$$

(3) If $A_\ell\,(i) < d\,[X^i, C_\ell(i)] < B$ store X^i for later processing

(4) If $d\,[X^i, C_\ell(i)] \leq A_\ell$ assign X^i to cluster ℓ
and update:

$$n_\ell(i+1) = n_\ell(i) + 1$$

$$C_\ell(i+1) = [(n-1)C_\ell(i) + X^i]/n$$

$$E = 1/n\,[(n-2)\,\Sigma_\ell(i) + (n-1)\,C_\ell(i)\,C_\ell^T(i) + X^i X^{iT}]$$

$$\Sigma_\ell(i+1) = n/n-1\,[E - C_\ell(i)\,C_\ell^T\,(i)]$$

(C_ℓ and Σ_ℓ are unbiased extimators for the true mean and covariance, respectively)

(5) If $B_{\ell}(i) \leq d\,[X^i, C_{\ell}(i)]$, use X^i to establish a new cluster, say cluster N, and update:

$$K(i+1) = K(i) + 1$$

$$C_N\,(i+1) = X^i$$

$$\Sigma_N\,(i+1) = \frac{1}{K(i)}\,[\Sigma_1 + \ldots + \Sigma_{K(i)}]$$

$$A_N\,(i+1) = \frac{1}{K(i)}\,[A_1 + \ldots + A_{K(i)}]$$

$$B_N\,(i+1) = f_B \cdot A_N$$

$$i = i+1$$

The clusters of the training patterns are used to determine the corresponding centroids, covariance matrices, and A_k, B_k thresholds. A new coming pattern X is assigned to class C_k if k is the nearest cluster and $d(X, C_k) \leq A_k$. If $d(X,C_K) \geq B_k$ then X constitutes the nucleus for a new class. If $A_k \leq d(X,C_k) \leq B_k$, classification of X is deferred for a later stage. In between stages more observations on X are taken, and the learning process is activated (Section 6.4).

Once a new class is established, its classification as a threat or non-threat is determined as follows:

(1) Compute distance from the new class centroid to the nearest threat class centroid, d_1, and to the nearest non-threat class centroid, d_2.

(2) Compute d_1/d_2.

(3) If $d_1/d_2 < R_2/R_1$, classify as threat

$d_1/d_2 \geq R_2/R_1$, classify as non-threat

where:

R_1 - the risk involved in classifying as threat when in fact it is a non-threat.

R_2 - the risk involved in classifying as non-threat when in fact it is a threat.

2.3.2 Learning in Real Time. For a given object, the data which is accumulated between t_K and t_{k+1} for k = 1,2, ..., E-1 is used for updating the pattern of this object. Let $X = (x_1, x_2, \ldots, x_n)$ denote the pattern at time t_k. Then for each observation during the period (t_k, t_{k+1}), the following transformation takes place:

$$x_j \rightarrow [(S_j-1)\, x_j + x_j^1]/S_j$$

where S_j denotes the total number of observations on feature x_j including the last one, and x_j^1 is the last value observed. Note that at a given point of time, the value of S_j may be different for different j's. It represents the number of times the x_j sensor(s) were activated for this particular object. In the present system, S_j is assumed the same for every j and every object.

Since at time t_E, a classification decision has to be made, the gap between A_k and B_k for k = 1,2, ..., m needs to be closed by time t_E.

The magnitude G_t by which A_k is increased and B_k is decreased at time t is determined by:

$$G_t = \frac{A_k - B_k}{2} [t/t_E]^{\alpha} \quad \alpha>0 \quad t = t_1, t_2, \ldots, t_E \quad .$$

For every $\alpha>0$, G_t is an increasing function of t, which means that larger pieces of the initial gap are cut as time progresses. For $0<\alpha<1$, G_t is concave while for $\alpha>1$, G_t is convex. This means that for $0<\alpha<1$, the rate of increase in G_t is more moderate than for $\alpha>1$. See Figure 2-3 for illustration.

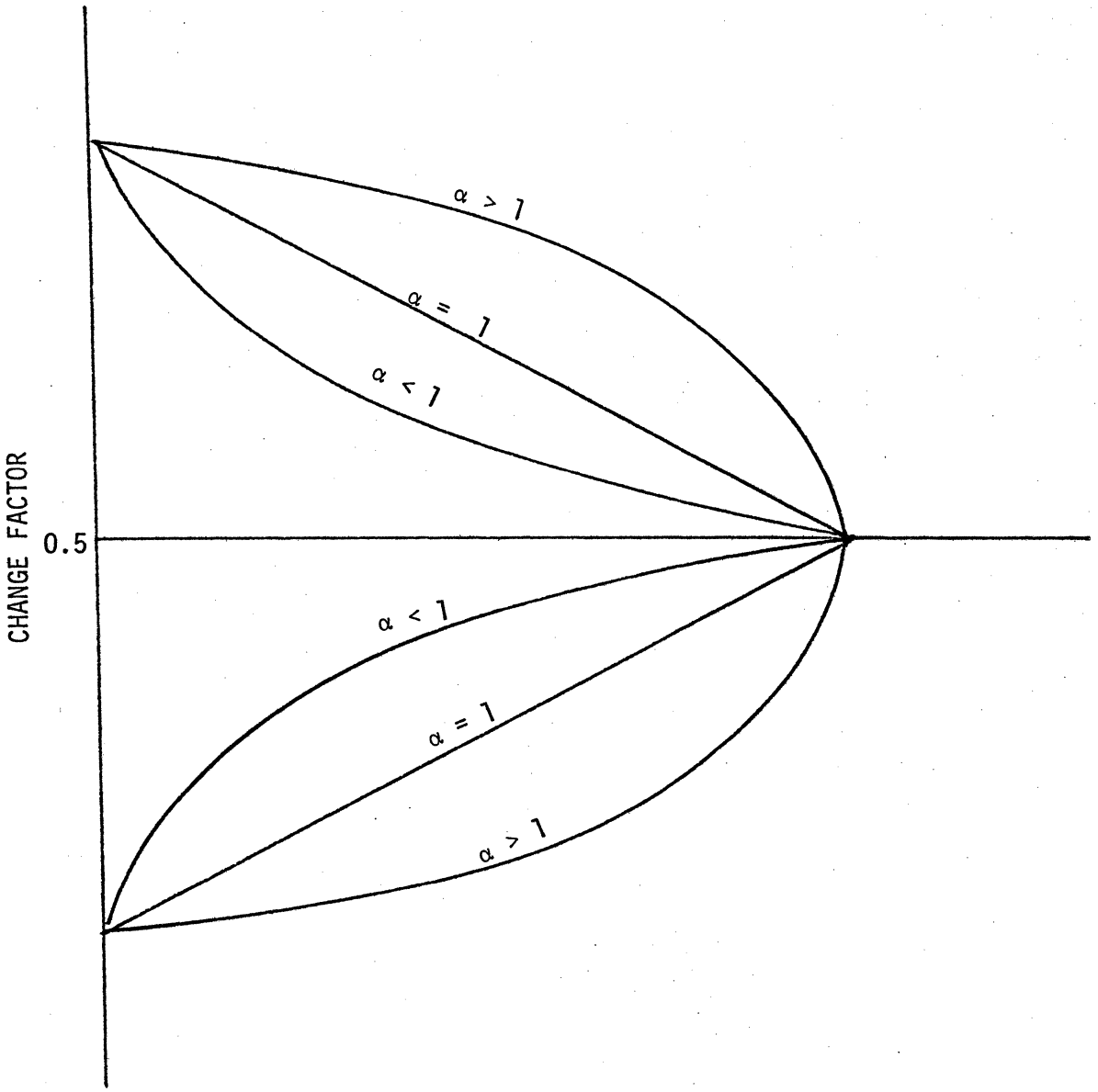


FIGURE 2-3. CONVERGENCE OF A_k and B_k

3. EVALUATION

3.1 Overview

The evaluation of the algorithm described in the previous sections was performed in six major phases. In each of the first three phases, one level of the algorithm was independently tested, while in the fourth phase the "learning in real time" module was evaluated. The purpose of those four phases was to test the validity and sensitivity of individual portions of the system and to decide upon the specific parameters to be used. For instance, the evaluation of Level II included experiments to determine the training set size, the distance function, the k,k′ values, the effect of editing the training set, and the class boundary threshold. In the fifth phase, the performance of our algorithm is compared with the performance achievable with an existing RV discrimination system called "ODE".

For each phase, training and testing patterns were randomly generated by a simulation process. The randomness of these patterns was controlled to serve the special purposes of the particular experiment. For instance, in the first phase, the random process was directed to generate many patterns which can be classified by using deterministic rules. These patterns were of little use for the second and third phases because, if the deterministic rules are successful, these patterns would not reach later stages.

3.2 Approach

The design of the evaluation plan is organized in two parts. The first part specifies the issues for examination, while the second part describes the actual experiments to be performed in order to test these issues. In general, each experiment was designed to test more than one issue. On the other hand, there are issues which are tested by more than one experiment. The design of the evaluation plan concludes with a cross reference table of issues to be tested against experiments to be performed.

By this approach we first determine "where we want to go", and next, "how to get there". It enables experiments to be designed in such a way that the maximum information will be extracted. The list of issues tested and their corresponding experiments are described in the TRW report (CDRL Item A005, 17 January 1977).

4. REFERENCES

Bentley, J.L. A Survey of Techniques for Fixed Radius Near Neighbor Searching. SLAC Laboratory, Stanford University (Stanford, CA). SLAC Report No. 186, 1975.

Cover, T.M. and Hart, P.E. Nearest Neighbor Pattern Classification. IEEE Transactions on Information Theory, 1967, IT-13: 21-27.

Devijver, P.A. Error and Reject Tradeoff for Nearest Neighbors Decision Rules. MBLE Research Laboratories (Brussels, Belgium). Report R322, 1976.

Duda, R.O. and Hart, P.E. Pattern Classification and Scene Analysis. New York: Wiley and Sons, 1973.

Dudani, S.A. The Distance Weighted k-Nearest Neighbor Rule. IEEE Transactions on Systems, Man, and Cybernetics, 1976, SMC-6: 325-327.

Friedman, J.H., Bentley, J.L., Shustek, L.J. An Algorithm for Finding Nearest Neighbors. In Press, IEEE Transactions on Computers, 1977.

Fukunaga, K. and Narenda, P.M. A Branch and Bound Algorithm for Computing k-Nearest Nieghbors. IEEE Transactions on Computers, 1975, C-24: 750-753.

Hellman, M. The Nearest Neighbor Classification Rule with a Rejection Option. IEEE Transactions on Systems, Science, and Cybernetics, 1970, SSC-6: 179-185.

Kanal, L.N. Patterns in Pattern Recognition: 1968-1974. IEEE Transactions on Information Theory, 1974, IT-20: 697-722.

MacQueen, J. Some Methods for Classification and Analysis of Multivariate Observations. Proc. of the 5th Berkeley Symposium on Mathematical Statistics, 1967, Vol. 1: 218-297.

Patrick, E.A. Fundamentals of Pattern Recognition. Englewood Cliffs, New Jersey: Prentice Hall, 1972.

Pendro, C.S. and Wagner, T.J. Another Look at the Edited Nearest Neighbor Rule. IEEE Transactions on Systems, Man, and Cybernetics, 1977, SMC-7: 92-94.

Sebestyen, G.S. and Edie, J. An Algorithm for Non-Parametric Pattern Recognition. IEEE Transactions on Electronic Computers, 1966, EC-15: 908-915.

Tomek, I. An Experiment with the Edited Nearest-Neighbor Rule. IEEE Transactions on Systems, Man, and Cybernetics, 1976, SMC-6: 448-452.

Wagner, T.J. Convergence of the Nearest Neighbor Rule. IEEE Transactions on Information Theory, 1971, IT-17: 566-571.

Wagner, T.J. Convergence of the Edited Nearest Neighbor. IEEE Transactions on Information Theory, 1973, IT-19: 696-697.

Wilson, D.L. Asymptotic Properties of Nearest Neighbor Rules Using Edited Data. IEEE Transactions on Systems, Man, and Cybernetics, 1972, SMC-2: 408-421.

Yunck, T.P. A Technique to Identify Nearest Neighbors. IEEE Transactions on Systems, Man, and Cybernetics, 1976, SMC-6: 678-681.