AMOEBA Clustering Revisited

Jack Bryant
Center for Approximation Theory, Department of Mathematics, Texas A&M University, College Station, TX 77843-3368

ABSTRACT: A description of the clustering, classification, and image display program AMOEBA is presented. Using a difficult high resolution aircraft-acquired MSS image, the steps the program takes in forming clusters are traced. A number of new features are described here for the first time. Usage of the program is discussed. The theoretical foundation (the underlying mathematical model) is briefly presented. The program can handle images of any size and dimensionality.

INTRODUCTION

IN THE BROADEST SENSE, *cluster analysis* finds regularity or structure in data and *classification* assigns labels. An image analyst performs these (and many other) tasks in the process of understanding an image. The analyst clusters by observing that different parts of the image represent very similar entities from the real world. Labeling is often the primary objective, or at least an essential first step. To an analyst, the data units are spatial associations of image elements (pixels); the size and shape of the associations not only vary but probably influence the classification process itself. It would seem quixotic for an analyst to even attempt to understand an image from a huge list of the measurement vectors.

Present-day computers, on the other hand, are best at dealing with much smaller and more regular chunks of an image. Because the pixels in a multispectral image are vectors, a natural approach to automatic image analysis is the use of general-purpose multivariate statistical techniques, including clustering algorithms (see, e.g., Anderberg, 1973; Duran and Odell, 1974; Bryant, 1978; Jain and Dubes, 1988). This yields a useful first cut at clustering which can often be improved by interactive manipulation. However, the cluster map is unsatisfactory for low resolution (e.g., Landsat) imagery: many isolated misclassified pixels are seen which must be dealt with one at a time. We will examine the reason for this below: briefly, it is the unavoidable consequence of processing the image a pixel at a time. Some (often many) pixels do not fit the statistical model underlying per-pixel classification.

A multi-dimensional image, of course, is not arbitrary multivariate data. The spatial structure of the image is often the most significant attribute, without which an analyst would be hopelessly lost. Yet it is difficult to incorporate spatial information in image processing programs. The purpose of this note is to describe Version 13 of AMOEBA, a clustering, classification, and image display program which makes essential and extensive use of spatial structure.

I begin with a literature review. Then I outline the mathematical model. A general description of the program follows: many details are given here for the first time. Images are included to illustrate each major step; one should probably skip the details and just look at the pictures on first reading. I then explain usage and and give execution characteristics of AMOEBA. Finally, I suggest directions future work in the general area surrounding AMOEBA might take.

LITERATURE ON AMOEBA

Version 6 has been described (Bryant, 1978; Bryant, 1979), and descriptions of two features of the current version have recently appeared (Bryant, 1988; Bryant, 1989). The first tests of the program to be published (except in reports to the NASA-JSC) are by Jenson *et al.*, (1982). They find the program (Version 8) to be faster than and as accurate as ISOCLAS (a variant of ISODATA developed by the NASA). Gurney and Townshend (1983), in a survey on the use of contextual information in classification, mention the improvement which results from the use of spatial information by AMOEBA. Hill and Kelly (1986) find about 80 percent classification accuracy and 95 percent mapping accuracy; the areas studied were so large and separated that supervised classification could not be done. Hill and Kelly (1987) used a preliminary classification to form a mask to get a subdivision of the original clusters. (They were using Version 10; this operation has been made much easier in Version 13.) See also Kelly and Hill (1987). Strickland and Schowengerdt (1988) use Version 12 in the course of automatic inspection of printed circuit boards; they also employ masking.

The literature on AMOEBA has its darker side. The basic paper on AMOEBA is referenced in Haralick and Shapiro (1985) but not mentioned in the body of the survey. Several authors (Chittinei, 1983; Strahler *et al.*, 1986; Trivedi and Bezdek, 1986; Woodcock and Strahler, 1987; and Yarman-Vural and Ataman, 1987) mention AMOEBA, but, from what they write, I think they misunderstand the basic ideas. Perhaps this is my fault—the details are hard to understand without lengthy explanation, and that has never been done.

While it might be fashionable nowadays for one to be misunderstood, this should not extend to a program with so many capabilities. An overview of the program, including the more developed model, the new context classifier, and a discussion of program operation, has never been published. I hope the concise presentation here will expose this public domain program to more potential users.

THE MATHEMATICAL MODEL

Before I begin, I need some notation. Suppose the multi-image has rectangular organization with pixel coordinate pairs of integers. Let us use the notation p for the spatial location of a pixel and $\mathbf{p} = [p_1\ p_2\ \ldots\ p_m]^T$ for the measurement vector at location p. Denote by $|p - q|$ the spatial distance between p and q and $\|\mathbf{p} - \mathbf{q}\|$ the euclidean distance $[\Sigma\ (p_k - q_k)^2]^{1/2}$ between the measurement vectors. (In the model, the measurements are vectors with real components. In the program, the components are 16 bit two's complement integers. The original measurements are presently eight bit unsigned integers.)

My first assumption is simple: I assume that real classes exist. A pixel is said to be *pure* if it and all four nearest neighbors are in the same real class. It is clear that if p and q are pure pixels and $|p - q| \leq 2$, then p and q are in the same real class. A *path* is a sequence $p_1,\ldots,p_n$ such that p_i is one of the four nearest neighbors of p_{i+1}, $i = 1,\ldots,n - 1$. A set of pixels is said to be *connected* if each pair of points in the set can be joined by a path lying in the set. Denote by P the set of all pure pixels. The *components* of P are the maximal connected subsets of P; they are called *patches*. Each pure pixel is contained in a uniquely determined patch. The *classifier* is the (unknown) mapping from

PHOTOGRAMMETRIC ENGINEERING AND REMOTE SENSING,
Vol. 56, No. 1, January 1990, pp. 41–47.

0099-1112/90/5601–47$02.25/0

the set of all pixels to the (again unknown) set of labels. If F is a patch, then points in F are classified alike. Under the assumption that the patches can be estimated accurately, this observation furnishes an automatic sampling of spectrally different points which should bear the same label.

A nearest neighbor classifier (NNC) is a spectral classifier that classifies a measurement space vector into one of c classes. Let $d(\mathbf{p},\mathbf{q})$ denote the distance between $\mathbf{p}$ and $\mathbf{q}$. The *attractors* are vectors $\mathbf{q}_1 \ldots \mathbf{q}_c$, and $\mathbf{p}$ is assigned to class m if $d(\mathbf{p},\mathbf{q}_m) \leq d(\mathbf{p},\mathbf{q}_i)$ for $i = 1,\ldots,c$. (Ties can be decided by selecting the smallest value of m). An NCC is uniquely determined by the distance function and the set of c cluster attractors. The next assumption has the remarkable effect of telling what the distance function should be for the lowest level classification decisions. A measurement vector $\mathbf{r}$ is said to be a *convex combination* of vectors $\mathbf{p}$ and $\mathbf{q}$ if $\mathbf{r} = \alpha\mathbf{p} + (1 - \alpha)\mathbf{q}$ for some α between 0 and 1. I assume that if p and q are pure pixels in the same real class and if $\mathbf{r}$ is a convex combination of $\mathbf{p}$ and $\mathbf{q}$, then an NCC should classify $\mathbf{r}$ in the same class as p and q.

A distance function is said to be *natural* if it is induced by a norm on the space R^m in which the measurement vectors are embedded. For example, the l_1 distance $\Sigma|p_k - q_k|$ is derived from the norm $\|\mathbf{p}\| = \Sigma|p_k|$. This distance function is used in ISOCLAS. Another popular distance function is implicit in the maximum likelihood classifier (see, *e.g.*, Andrews, 1972). The assumption implies that the decision regions should be convex, and this leads to the following: any natural NCC is weighted Euclidean distance $[\Sigma w_k (p_k - q_k)^1]^{1/2}$. Because clustering is applied in the preliminary exploratory phase of analysis, it is natural to weight the measurements equally. Therefore, the distance function for the NCC should be Euclidean distance. This is probably even true in problems of supervised classification. See Bauer *et al.* (1977) and Kast and Davis (1977).

Let $C{:}P \rightarrow \{1,\ldots,c\}$ be a classifier defined on the set P of pure pixels. Consider the classification of pure pixels p and q: there are four possibilities:

(1) p and q are in the same real class and $C(p) = C(q)$.
(2) p and q are in different real classes and $C(p) \neq C(q)$.
(3) p and q are in the same real class and $C(p) \neq C(q)$.
(4) p and q are in different real classes and $C(p) = C(q)$.

Clearly Cases 3 and 4 are errors. The pair probability of misclassification is the sum PPMC of the conditional probability of Case 3 and the conditional probability of Case 4. (Although similar to the Rand statistic, the PPMC is much different when there are many clusters because there are many more Case 4 events than Case 3. See Rand (1971).) With the distance function for pure pixel classification fixed, it remains to select the cluster attractors – the vectors which the NNC uses. I assume the attractors should be selected to minimize the PPMC.

While NNC classification is anticipated for pure pixels, it is certainly not suitable for all pixels. After a preliminary per-pixel nearest neighbor classification of the image, let a denote the number of the eight nearest neighbors of p which have the same classification as p. I assume that the classification of p is acceptable if and only if $a \geq 2$. Unacceptably classified pixels will be reclassified. To see how this should proceed, consider a pixel p on a spatial boundary between two pure pixel regions. In this simple example, each of the measurements p_i will be a convex combination of the pixel measurements nearby. (Because of registration errors the α_i may be different.) Such a vector is completely unsuited for nearest neighbor classification because a third attractor, unrelated to the nearby pure pixel mesurements, may be closer to the measurement space mixture. More elaborate modeling with several classes lends support to the assumption that an unacceptably classified pixel should be reclassified in the class of one of the acceptably classified eight nearest neighbors, provided this is possible. Note that the mixture, in the absence of registration errors, would be closer to the nearest class than half the distance from that class to the others of which it is a mixture. With registration errors, this should be increased by a factor of $\sqrt{2}$, but pure pixels are not as bothered by registration problems. I assume that no pixel should be classified by the per pixel nearest neighbor classifier if the distance to the attractor exceeds this threshold.

THE PROGRAM

AMOEBA now consists of about 3,600 lines of FORTRAN77 code (12,000 lines including comments). I maintain the VAX* version (the only one I can supply), although various versions of the program have been converted to such diverse systems as the IBM 3090, the HP 3000, the IBM PC, and the MIPS 2000. A brief description of the VAX version follows. To illustrate the complex steps being described, I will use a 512- by 512-pixel originally 11-band aircraft image. Figure 1 shows the first band of the three-band product which is intended to be displayed as red. The technique of Bryant (1988) was used. Although significant information from the original is missing in this product, one can see in a general way the relationships I want to illustrate.

THE MAIN PROGRAM

The program is modular; system dependent features have been isolated and are confined to the main program and the routines which read and write files. Care is taken to minimize system resources taken by the program. This, together with file management, is the responsibility of the main program. Because of the many options and the dependence on the size of the image being analyzed, memory requirements vary from small to fairly large. The main benefit of this relatively minor effort is that the program is easy on the system; this, in turn, can result in better system performance. The program can handle images of any size and dimensionality, and all of this is transparent to the user.

FIG. 1. Band one of the test image.

* The mention of tradenames here is in no way an endorsement of products or manufacturers.

First Patches

On the first pass through the image, points which are possibly not pure pixels are identified. Two techniques are provided: one uses a multidimensional version of the quickly computed Robert's gradient of distance 2 (see Haralick and Dinstein (1975)). It is suitable for high to medium resolution low noise data. When the gradient exceeds a threshold, boundary is decided. The other method operates by comparing the multivariate measurement differences of spatially adjacent pixels to a threshold vector. In each case the threshold adapts to furnish a user-supplied number of pure decisions (default 45 percent thus, more decisions are boundary than pure in the default selection). An example of the pure pixels discovered can be seen in Figure 2. Note the large oval shaped bright field in Figure 1 which is not found.

Sampling Patches

The next task is to sample the patches found. After sampling, the measurements loose explicit spatial identity, but actually preserve spatial structure implicitly because they are separated samples from the same patch. There are two conflicting problems: because there are a huge number of patch points in a large image, the sample should be sparce. On the other hand, one must not lose a sample from a rare real class. Initially the patches are scanned and sets of five points, called test sets, as spatially separated as possible, are selected from each patch. The reason for selecting five pixels from each patch will be explained later. The first 25 test sets encountered which have the last pixel of the test set different from the first pixel in that set are saved in a file. Because the initial patch estimate was very conservative, I am confident these test sets are samples of real classes.

For the remaining patches, let d denote the spectral distance from the first pixel to the last pixel of the potential test set, and let a denote the running average of this distance over test sets found acceptable so far. A new test set is added if it is interesting, believable, and new. To be interesting, $d \geq a/2$; to be believable, $d \leq 4a$; and to be new, the first pixel in the test set must not be closer to the first member of one of the last 25 collected than d. Note that a probably gradually increases.

Many test sets remain, however. To form the initial tentative cluster attractors, use the third test pixel of each test set. When the number of attractors exceeds 156, classify the first and last of each of the test pixel sets collected so far. Count the number of classifications assigned to each attractor by test set and by attractor (forming the *weight* of the attractor). If two distinct attractors get assigned to the pair from a test set, and if the weight of each is exactly 1, then combine the two attractors, assigning the new average attractor a weight of 2. Otherwise, if the weight of one is 1, eliminate it and increment the weight of the other. Next, eliminate any attractor with weight of zero (these are duplicates). If over 100 attractors remain, classify the second and fourth test pixel; until fewer than 101 attractors, eliminate any set with weight of one, then two, and finally three. These steps are order dependent, so the means are mildly shuffled after each pass.

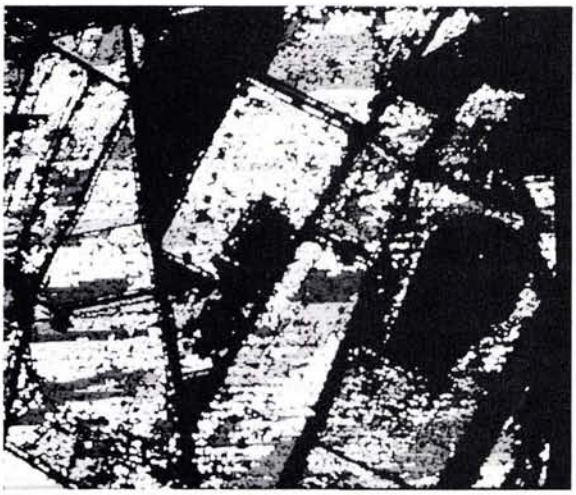

Fig. 2. Patches.

Eventually, it is likely that more than 356 test sets will be collected. Classify each test set (by classifying the third member) and count the number of times an attractor is assigned a test set. If there are 100 attractors and 300 test sets, then, on the average, each attractor will receive three classifications. Test sets assigned to those attractors with the most test sets attracted are duplicates, and are simply removed until 300 test sets remain. (The most over-represented are thinned first.) If over 100 attractors are present, invoke the attractor thinning procedure just described.

The numbers 156, 100, and 356 are not entirely arbitrary; they were selected to allow the data storage to fit in one 32K word memory segment. In that sense they are inherited from Version 8 (the HP 3000 version). On the other hand, any clustering process must be highly emperical, even one based on a model. I have experimented with increasing the numbers. No difference in the final clustering is noticed, and more computer time is taken. With substantially smaller numbers a small cluster is sometimes lost at this stage, although the initial classification logic will probably restore it.

Cluster Formation

The test sets are used directly to estimate error Case 3 above. Because there are ten distinct pairs in a test set of five sample pixels, this gives ten tests per test set. There are ten unordered distinct pairs in a set containing five elements; thus, we compare the classification of the first with the second, the third,..., and finally the classification of the fourth with the fifth. To estimate Case 4, use the initial cluster attractors to find the first principal component of the set of distinct pair differences of attractors. Earlier versions used the sum of the measurements; I have found the principal component mapping to be superior when thermal or far IR bands are present. It is quickly computed. This first principal component defines a one-dimensional attribute; transform the test sets based on this linear mapping and arrange them in increasing order. Let there be t test sets. Using synthesized data, I determined experimentally that pairs with index differing by $s = 0.075t$ were in general close samples from different real classes, and thus would be useful as sharp tests of the different real class-same cluster error case. For each test set i, use the classification of its center and the ten from the two test sets with index $i \pm s$. (If one of these is not a valid index, replace it with $i \pm 2s$; one of these will serve.)

Each of the two error case tests (which are equal in number and therefore give estimates on the PPMC without unequal weighting) counts an error when an attractor is involved in one of the two error events. Until the number of attractors is reduced to 32, eliminate the attractor contributing the most errors and reassign test pixels which were assigned to the attractor. (The number of bits in the VAX data bus is 32; thus, I can encode a subset of the original 32 attractors as a single computer word for quick evaluation.)

When 32 attractors are attained, the search is widened: essentially *all* of the arrangements of the $2^{32} - 1$ possible subsets which have any chance at all of having the minimal PPMC are evaluated. The number of arrangements being considered at a time, called the *width* of the search, is 45 in the present version. The number of times a daughter arrangement has her daughters

evaluated, called the *depth* of the search, is presently three. A variant of the alpha-beta algorithm (originated by Claude Shannon) is used to prune searches which are not productive. Tests on real and simulated data show precisely the same solution is obtained with this method as with the search with a width of 32,768 and a depth of five (involving over 21 million evaluations). The method is fast because of two factors: wasteful duplicate evaluations are quickly avoided by saving in a 32 bit word the arrangement of attractors in an already evaluated set; and because each step requires few classifications to update the PPMC calculation for a subset which differs by the absence of exactly one attractor.

An option of the program allows one to produce a three-band color display from the greater than three-band multi-image; this has recently been described (Bryant, 1988) in some detail. Color reproductions of the three-band composite image and the cluster map can be found there. The computations of cluster formation just described are interrupted when 32 attractors have been formed, and the dimensionality reduction process and image formation is started. Then the entire program is restarted using the three-band image instead of the original.

Initial Classification

Now that the cluster attractors have been formed, the rejection thresholds can be computed. For each attractor **a**, find the most distant other attractor (say **b**), and let $p_a = \|\mathbf{a} - \mathbf{b}\|/2$ be the rejection threshold for the cluster A attracted by **a**. During the initial per pixel classification step, only classes A are considered for classification of **p** which have $\|\mathbf{p} - \mathbf{a}\| < p_a$. In particular, there is a possibility that no class exists; in that case the offending pixel is labeled with 0 in the classification process.

I now describe the per-pixel classification process. Recall that each pixel is a vector in R^m; with c classes, it at first appears to require $3m$ arithmetic operations to compute the distance from the pixel to an attractor, and so $3cm$ such computations to classify a pixel. However, the image is being processed in a spatially natural order so that most of the time the next pixel encountered is like the one just processed. By using this fact to exit the distance function calculation loop early (and several similar programming tricks—see Bryant (1989)), I have made the per-pixel classifier in Version 13 about three times faster than earlier versions.

Improved Boundaries

In noisy images false boundaries are detected. An odd and, perhaps, hard to comprehend phenomenon is that even the best local boundary detectors fail to find boundaries in high resolution images. High resolution requires a large window to find boundaries. Woodcock and Strahler (1987) and others have found that for each analysis problem there is a range of acceptable resolution, and that too fine is as detremental as too coarse. After a per-pixel classification, however, a point which fails to be acceptably classified is very likely to be a boundary point, independently of noise and resolution. Turn this around: a point which is classified like all four of its nearest neighbors should be a pure pixel in the setting of the model; accordingly, under the assumption that the original set of attractors leads to a good classification, a new boundary map is produced. The new map marks points which are not classified like all four of their neighbors as boundary. The zero (rejected) classification is included as a legitimate class so that a blob of rejected pixels will have another chance. (This is very important in high resolution data — I have examples of aircraft MSS data taken of oil fields where the drilling sites were not found by the program without this feature.) Then the entire process is restarted with new boundaries. That is, the program is iterative, although two iterations suffice to arrive at an acceptable clustering. In Figure 3, I display the boundaries which result from this strategy.

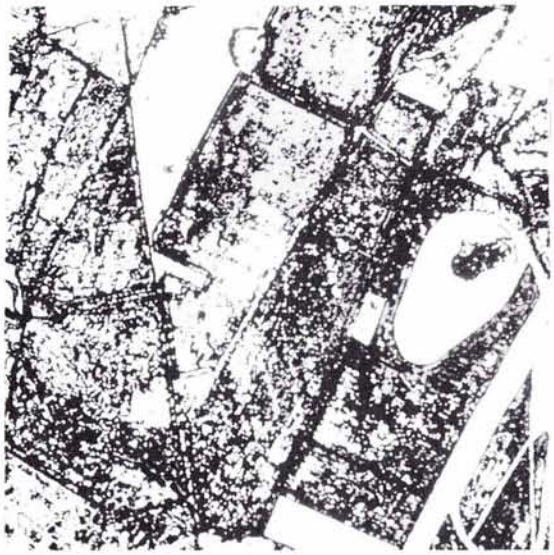

FIG. 3. Classification-based estimate of the boundary.

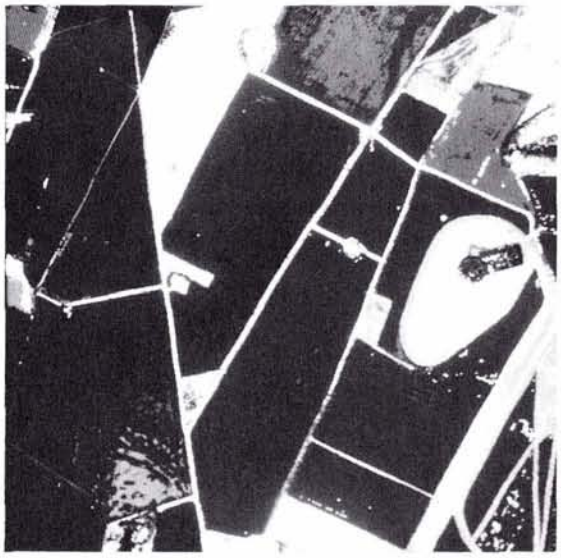

FIG. 4. Per-pixel classification.

Once the cluster attractors are known, it is time to classify. Three classifiers are provided. The per-pixel classifier, modified by the rejection thresholds discussed above, is also used as a preliminary classifier for the other two; see Figure 4. If per-pixel classification is to be the end product, then unclassified points result in an expanded list of cluster attractors, used only to classify other unclassified points (so that the exceptional classes are small but often significant).

The second classifier, with results displayed in Figure 5, carries out the reclassification process suggested by the model, except that the user can specify the number of eight nearest neighbors which should be classified like the central point for accepting a per-pixel classification. To allow for misregistration, the rejection thresholds are multiplied by $\sqrt{2}$. Table 1 shows the statistical summary of the final classification: listed is the index (i.e., cluster label), the number classified in that cluster, and the cluster attractor. There are three bands in this example because a dimensionality reduction from the original 11 bands to three was selected. (The band order is RGB, attempting to imitate color

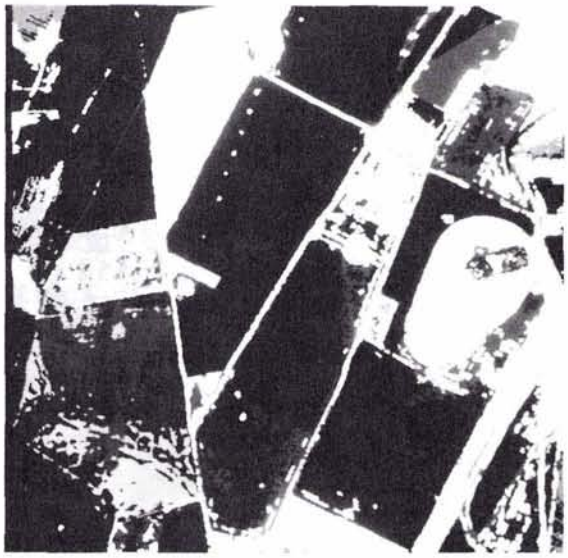

FIG. 5. Spatial improvement of per-pixel classification.

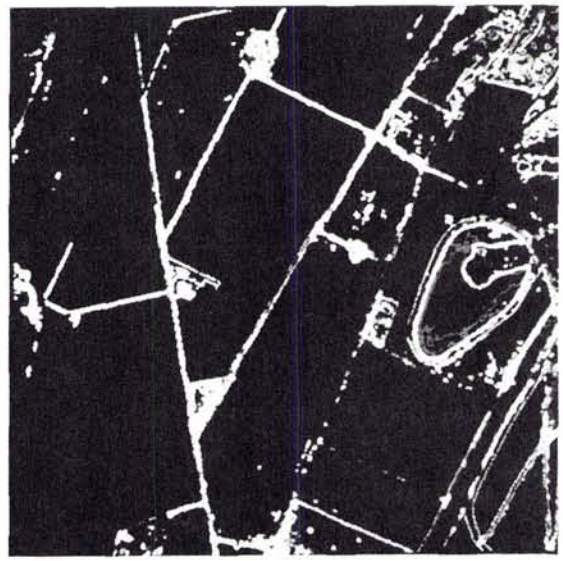

FIG. 6. Context classification.

TABLE 1. CLUSTERS FOUND BY AMOEBA USING ALL DEFAULT PROGRAM PARAMETERS.

Index	Number	Attractor		
1	144,886	65	36	28
2	32,516	81	88	36
3	2,568	54	80	87
4	12,419	218	44	28
5	7,832	103	93	95
6	210	38	253	29
7	6,425	103	132	88
8	12,230	113	110	83
9	26,220	198	49	89
10	6,475	238	43	63
11	1,945	145	127	84
12	3,583	99	177	87
13	1,827	127	136	108
14	1,159	164	101	120
15	390	180	247	150
16	1,468	152	212	247

infrared (IR) film products, so that cluster 1 seems to be vegetation.)

The third classifier is intended as an tool for an analyst who needs to know the classification of the points in the 3 by 3 neighborhood of the point as an image. I display such a product in Figure 6; the solid looking dark areas are uncomplicated homogeneously classified regions. The lacy pattern surrounding them is impossible to interpret without an interactive display system, but contains information hard to obtain from an ordinary classification map. In the statistical summary furnished with the image, one obtains the usual count and attractor of the attractors for the homogeneous areas. (These have low class numbers.) Following that is a list of the class numbers of the mixed classifications: included is the index, the size of the class, the index of and count of the most popular pure class present, the same for the second most popular class, and whether four or more classes were present in the 3 by 3 neighborhood. From the output map and statistics file, it is easy to implement your own spatial classifier, e.g., a majority rule classifier. I have experimented with 9 by 9 majority rule classifiers with limited success. (They work best on extremely high resolution data.) This program option is intended as a preprocessor for a very fast 3″ by 3″ context classifier which has been designed but not implemented. See also Wharton (1982).

The context classifier operates by packing the arrangement of classifications in the 3 by 3 neighborhood in a 16-bit word. These are histogrammed, with the most popular being used as "leaders" in a modification of the leader algorithm. This method takes the first data unit encountered as the leader and joins subsequent data to it when the distance is less than a threshold; otherwise, it adds new leaders. The modification simply starts with p pure contexts and uses the $255 - p$ mixed contexts as leaders. The distance between contexts is larger when the contexts are very different. For example, consider three classes X, Y, and Z. Suppose the context in four 3 by 3 neighborhoods is as follows:

A: 7 X and 2 Y B: 4 X and 5 Y
C: 6 X and 3 Z D: 7 X and 2 Z

The distance from A to B and C is 6, and to D is 4. The distance from B to C and D is 10. Finally, the distance between C and D is 2. (The distance function is defined when more than two classes are present but it is more complicated). The program does not consider contexts separated by more than 8 to be candidates for identification; thus, B and C would never be merged. The program ordinarily yields the pure classes plus context classes giving 255 total; an option merges context classes as much as possible (yielding as few context classes as possible), again following the logic of the leader algorithm.

MASKING UNWANTED CLASSES

It sometimes happens that one class dominates the image. Indeed, the example used here illustrates this phenomenon: the large class covers about 55 percent of the image. The class looks like pasture, but suppose you need more: imagine the soil moisture or ground cover type could be detected from your image, and you would like to investigate the sub-structure of some of the preliminary clusters. The program provides an option whereby you can select certain clusters to be combined and then reclustered, treating the others as a mask. In Figure 7, I show the result of masking clusters 2 to 16. That is, the program tries to find finer structure there in the part of the image formerly placed in cluster 1. (The program allows any selection or combination of classes for finer clustering, including context classification classes.) The new map adds clusters to the old ones without changing their clustering; however, the numbering

FIG. 7. Detailed classification of the pasture class only.

of the new clusters will be different. (Clusters labels are always assigned in increasing order by sum of the bands in the attractor.)

RUNNING AMOEBA

Operation of the program is simple if the user selects default values for the optional parameters. The mandatory parameters are required to specify the input filename, its organization, its size (unless this is read from the header), and the output file name. The VAX version assumes International Imaging System header structure as default, but the user can supply all needed information at run time simply by telling the program the file is not standard and supplying required information. In addition, the four short modules which use header information or perform image input/output can be (and have been) modified to use other systems.

There are nine optional parameters not related to file structure:

Three bands? A Boolean parameter: if the original image is at least four bands, then a three-band composite image will be produced. You will have to furnish two band numbers, one the index of a visible band and another the index of an infrared band if you want the program to produce images that look like color IR film products. The default is to produce the three-band product with default visible band 2 and IR band 4.

Mask? A Boolean variable; if specified, then band 1 of the image will be treated as a mask: pixels with value 0 in band 1 will not be processed, and they will be labeled 0 in the classification products. Mask is the default.

Class Mask? A Boolean variable, default no. If specified, then you will be prompted for the file name of a previous cluster map and the list of the cluster indices you would like to select for subsequent clustering. I have used this option, new to Version 13, to subdivide three classes of shallow water and wetlands in a complex area into nine classes. Before it was possible, but required several steps and was not as efficient.

Roberts? It this Boolean variable is set, then Robert's gradient will be used to find the initial estimate of boundary. The default is no Robert's gradient.

Classification Method. The classification method: an integer parameter, as follows:

(1) Per-pixel classification.
(2) Spatially corrected classification (the default); you must supply the number of 8-nearest neighbors which should match in order for a classification to be accepted (default 2 neighbors alike).
(3) Context classification; you must specify whether you want many finely detailed context classes or as few as is possible (default many).

Iterations. The number of iterations (default 2, range 1 to 5).

Pure pixels. The percent of pixels which are pure (range 25 to 75, default 45).

Error weights. The weights for the two error cases discussed above (default 1, range 1 to 9). The weights can be used to slightly favor more or fewer clusters. For example, if the same real class-different cluster weight is made higher, than fewer clusters will be produced (because the extreme one-cluster "solution" will be perfect in this error Case).

EXECUTION CHARACTERISTICS

The following data refer to test runs on a 512- by 512-pixel three-band image. The computer is a low end VAX 2000 workstation running VMS 4.5. This machine is slightly slower than a VAX11/780, a system which is often used for benchmark comparison of computers. Although no floating point hardware is present, the modules which do use floating point arithmetic take less than 1 percent of the overall time, so that little improvement would result from its presence.

The first pass through the data required 65 percent of the 26 minutes the basic two iteration (per-pixel classification) requires. The next pass takes less time because a much better estimate is available for the set of pure pixels. The spatial fixup operations require an additional 40 seconds, and context classification takes almost three minutes more. One might say the added burden of the more specialized classifier is insignificant. The size of the image has the predictable almost linear influence on the time; however, the dimensionality (number of bands) has only minor influence. A good estimate is that the time taken is proportional to the square root of the number of bands. Thus, nine-band data should take about twice as much time as three-band data. For example, the original image in this test contained 11 bands. Two iterations took slightly over an hour.

SUMMARY AND DIRECTIONS OF FUTURE WORK

I have described new features of the latest version of AMOEBA and illustrated them on a difficult high resolution image. The main line of reasoning which leads to the method has been outlined. Many of the details of the program are revealed here for the first time.

Version 1 of AMOEBA appeared in 1976; it shows no sign of stopping with Version 13. Problems being pursued here include the development of the 3" by 3" context classifier, the problem of the use of texture in classification, the boundary estimation problem in high resolution low noise imagery, and the problem of approximately recovering original data from the three-band product.

ACKNOWLEDGMENTS

The program AMOEBA through Version 6 was supported by the NASA. The USGS/EROS Data Center supported development through Version 10. Professor Townshend's group, particularly Jeff Settle, supplied data and detected bugs. Hsien-Min Yang of the University of Arizona, working with Professor Schowengerdt, made useful comments. It is a pleasure to acknowledge this assistance.

REFERENCES

Anderberg, M. R., 1973. *Cluster Analysis for Applications*, Academic Press, New York.

Andrews, H. C., 1972. *Introduction to Mathematical Techniques in Pattern Recognition*, Wiley-Interscience, New York.

Bauer, M. E., L. F. Silvia, R. M. Hoffer, and M. F.. Baumgardner, 1977. *Agricultural Scene Understanding*, LARS Contract Report 112667, Laboratory for Applications of Remote Sensing, Purdue University, West Lafayette, Indiana.

Bryant, J., 1978. On the clustering of multidimensional pictorial data, *Pattern Recognition* 11:115–125.

———, 1979. *Applications of Clustering in Multi-Image Data Analysis*, Report No. 18, 75 pp., Department of Mathematics, Texas A&M University, College Station, Texas.

———, 1988. On displaying multispectral imagery, *Photogrammetric Engineering and Remote Sensing* 54:1739–1743.

———, 1989. A fast classifier for image data, *Pattern Recognition* 22:45–48.

Chittinei, C. B., 1983. Probabilistic cluster labeling of imagery data, *IEEE Trans. on Geoscience and Remote Sensing* GE-21:145–155.

Duran, B. S., and P. L. Odell, 1974. *Cluster Analysis*. Lecture Notes in Economics and Mathematical Systems, Vol. 100, Springer, New York.

Gurney, C. M., and J. R. G. Townshend, 1983. The use of contextural information in the classification of remotely sensed data, *Photogrammetric Engineering and Remote Sensing* 49:55–64.

Haralick, R. M., and I. Dinstein, 1975. A spatial clustering procedure for multi-image data, *IEEE Trans. Circuits and Systems* CAS-22:440–450.

Haralick, R. M., and L. G. Shapiro, 1985. Image segmentation techniques, *Comput. Vision, Graphics, and Image Processing* 29:100–132.

Hill, G. J. E., and G. D. Kelly, 1986a. Integrating Landsat and land systems for cover maps in southern inland Queensland, *Australian Geographical Studies* 24:235–243.

———, 1986b. Habitat mapping by Landsat for aerial census of kangaroos. *Remote Sensing of Environment* 21:53–60.

Jain, A. K., and R. C. Dubes, 1988. *Algorithms for Clustering Data*, Prentice Hall, Englewood Cliffs, N. J.

Jenson, S. K., T. R. Loveland, and J. Bryant, 1982. Evaluation of AMOEBA: a spectral-spatial classification method, *J. Appl. Photographic Engineering* 8:159–162.

Kast, K. L., and Davis, B. J., 1977. *Tests of a Spectral/Spatial Classifier*, LARS Contract Report 112877, Laboratory for Applications of Remote Sensing, Purdue University, West Lafayette, Indiana.

Kelly, G. D., and G. J. E. Hill, 1987. Updating maps of climax vegetation cover with Landsat MSS data in Queensland, Australia, *Photogrammetric Engineering and Remote Sensing* 53:633–637.

Rand, W. M., 1971. Objective criteria for the evaluation of clustering method, *J. Amer. Statis. Assoc.* 66:846–850.

Strickland, R. N., and R. A. Schowengerdt, 1988. *Intergrated Circuit Inspection by Intelligent Computer Visiion*, Status Report, Engineering Experiment Station, College of Engineering and Mines, University of Arizona, Tuscon, Arizona, April 1988.

Strahler, A. H., C. E. Woodcock, and J. A. Smith, 1986. On the nature of models in remote sensing, *Remote Sensing of Environment* 20:121–139.

Trivedi, M. M., and J. C. Bezdek, 1986. Low-level segmentation of aerial images with fuzzy clustering, *IEEE Trans. Systems, Man, and Cybernetics* SMC-16:598–598.

Wharton, S. W., 1982. A contextual classification method for recognizing land use patterns in remotely sensed data. *Pattern Recognition* 15:113–120.

Woodcock, C. E., and A. H. Strahler, 1987. The factor of scale in remote sensing, *Remote Sensing of Environment* 21:311–332.

Yarman-Vural, F., and E. Ataman, 1987. Noise, histogram and cluster validity for guassian-mixed data, *Pattern Recognition* 20:385–401.