Decomposition of Two-Dimensional Shapes by Graph-Theoretic Clustering

LINDA G. SHAPIRO, ASSOCIATE MEMBER, IEEE, AND ROBERT M. HARALICK, SENIOR MEMBER, IEEE

Abstract—This paper describes a technique for transforming a two-dimensional shape into a binary relation whose clusters represent the intuitively pleasing simple parts of the shape. The binary relation can be defined on the set of boundary points of the shape or on the set of line segments of a piecewise linear approximation to the boundary. The relation includes all pairs of vertices (or segments) such that the line segment joining the pair lies entirely interior to the boundary of the shape. The graph-theoretic clustering method first determines dense regions, which are local regions of high compactness, and then forms clusters by merging together those dense regions having high enough overlap. Using this procedure on handdrawn colon shapes copied from an X-ray and on handprinted characters, the parts determined by the clustering often correspond well to decompositions that a human might make.

Index Terms—Clustering, graph-theoretic clustering, relation clustering, shape, shape decomposition, shape matching.

I. Introduction

WE USE many pieces of information about an object in order to recognize it. The size, shape, color, and position of the object are obviously important. The recognition decision may also be based on previous encounters with similar objects. However, humans can recognize objects from gray-scale pictures with limited views, and often can recognize or at least guess at objects from their shapes alone. In this paper we suggest an approach which might enable a computer to do so, too.

A complex object may be composed of many parts, each with its own shape. The entire object also has a shape—the shape of its silhouette (or the set of shapes of the silhouettes of its characteristic views). Since humans can recognize objects from their silhouettes, it seems reasonable to develop computer algorithms to analyze the shapes of silhouettes as part of an effort to recognize the corresponding objects. In this paper we will disregard the problems of extracting the silhouettes from digitized pictures and assume that a representation of the object boundary is provided. Such a representation can consist of an ordered sequence of (x,y) coordinates of points around the boundary of the object (obtained, for example, from a chain encoding, Freeman [8]) or an ordered sequence of line segments representing a piecewise linear approximation to the boundary. The task of the computer is to produce a structural description of the shape of the object. This description would then be used by matching procedures attempting to recognize the object.

Manuscript received July 22, 1978; revised April 7, 1978. This paper was supported in part under Grants AFOSR 77-3307, MCS 77-22386, and MCS 77-23 945.

L. G. Shapiro was with the Department of Computer Science, Kansas State University, Manhattan, KS 66506. She is now with the Department of Computer Science, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061.

R. M. Haralick was with the Department of Electrical Engineering, University of Kansas, Lawrence, KS 66044. He is now with the Department of Electrical Engineering, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061.

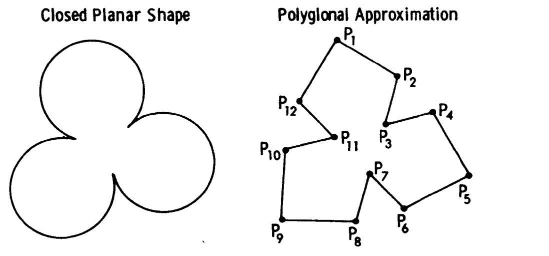


Fig. 1. A closed planar shape and a polygonal approximation to it.

One test for a good object description is by comparison with the perceptual judgment which humans make. There is a body of psychological literature describing such judgments. However, in this paper we are not interested in finding algorithms which exactly imitate human perceptual judgments. Instead, we wish to explore algorithms which can produce from a polygonal shape a description rich enough to permit successful shape matching or similarity determination.

Consider the curved two-dimensional object shown in Fig. 1. A human might describe this object as being composed of three lobes of approximately the same size and about equally spaced around the object. When a human looks at the picture, the lobes stand out as almost separate entities; they are the simple parts of the object. This motivates the need for an algorithm to decompose an object into possibly overlapping simple parts and a model for describing the relationships among the parts. In this paper we will propose a new method for decomposing a polygonal shape into meaningful parts.

Our method (which will be described in detail in Section III) calculates a binary relation which captures important information about the shape of a polygonal object. The binary relation is constructed as follows. Given two points P_1 and P_2 on the boundary of the object, we can construct a straight line from P_1 to P_2. If the line lies completely interior to the boundary of the object, then the ordered pair (P_1, P_2) belongs to the relation. Given a piecewise linear approximation to the boundary of the object, we can calculate a similar relation which approximates the same shape information using line segments. We suggest that the clusters of either relation can define the simple parts of the object.

Section II briefly reviews some related literature. Section III is the body of the paper and describes the clustering procedure for the decomposition of a shape. Section IV suggests some ways of working with boundary lines rather than with boundary points.

II. Related Literature

Pavlidis has tackled the shape-recognition problem from several different approaches. One early approach was the decomposition of shapes into primary convex subsets (Pavlidis [14]). The primary convex subsets and nuclei (regions where primary convex subsets overlap) form the nodes of a labeled graph representing the original shape (Pavlidis [15]). A related approach was the decomposition into convex parts, T-shaped parts, and spirals (Feng and Pavlidis [7]). A later approach was the syntactic analysis of shapes and structural description by regular expressions (Pavlidis and Ali [17]). Recent algorithms have used piecewise linear approximations to the original shape as input data (Pavlidis and Horowitz [16]). Pavlidis has also compiled an excellent bibliography on algorithms for shape analysis (Pavlidis [18]).

Maruyama [11] suggests a decomposition of shapes into angularly simple regions. Each angularly simple region has at least one interior point which can "see" its entire boundary. Davis [3]-[5] has worked on the problems of shape representation and matching. He decomposes a shape into a piecewise approximation represented by sides and angles. Two shapes are considered similar if a mapping can be found from the angles of one shape to the angles of the other. A relaxation procedure is used to determine the mapping with the constraint being the distance between the angles.

Other work in shape analysis, which is not as closely related to our proposed work, includes the chain encodings of Freeman [8], the medial axis transformation of Blum [2], and the analysis of convex blobs by computing dominant points (Rosenberg [19], [20]; Langridge [10]; O'Callaghan [13]).

III. Decomposition of a Shape Represented by a Sequence of Boundary Points

A. The Interior Line-Segment Relation

Let $P = \{P_1, P_2, \cdots, P_n\}$ be an ordered set of points representing the vertices of a polygonal approximation to the boundary of a planar shape. (The boundary may be open or closed; however, our examples will be for closed boundaries.) We would like to partition the set P into possibly overlapping ordered subsets, where each subset represents a polygonal approximation of a simple part of the original shape. $\{\{P_{12}, P_1, P_2, P_3\}, \{P_4, P_5, P_6, P_7\}, \{P_8, P_9, P_{10}, P_{11}\}\}$ is an intuitive partition of the points of Fig. 1.

Consider a line segment joining any two points of the polygonal approximation to the shape. Such a line segment falls into one of three categories.

1) The entire line segment lies interior to the boundary of the polygonal approximation. (This includes the original edges of the approximation.)

2) The entire line segment lies exterior to the boundary.

3) The line segment intersects the boundary in one or more points.

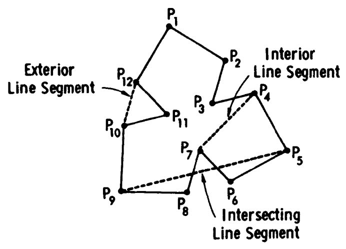


Fig. 2. The three different kinds of line segments.

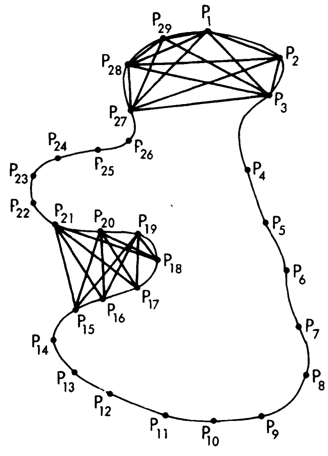


Fig. 3. How interior and exterior line segments relate to protrusions and intrusions.

Fig. 2 illustrates the three kinds of line segments on the curve of Fig. 1. The line segment from P_4 to P_7 lies inside the boundary of the polygonal approximation and is called an *interior* line segment. The line segment from P_{10} to P_{12} lies outside the boundary of the polygonal approximation and is called an *exterior* line segment. The line segment from P_5 to P_9 intersects the boundary of the polygonal approximation at two points and is an *intersecting* line segment.

Intuitively, a set of points which should be grouped together as a simple part of the curve will be joined by interior line segments. A set of points where all line segments between points of the set are exterior line segments represents an intrusion to the object. Sets of points joined by intersecting line segments have, in general, no spatial relationships.

Fig. 3 illustrates these concepts. The set of points $\{P_1, P_2, P_3, P_{27}, P_{28}, P_{29}\}$ form a protrusion of the curve which is, intuitively, a simple part of the curve. The sets of points $\{P_{15}, P_{16}, P_{17}, P_{18}\}$ and $\{P_{19}, P_{20}, P_{21}\}$ have only exterior line segments between them. These two sets correspond to the two sides of a U-shaped intrusion to the object. We will show how the relationship defined by the interior line segments can be used to partition the object into simple parts. We expect, in future research, to use the intrusions defined by the exterior-line-segment relation to aid in the partitioning algorithms.

The *interior-line-segment* relation L_I consists of all pairs of vertices (P_i, P_j) such that the line segment joining P_i and P_j is an interior line segment. We can represent this relation by a graph G where each vertex is represented by a node, and an edge joining vertices P_i and P_j indicates that (P_i, P_j) and (P_j, P_i) belong to L_I. If A is a subset of vertices of P satisfying the condition that every line segment joining two vertices of A is an interior line segment, then the Cartesian product $A \times A$ is contained in the relation L_I, and A is a complete subgraph of G representing a convex polygonal part of the shape. Furthermore, if A is a maximal set with this property, then it represents a maximal convex polygonal part of the shape.

Clearly, partitioning a polygon into maximal convex parts sometimes yields a good intuitive decomposition. Fig. 4 shows a simple polygon and the graph representation of the relation L_I on the vertices of the polygon. In Fig. 4, the sets $A_1 = \{P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8\}$ and $A_2 = \{P_8, P_9, P_{10}, P_{11}, P_{12}, P_{13}, P_{14}, P_{15}, P_{16}, P_1\}$ are the maximal subsets such that $A_1 \times A_1$ and $A_2 \times A_2$ belong to the relation L_I. A_1 and A_2 are also the maximal convex polygonal parts of the original polygon and represent the simple parts into which humans would most likely partition the figure.

One important property of the relation L_I for shape decomposition is that L_I is invariant under any linear transformation of the original vertices of the object. Thus translation, rotation, scaling, skewing, projection, and perspective transformation of the object will not alter its decomposition. Fig. 5 shows the relation L_I on several different transformations of a simple object.

While maximal convex polygons sometimes provide a basis for decomposition, they are not tolerant of small perturbations that might appear along the boundary of a polygonal approximation of a digitized picture of a shape. For example, Fig. 6 shows a polygon which is very similar in shape to the polygon of Fig. 4, but with some "boundary noise" added. The maximal convex subset decomposition will not divide this polygon into the two natural parts, due to concavities introduced by the noise.

We would like a simple-shaped part to be one which is maximal in size and is as close to being convex as possible. We call a subset of vertices with this property a *cluster* of the relation L_I. In the next section we will discuss graph-theoretic clustering techniques which can be used to determine the clusters of the relation L_I and, therefore, a decomposition of the shape into its simple-shaped parts.

B. A Graph-Theoretic Method for Determining Clusters

In Section III-A we defined a binary relation L_I on a set of points P representing a polygonal approximation to a planar curve and proposed that the clusters of L_I would represent simple parts of the object bounded by the curve. In this section we present a general clustering algorithm which can be used to compute clusters of the L_I relation. The graph-theoretic clustering method is motivated by Gottlieb and Kumar [9] and Auguston and Minker [1] who determined the clusters of a graph by finding all the cliques of the graph and iteratively merging those cliques having high enough overlap.

We begin with some definitions. Let S be a set and $R \subseteq S \times S$ be a binary relation on S. The pair (S, R) is called a digraph or, when R is symmetric, a graph. The elements of S are called

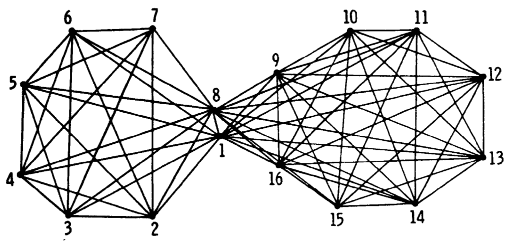


Fig. 4. A graph of the internal line segments of a simply decomposable shape.

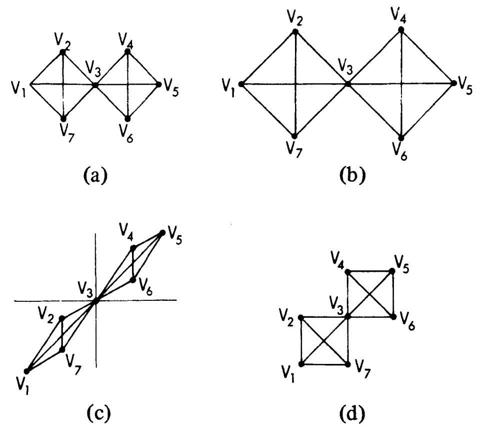


Fig. 5. The relation L_I on a shape and on 4 linear transformations of the shape. (a) Original object. (b) Scaled object. (c) Skewed object. (d) Rotated object.

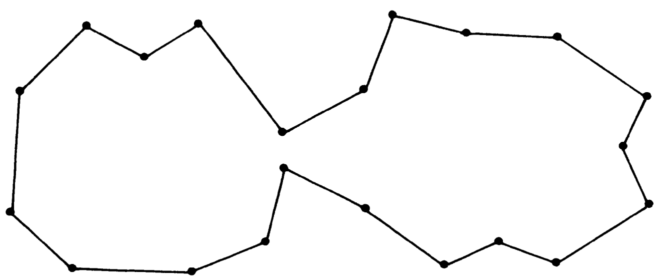
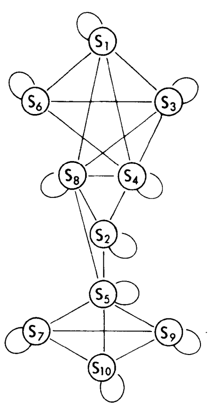


Fig. 6. A simple decomposable shape which is similar to the one in Fig. 4, but whose decomposition does not correspond to its maximally convex parts.

nodes. If a pair of nodes (X, Y) belongs to R, then we say that *Y is a neighbor of X*. The set of all nodes Y such that Y is a neighbor of X is called the *neighborhood* of X (neighborhood (X)). (Note that unless R is symmetric, Y being a neighbor of X does not imply that X is a neighbor of Y.) In this paper we will be concerned with symmetric relations since the relation L_I is symmetric.

Fig. 7 illustrates a symmetric and reflexive relation on ten nodes in graph representation. We will use this relation to illustrate the concepts presented in this section.

In the spirit of Auguston and Minker [1], we define a *cluster* as a maximal chain of dense or highly connected neighborhoods. Auguston and Minker used cliques, the most highly compact neighborhoods possible. However, in order to reduce the strictness of the clique criterion as well as the computational burden of finding all cliques, we will define a density



Node	Neighborhood of Node
1	1,3,4,6,8
2	2,4,5,8
3	1,3,4,6,8
4	1,2,3,4,6,8
5	2,5,7,8,9,10
6	1,3,4,6
7	5,7,9,10
8	1,2,3,4,5,8
9	5,7,9,10
10	5,7,9,10

Fig. 7. A 10-node graph which is used to illustrate how the graph-theoretic clustering procedure works.

function and use it to determine the dense region around each node.

The *conditional density* $D(X|Y)$ of node X given node Y is defined as the number of nodes in the neighborhood of Y which have X as a neighbor. That is

$$D(X|Y) = \#\{N \in S | (N, X) \in R \text{ and } (Y, N) \in R\}.$$

Notice that if R is symmetric, then $D(X|Y)$ is just the number of neighbors common to both node X and node Y. In this case

$$\begin{aligned} D(X|Y) &= D(Y|X) \\ &= \#(\text{neighborhood } (X) \cap \text{neighborhood } (Y)). \end{aligned}$$

If M is a maximal subset of S such that $M \times M \subseteq R$, then M is a *clique* of the relation R. If X is a node of S, then there is some number $N(X) \geqslant 1$ of cliques of R containing X, and of these cliques there is one or more that is largest in size. These largest cliques are called *major cliques*. In Fig. 7, the sets $B_1 = \{S_1, S_3, S_4, S_8\}$, $B_2 = \{S_5, S_7, S_9, S_{10}\}$, $B_3 = \{S_2, S_4, S_8\}$, $B_4 = \{S_2, S_5, S_8\}$, and $B_5 = \{S_1, S_3, S_4, S_6\}$are cliques of the relation. Note that node S_8 is in cliques B_1, B_3, and B_4. B_1 is the major clique of node S_8, but B_3 and B_4 are also major cliques since they are the only cliques of node S_2.

The density function $D(Y|X)$ can be used to find large regions which are not quite as dense as major cliques. For each node, X in S and integer K define the region $Z(X, K)$ by

$$Z(X, K) = \{Y \in S | D(Y|X) \geqslant K\}.$$

For small values of K, the region $Z(X, K)$ around node X is likely to be large and loose. As K becomes larger, the region $Z(X, K)$ becomes smaller and more tightly interconnected and compact.

If C is a major clique of size M, then $X, Y \in C$ implies that $D(Y|X) \geqslant M$. Thus $C \subseteq Z(X, M)$. Therefore, the largest sized clique associated with each node X must be contained in $Z(X, K)$ for any $K \leqslant M$ and we have $K \leqslant M \leqslant \#Z(X, K)$. Then any value of K which does not satisfy $K \leqslant \#Z(X, K)$ could not be the size of a major clique for node X. Hence it is only natural to consider

$$Z(X) = Z(X, N)$$

D(X|Y)

Y \ X	1	2	3	4	5	6	7	8	9	10
1	5	2	5	5	1	4	0	4	0	0
2	2	4	2	3	3	1	1	4	1	1
3	5	2	5	5	1	4	0	4	0	0
4	5	3	5	6	2	4	0	5	0	0
5	1	3	1	2	6	0	4	3	4	4
6	4	1	4	4	0	4	0	3	0	0
7	0	1	0	0	4	0	4	1	4	4
8	4	4	4	5	3	3	1	6	1	1
9	0	1	0	0	4	0	4	1	4	4
10	0	1	0	0	4	0	4	1	4	4

Node N	Z (N)	C(Z(N))
1	{1,3,4,6,8}	0.92
2	{2,4,5,8}	0.875
3	{1,3,4,6,8}	0.92
4	{1,3,4,6,8}	0.92
5	{5,7,9,10}	1.0
6	{1,3,4,6}	1.0
7	{5,7,9,10}	1.0
8	{1,2,3,4,8}	0.84
9	{5,7,9,10}	1.0
10	{5,7,9,10}	1.0

Fig. 8. The table of conditional densities for the graph of Fig. 7 and the tentative compact neighborhoods around each node derived from the conditional density function.

where

$$N = \max \{K | \#Z(X, K) \geqslant K\}$$

as the candidate dense region around X. Fig. 8 shows the conditional densities for each pair of nodes and the sets $Z(X)$ for the relation of Fig. 7.

Because each $Z(X)$ contains only those nodes with highest conditional densities, there is a reasonable basis for expecting that each $Z(X)$ will not contain much more than the nodes of the major cliques of node X. We only want to consider a set $Z(X)$ if it is large enough, and it includes only those nodes that are related to a large percentage of the other nodes in the set. To this end we define a measure of the association (or relatedness) of a node to a set of nodes and a measure of compactness (or interrelatedness) of a set of nodes.

For any node N in a subset B of S, we define the *association* $A(N|B)$ of node N to subset B as the ratio of the number of nodes in B that are neighbors of N to the total number of nodes in B.

$$A(N|B) = \frac{\text{neighborhood } (N) \cap B}{\#B}$$

If all the nodes in B are neighbors of N, then $A(N|B) = 1$, and if none of the nodes in B are neighbors of N, then $A(N|B) = 0$.

For any subset B of S, we define the *compactness* $C(B)$ as the average association of the nodes of B.

$$C(B) = \frac{1}{\#B} \sum_{N \in B} A(N|B).$$

If each of the nodes in subset B is related to itself and to every other node in B, then $C(B) = 1$. If none of the nodes in B are related to any node in B, then $C(B) = 0$.

We will consider only those sets $Z(X)$ which are large enough, which include only those nodes whose association is high enough, and whose overall compactness is high enough.

Given an association threshold MINASSOCIATION, a (possibly higher) compactness threshold MINCOMPACTNESS, and a size threshold MINSIZE, we define a *dense region* of R to be a subset $B \subseteq S$ satisfying

1) $B = \{N \in Z(X) | A(N|Z(X)) \geqslant \text{MINASSOCIATION}\}$ for some $X \in S$;
2) $C(B) \geqslant$ MINCOMPACTNESS;
3) $\#B \geqslant$ MINSIZE.

In Fig. 8, the dense regions for MINCOMPACTNESS = 0.8, MINASSOCIATION = 0.5, and MINSIZE = 1 are $\{S_1, S_3, S_4, S_6, S_8\}$, $\{S_2, S_4, S_5, S_8\}$, $\{S_5, S_7, S_9, S_{10}\}$, $\{S_1, S_3, S_4, S_6\}$, and $\{S_1, S_2, S_3, S_4, S_8\}$.

The clusters of R can be determined by taking the union of those dense regions of R which overlap one another to a high enough degree. Let MINOVERLAP be the specified degree of overlap. We define the *dense-region relation F* as the set of pairs of dense regions (D_1, D_2) such that the fraction of nodes in D_1 which are also in D_2 is greater than MINOVERLAP or the fraction of nodes in D_2 which are also in D_1 is greater than MINOVERLAP.

$$F = \{(D_1, D_2) | D_1, D_2 \text{ are dense regions of } R,$$
$$\#D_1 \cap D_2 / \#D_1 \geqslant \text{MINOVERLAP or}$$
$$\#D_1 \cap D_2 / \#D_2 \geqslant \text{MINOVERLAP}\}$$

For example, if $D_1 = \{S_1, S_2, S_3, S_4, S_8\}$, $D_2 = \{S_1, S_3, S_4, S_6, S_8\}$, and MINOVERLAP = 0.8, then (D_1, D_2) and (D_2, D_1) belong to F.

The relation F is symmetric and reflexive. Its transitive closure is an equivalence relation. We define a cluster to be the union of the set of dense regions in each equivalence class. Although the equivalence classes whose members are dense regions are naturally disjoint, it is possible for the union of the dense regions in one class to be contained in the union of the dense regions in another class. The simplest way to handle this is to iterate by merging clusters which overlap much in the same way that the dense regions were merged. The dense regions at each iteration are the clusters of the previous iteration. The iterations can proceed until the clusters can no longer be merged.

For MINOVERLAP = 0.75, MINCOMPACTNESS = 0.8, MINASSOCIATION = 0.5, and MINSIZE = 3, the clusters of the relation of Fig. 7 are

$$\{S_1, S_2, S_3, S_4, S_6, S_8\}$$
$$\{S_2, S_4, S_5, S_8\}$$
$$\{S_5, S_7, S_9, S_{10}\}$$

C. Results of Graph-Theoretic Clustering on L_I

A series of experiments on shape analysis has been carried out using the clustering method just discussed in the following manner. Values for MINOVERLAP, MINCOMPACTNESS, MINASSOCIATION, and MINSIZE were read in prior to processing each object and a set of clusters was produced as in Section III-B. The value of MINOVERLAP was then decreased by multiplying the current value by 0.98. The clusters were relabeled as dense regions and fed back into the procedure with the new value of MINOVERLAP to produce a new set of clusters. This process was repeated for a total of fifteen iterations on each object with the initial value of MINOVERLAP at 0.9, and the values of MINCOMPACTNESS, MINASSOCIATION, and MINSIZE constant at 0.8, 0.75, and 7, respectively. After the fifteen iterations were completed, any node N which was not included in any cluster was added by a best fit routine to that cluster S to which it was most highly associated; that is, S satisfied $A(N|S) \geqslant A(N|T)$ for all clusters T. In case the maximal association was the same for several clusters, node N was added to the first such cluster.

In general, the first iteration on each object produced more clusters than a human would intuitively pick. This was due to the fact that the relatively high initial MINOVERLAP of 0.9 did not permit enough merging of dense regions. With the clusters relabeled as dense regions and MINOVERLAP reduced, more merging took place and the number of clusters was reduced.

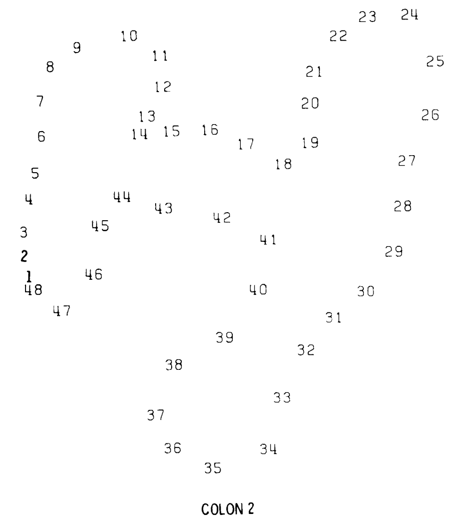


Fig. 9. The boundary points of the shape COLON2.

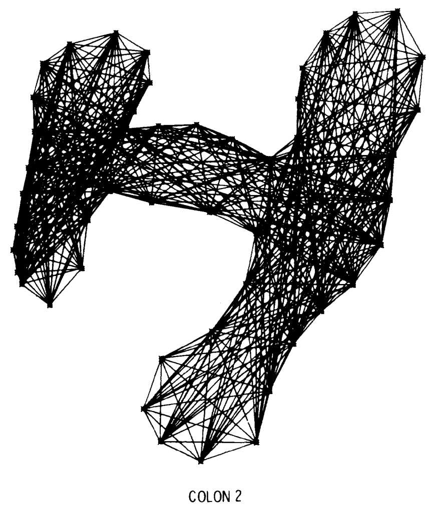


Fig. 10. The relation L_I on the shape COLON2.

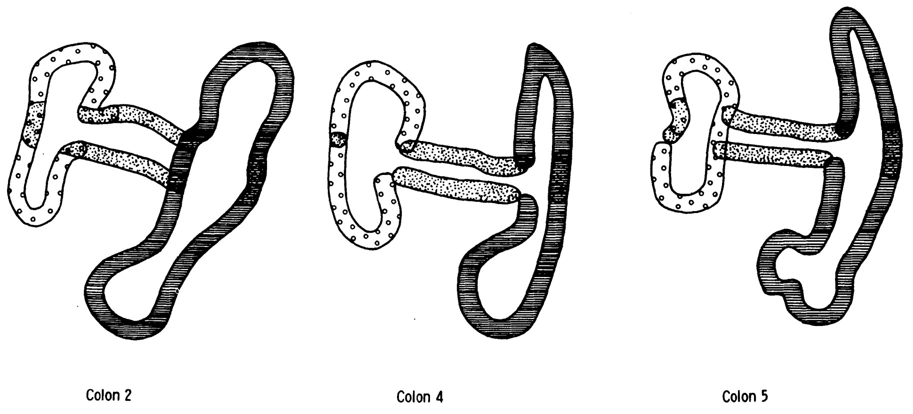


Fig. 11. The decompositions of COLON2, COLON4, and COLON5.

In most cases, the number of clusters stabilized for some value of MINOVERLAP < 0.9 and the clusters remained constant for the rest of the iterations.

One test of the procedure used objects produced by having several different people copy the shape of a colon from an anatomy book (Meschan and Farrer-Meschan [12]). After digitizing, the average number of boundary points per colon shape was 107. To reduce costs, every other point was used to determine the L_I relation. Fig. 9 shows the boundary points of one of the shapes, COLON2. These points were the input to the procedure. Fig. 10 shows the relation L_I on the vertices of COLON2. Fig. 11 shows the clusters obtained from three colon shapes: COLON2, COLON4, and COLON5. The clusters in Fig. 11 are delimitted by closed curves drawn around groups of boundary points which belong to the same cluster. Each cluster is marked with a different symbol. Thus the areas covered by small dots, the areas covered by large dots, and the areas covered by stripes represent three different clusters. Areas containing more than one type of symbol represent points that are in more than one cluster. The number of clusters produced from each of the three objects stabilized to three—at iteration 7 for COLON2 and COLON4, and at iteration 11 for COLON5.

The clusters shown in Fig. 11 correspond well to an intuitive decomposition of the objects. Each of COLON2, COLON4, and COLON5 has a left part, a horizontal midsection, and a right part. Thus a very simple matching procedure could be used to find that COLON2, COLON4, and COLON5 are similar.

One property of the clustering method is that the clusters are not necessarily disjoint. This means that if there are several different intuitive decompositions of an object, the procedure will not choose among them, but instead will produce clusters representing all the possible decompositions. For example, in the decompositions of each of COLON2, COLON4, and COLON5, the horizontal cluster overlaps both the left cluster and the right cluster. The horizontal cluster shows that there is a piece of the object that extends all the way across it. The left cluster and right cluster do not have to be broken into two parts just because the horizontal cluster crosses them. Similarly, the horizontal cluster does not have to be cut short just because the left cluster and right cluster cross it.

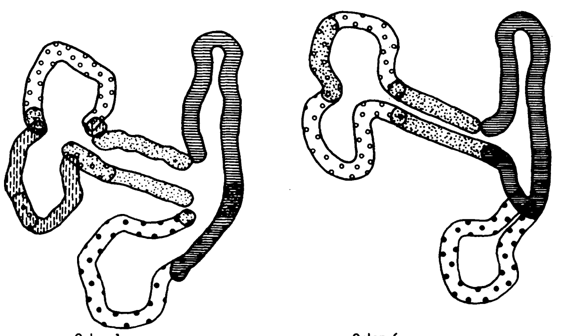


Fig. 12. The decompositions of COLON1 and COLON6.

Fig. 12 shows the decompositions of two similar shapes COLON1 and COLON6. COLON1 has five clusters: upper and lower left parts, a horizontal midsection, and upper and lower right parts. COLON6 has four clusters: one left part, a horizontal midsection, and upper and lower right parts. A matching procedure would have to decide whether COLON1 and COLON6 are similar enough. This could be done by a heuristic procedure with built-in thresholds for merging simple parts. For example, two simple parts might be merged if they are adjacent and the angle formed by their major axes is close to 180°.

The clustering procedure was also tested on nine handprinted characters—three H's, three K's, and three X's. These shapes were digitized to produce an average of 56.22 boundary points. Again every other boundary point was used as input to the program. Fig. 13 shows the decomposition of these characters for MINOVERLAP = 0.9, MINCOMPACTNESS = 0.8, MINASSOCIATION = 0.75, and MINSIZE = 7. Fig. 14 shows the decomposition of the same characters with MINASSOCIATION changed to 0.7. In Fig. 13, the K's all have the same decomposition, but the H's and X's would need some heuristic merging to be judged similar. In Fig. 14, H3 has a more suitable decomposition making it similar to H2, but not to H1 which has a more well-defined horizontal section. X2 and X3 are now similar, but X1 needs some heuristic merging. K2 and K3 remained the same, but the decomposition of K1 is worse than in Fig. 13. These results suggest that merging of clusters should be based at least partly on geometric relationships.

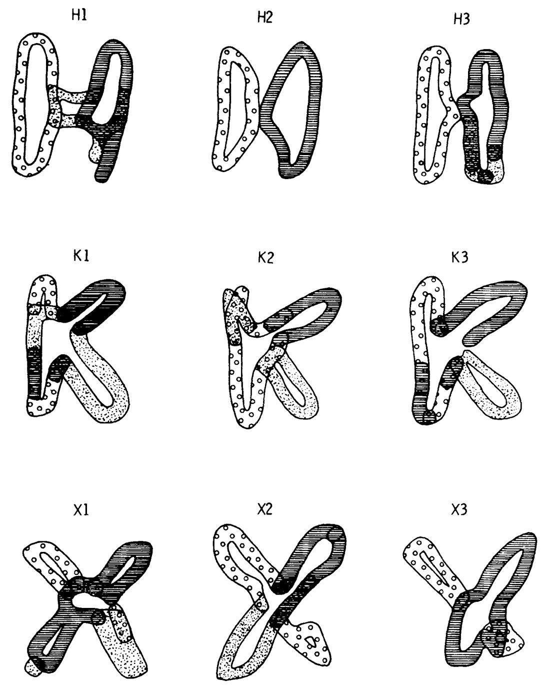


Fig. 13. The decompositions of nine handprinted characters with MINOVERLAP = 0.9, MINCOMPACTNESS = 0.8, MAXDIFFERENCE = 0.25, and MINSIZE = 7.

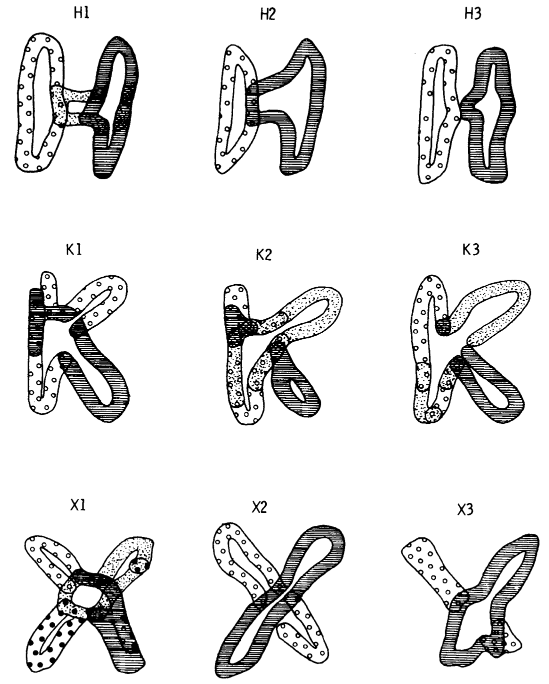


Fig. 14. The decomposition of the nine handprinted characters of Fig. 13 with MINOVERLAP = 0.9, MINCOMPACTNESS = 0.8, MINSIZE = 7, and MAXDIFFERENCE changed to 0.3.

D. The Algorithm and Its Complexity

The decomposition procedure just described can be divided into two main parts: 1) finding the relation L_I and 2) performing the clustering. We will discuss the complexity of each part.

1) Algorithm for Finding the Relation L_I: Given a sequence of points $\{P_1 = (X_1, Y_1), P_2 = (X_2, Y_2), \cdots, P_n = (X_n, Y_n)\}$ representing the boundary of a closed planar polygonal shape, the following algorithm determines whether the line segment from P_i to P_j is an interior line segment.

```
procedure INLINE($I, J, N$);
declare $I, J, N$ integer;
comment If the line segment $L_S$ between the points
        $P_I$ and $P_J$ is an interior line segment
        return SUCCESS, else return FAILURE;
if $P_I$ is adjacent to $P_J$
then return (SUCCESS);
$L_S$ = the line segment from $P_I$ to $P_J$;
MID = the midpoint of $L_S$;
INTERSECTION_COUNT = 0;
do for all $N$ line segments $L$ around the boundary
  of the curve
  begin
    if a vertical line passing through MID intersects
      $L$ below MID
    then INTERSECTION_COUNT
       = INTERSECTION_COUNT + 1;
    if $L_S$ intersects $L$
    then return (FAILURE);
  end
comment $L_S$ is not an intersecting line segment.
        if INTERSECTION_COUNT is odd, then $L_S$
        is interior. If INTERSECTION_COUNT is
        even, then $L_S$ is exterior;
if INTERSECTION_COUNT is odd,
then return (SUCCESS);
else return (FAILURE);
end INLINE;
```

For each pair of points, a maximum of n times through the loop in INLINE determines if the line segment between the points is interior. For n points, there are $n(n - 1)/2$ line segments to be tested. Thus, in the worst (convex) case, the complexity of the algorithm is proportional to n^3. For a large object with closely spaced boundary points, this would be prohibitive. Shamos, in a private communication, has indicated that an algorithm of order $0(n^2)$ exists, and we plan to implement this algorithm in the future.

2) The Graph-Theoretic Clustering Algorithm: The graph-theoretic clustering method can be summarized as follows.

```
0) ITERATION_COUNT = 0
1) Find the dense regions around each node
   do while ITERATION_COUNT <
   NUMBER_OF_ITERATIONS;
     begin
      2) Form a relation of the dense regions (two dense
         regions belong to the relation if they overlap
         one another enough)
      3) Find the clusters (equivalence classes of the rela-
         tion of 2)
      4) Create a new set of dense regions (the clusters
         of 3)
      5) ITERATION_COUNT = ITERATION_COUNT + 1
     end
6) Put any leftover nodes in the best fit cluster.
```

To determine the dense region around a node X, we first compute $D(Y|X)$ for every other node Y in S. Since $D(Y|X) = \#(\text{neighborhood }(X) \cap \text{neighborhood }(Y))$, this involves $n - 1$ set intersections. Since L_I is symmetric, $D(Y|X) = D(X|Y)$; so, for n nodes, computing the conditional densities takes $n(n - 1)/2$ set intersections. Secondly, we use the densities to determine a dense-region candidate set for node X. To do this, we need to find the largest positive integer K such that $\#\{Y|D(Y|X) \geqslant K\} \geqslant K$. This can be done with a binary search on all K between 1 and n, which involves $\log_2 n$ steps, or, for n nodes, $n \log_2 n$ steps.

After a candidate dense region is determined, we calculate what proportion of nodes in the candidate set each node in the candidate set is related to and throw out those nodes whose associations are too low. This takes one set intersection per node. Then the remainder of the nodes in the candidate dense region are checked to make sure they have a high enough average association. This checking requires a maximum of n set intersections. Finally, if the set passes the test, the number of nodes must be compared to MINSIZE. Thus, for each of the n possible candidate dense regions, we perform at most $2n$ set intersections plus one size comparison. Thus, for n nodes, finding the initial dense regions is, at most, proportional to n^2 operations.

After the initial dense regions are computed, we enter the loop where the main operations are 1) forming the dense-region relation which requires, at most, n^2 set intersections and 2) producing the equivalence relation which involves, at most, n^2 comparisons. The loop is executed NUMBER_OF_ITERATIONS times, where NUMBER_OF_ITERATIONS is an input parameter to the program and is constant for any given run of the program. To summarize, the number of operations outside the loop is proportional to n^2, the number of operations in the loop is, at worst, proportional to n^2 (and is generally much smaller), and the number of iterations of the loop is constant. Thus the complexity of the entire graph-theoretic clustering algorithm is, at worst, proportional to n^2. The actual execution times for six of the colon shapes run on an IBM 370/158 computer are given below:

COLON	Number of Points	Relation Finding Time (s)	Cluster Time (s)
1	79	128.8	8.7
2	48	31.8	5.4
3	40	16.8	3.6
4	45	23.4	4.7
5	47	28.8	4.6
6	62	66.3	6.4

IV. Decomposition of Piecewise Linear Approximations by Clustering

A. The Use of Piecewise Linear Approximations

For a large number of data points, the complexity of computing the relation L_I and clustering can be prohibitive. We can greatly reduce the computation by adding a preprocessing phase which transforms the point representation of the boundary to a piecewise linear approximation (Pavlidis and Horowitz [16]). Thus the input data are a relatively small list of line segments instead of a big list of points.

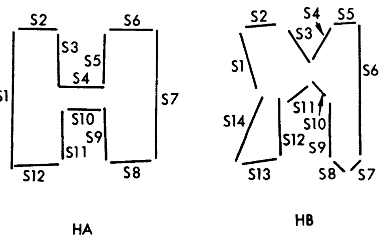


Fig. 15. A piecewise linear approximation to two letter H's.

HA

NODE	RELATED NODES
1)	1 2 3 11 12
2)	1 2 3 11 12
3)	1 2 3 4 11 12
4)	3 4 5 10
5)	4 5 6 7 8 9
6)	5 6 7 8 9
7)	5 6 7 8 9
8)	5 6 7 8 9
9)	5 6 7 8 9 10
10)	4 9 10 11
11)	1 2 3 10 11 12
12)	1 2 3 11 12

HB

NODE	RELATED NODES
1)	1 2 3 11 12 13 14
2)	1 2 3 11 12 13 14
3)	1 2 3 4 10 11 12 13 14
4)	3 4 5 6 7 8 9 10 11
5)	4 5 6 7 8 9 10
6)	4 5 6 7 8 9 10
7)	4 5 6 7 8 9
8)	4 5 6 7 8 9
9)	4 5 6 7 8 9 10
10)	3 4 5 6 7 8 9 10 11
11)	1 2 3 4 10 11 12
12)	1 2 3 11 12 13 14
13)	1 2 3 12 13 14
14)	1 2 3 12 13 14

Fig. 16. The intuitively derived relation L_I on the letter H's of Fig. 15.

To construct the relation L_I, we need a definition of when two line segments are related. Intuitively, two line segments are related if they can "see each other," and their "line of sight" is an interior line segment. Consider the two piecewise linear approximations to letter H's shown in Fig. 15. H_A represents a standard letter H, while H_B is an imperfect version. Fig. 16 shows some possible L_I relations derived intuitively on H_A and H_B. Running these relations through the clustering procedure produced the following results:

H_A
Cluster 1) 1 2 3 4 10 11 12
Cluster 2) 5 6 7 8 9

H_B
Cluster 1) 1 2 3 11 12 13 14
Cluster 2) 4 5 6 7 8 9 10

Both letter H's were decomposed into two clusters representing the left and right parts. H_B does not have a well-defined middle section, so its decomposition seems intuitively correct. In H_A, segments S_4 and S_{10} form a perceptual horizontal midsection, but segment S_4 is related to segments S_3, S_4, S_5, and S_{10}, while segment S_{10} is related to segments S_4, S_9, S_{10}, and S_{11}. Thus segments S_4 and S_{10} are not related enough to produce a cluster. In fact, they were not related enough to be initially included in cluster 1 or cluster 2, and were added later by the best fit procedure to cluster 1. (They both fit equally well in clusters 1 and 2, so they were left in the first cluster, cluster 1.)

The results of decomposing H_A and H_B show that clustering a relation defined from line segments instead of points is a feasible alternative. However, the results of decomposing H_A suggest that the clustering procedure may require some modification to produce more intuitive results. For instance, the results would be more intuitively correct if segments S_4 and S_{10} had been either added to both cluster 1 and cluster 2, omitted entirely from any clusters, or grouped to form their own cluster. The problem here is due to the fact that left-out line segments are a lot bigger than left-out points and, when added to a cluster they are only slightly related to, can cause the resulting simple part to lose its near-convexity property.

The next subsection discusses some possible definitions for determining when two boundary line segments are related.

B. Defining the Relation L_I on a Set of Line Segments

We would like two line segments to be related if, intuitively, they can "see each other" and their "line of sight" lies interior to the object being decomposed. The "see each other" concept does not suggest a unique definition for the relation L_I, but brings to mind several alternatives.

Definition 1: Segment S_i is related to segment S_j if and only if both the line segment joining the startpoint of S_i to the endpoint of S_j and the line segment joining the endpoint of S_i to the startpoint of S_j are interior line segments.

Fig. 17 illustrates this definition on H_B. Definition 1 is equivalent to saying that every point on S_i must be related to every point on S_j in order for S_i and S_j to be related. This can often be too strict a criterion. The following is a much more lenient definition.

Definition 2: Segment S_i is related to segment S_j if and only if a line segment from an endpoint of S_i to an endpoint of S_j is an interior line segment.

Fig. 18 illustrates Definition 2 on H_B. Definition 2 only requires that one point on S_i be related to one point on S_j, with the restriction that the points must be endpoints. This is clearly too lenient a criterion. We need a definition that is a compromise between Definition 1 and Definition 2.

A compromise definition might force a subsegment S_i' of S_i to be able to "see" a subsegment S_j' of S_j. This suggests the following definition.

Definition 3: Segment S_i is related to segment S_j if and only if there exists a subsegment S_i' of S_i whose length is at least Q percent of the length of S_i and a subsegment S_j' of S_j whose length is at least Q percent of the length of S_j and both the line segment joining the startpoint of S_i' to the endpoint of S_j' and the line segment joining the endpoint of S_i' to the startpoint of S_j' are interior line segments.

While definition 3 is probably the best definition from an intuitive point of view, it is much more computationally expensive than Definitions 1 and 2, since it involves determining at what points (if any) a line segment joining two segments becomes an interior line segment and how long it remains so. The computation can be reduced somewhat by specifying that the midpoints of the subsegments must coincide with the midpoints of the segments they are a part of.

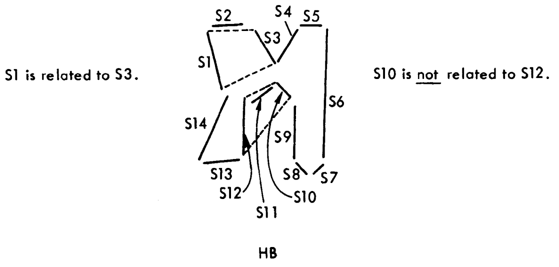


Fig. 17. Definition 1 of L_I for line-segment data.

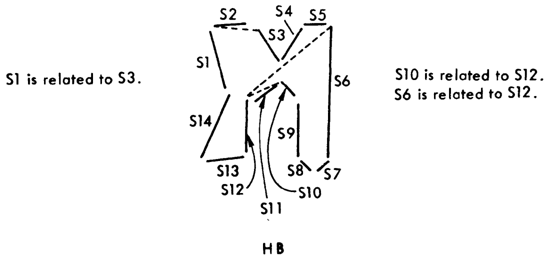


Fig. 18. Definition 2 of L_I for line-segment data.

That is, the middle Q percent of segment S_i must be able to "see" the middle Q percent of segment S_j. We can approximate this condition by checking only the midpoints. With the additional requirement that adjacent segments are always related, we obtain the following workable definition.

Definition 4: Let $\{S_1, \cdots, S_n\}$ be an ordered set of line segments comprising a piecewise linear approximation to the boundary of an object. Then (S_i, S_j) belongs to the relation L_I if and only if either 1) S_i is adjacent to S_j or 2) the line segment joining the midpoint of S_i to the midpoint of S_j is an interior line segment.

C. Results of Clustering Line Segments

The handprinted characters of Figs. 13 and 14 have been reduced to linear approximations as shown in Fig. 19. The linear approximations have an average of 12.67 line segments per letter as opposed to 56.22 points per letter in the original representation. The ratio between number of points and number of line segments is, of course, data dependent. Using the final definition of Section IV-B, the relation L_I was computed on each linear approximation and the results run through the graph-theoretic clustering procedure. Fig. 20 gives the resulting clusters for MINOVERLAP initially 0.9, MINSIZE = 3, MINCOMPACTNESS = 0.8, and MINASSOCIATION = 0.75. Each segment is marked with symbols indicating which clusters it belongs in. Segments which were added after the clustering procedure by the best fit routine have circles around their cluster symbol.

In Fig. 20, the three letter H's all have similar decompositions into two clusters representing a left part and a right part. In H1 and H2, the upper segment of the midsection of the H was assigned to the left cluster and the lower segment to the right cluster. In H3, both the upper and lower segments of the midsection were assigned (by the best fit routine) to the right

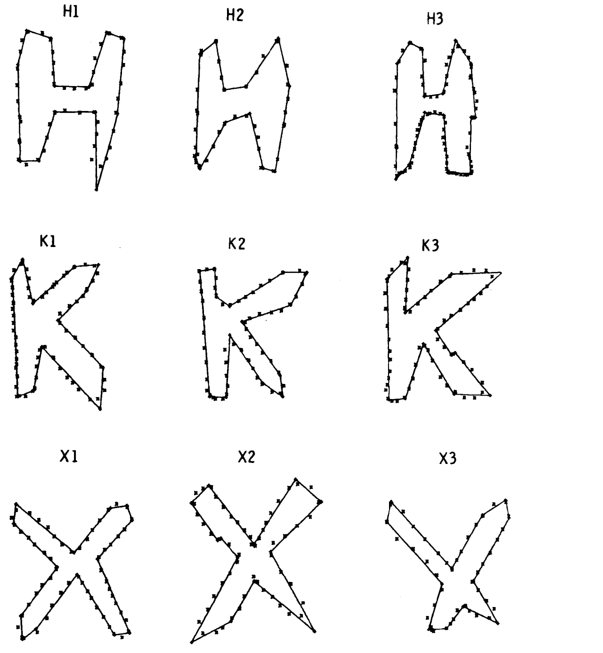


Fig. 19. The linear approximations of the nine handprinted characters of Fig. 13.

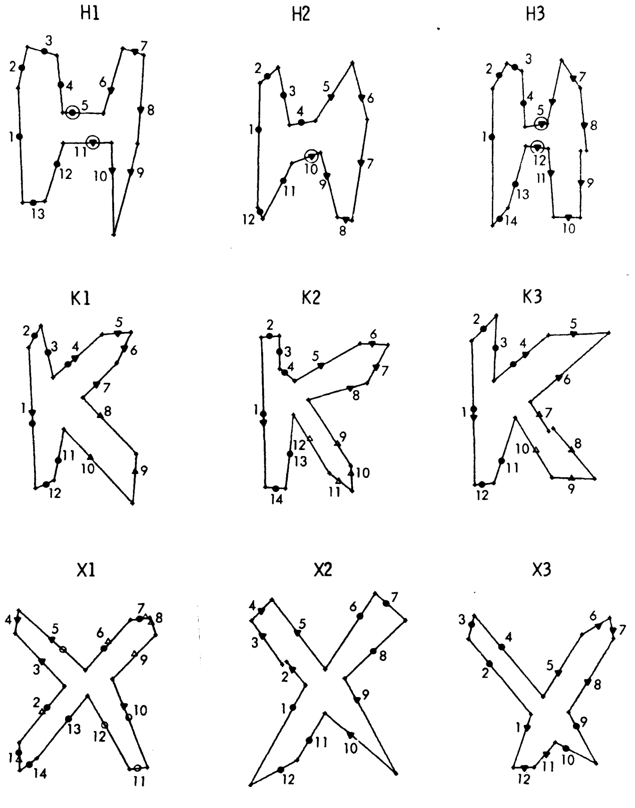


Fig. 20. The decomposition of the linear approximations of Fig. 19.

cluster. These differences might have to be resolved by a postprocessor. The three letter K's have three similar decompositions into three clusters, representing a left part, a top right part, and a bottom right part. K2 has a very good intuitive decomposition. In both K1 and K3, one segment of the top right part of the K is grouped in both the top right cluster and the left cluster. This is because this segment was related to four of the five segments in the left cluster. Again a postprocessor might be needed to resolve the problem.

Letters X2 and X3 have two similar decompositions into the two intuitive clusters. Letter X1, however, decomposed into four clusters since not enough segments were related to each other to reduce to two clusters. As with the point data, X1 needs some heuristic merging in order to look like the other two X's. Another possible approach that might lead to better results is to make L_I a labeled relation or weighted graph.

V. Summary

In this paper we have shown how to translate a boundary-point representation or piecewise linear approximation of a shape into a graph whose clusters are the possibly overlapping, simple-shape parts. We have discussed a graph-theoretic clustering procedure and illustrated its use on some example shapes. We have indicated that the complexity of the procedure of translating the shape to its graph and clustering the graph is order $0(n^2)$ (although our implementation used an algorithm of order $0(n^3)$ to translate the shape to its graph). Future work will be done in applying matching procedures to shape decompositions determined by this approach.

Acknowledgment

The authors wish to thank R. Young and D. Parikh for programming help in the implementation of the algorithms.

References

[1] J. Auguston and J. Minker, "An analysis of some graph theoretical cluster techniques," *J. Ass. Comput. Mach.*, vol. 17, pp. 571-588, Oct. 1970.

[2] H. Blum, "A transform for extracting new descriptions of shape," in *Proc. Symp. Models for the Perception of Speech and Visual Form* (Cambridge, MA: M.I.T. Press, 1964).

[3] L. S. Davis, "Understanding shape I: Angles and Sides," Comput. Sci. Center, Univ. of Maryland, TR-376, 1975.

[4] —, "Understanding shape II: Symmetry," Comput. Sci. Center, Univ. of Maryland, TR-441, 1976.

[5] —, "Shape matching using relaxation techniques," Comput. Sci. Center, Univ. of Maryland, TR-480, Sept. 1976.

[6] M. Eden, "Handwriting and pattern recognition," *IRE Trans. Inform. Theory*, vol. IT-8, pp. 160-166, 1962.

[7] H. F. Feng and T. Pavlidis, "Decomposition of polygons into simpler components: Feature generation for syntactic pattern recognition," *IEEE Trans. Comput.*, vol. C-24, pp. 636-650, June 1975.

[8] H. Freeman, "Computer processing of line drawing images," *Comput. Surveys*, vol. 6, pp. 57-97, Mar. 1974.

[9] C. Gottlieb and S. Kumar, "Semantic clustering of index terms," *J. Ass. Comput. Mach.*, vol. 15, pp. 493-513, Oct. 1968.

[10] D. Langridge, "On the computation of shape," in *Frontiers of Pattern Recognition*, S. Watanabe Ed. New York: Academic, 1972, pp. 347-365.

[11] K. Maruyama, *A Study of Visual Shape Perception*, Dep. Comput. Sci., Univ. of Illinois, Urbana, UIUCDCS-R-72-533, Oct. 1972.

[12] M. A. Meschan, and R. M. F. Farrer-Meschan, *An Atlas of Normal Radiographic Anatomy*. Philadelphia, PA: Saunders, 1959.

[13] J. F. O'Callaghan, "Recovery of perceptual shape organization from simple closed boundaries," *Comput. Graphics Image Processing*, vol. 3, pp. 300-312. Dec. 1974.

[14] T. Pavlidis, "Analysis of set patterns," *Pattern Recognition*, vol. 1, pp. 165-178, Nov. 1968.

[15] —, "Representation of figures by labelled graphs," *Pattern Recognition*, vol. 4, pp. 5-17, 1972.
[16] T. Pavlidis and S. L. Horowitz, "Segmentation of plane curves," *IEEE Trans. Comput.*, vol. C-23, pp. 860-870, Aug. 1974.
[17] T. Pavlidis and F. Ali, "Computer recognition of handwritten numerals by polygonal approximations," *IEEE Trans. Syst. Man. and Cybern.*, vol. SMC-5, pp. 610-614, Nov. 1975.
[18] T. Pavlidis, "A review of algorithms for shape analysis," Dep. Elec. Eng. Comput. Sci., Princeton Univ., Tech. Rep. 218, Sept. 1976.
[19] B. Rosenberg, "The analysis of convex blobs," *Comput. Graphics Image Processing*, vol. 1, pp. 183-192, 1972.
[20] —, "Computing dominant points on simple shapes," *Int. J. Man-Mach. Studies*, vol. 6, pp. 1-12, 1975.

Linda G. Shapiro (S'74-M'74-A'77) was born in Chicago, IL, in 1949. She received the B.S. degree in mathematics from the University of Illinois, Urbana, in 1970, and the M.S. and Ph.D. degrees in computer science from the University of Iowa, Iowa City, in 1972 and 1974, respectively.

She has been an Assistant Professor of Computer Science at Kansas State University, Manhattan, since 1974. Her research interests include pattern recognition, scene analysis, computer graphics, data structures, and programming languages. She is currently working on an undergraduate textbook on data structures with R. Baron.

Dr. Shapiro is a member of the Association for Computing Machinery and the Pattern Recognition Society.

Robert M. Haralick (S'62-S'67-M'69-SM'76) was born in Brooklyn, NY, on September 30, 1943. He received the B.S. degree from the University of Kansas, Lawrence, in 1966.

He has worked with Autonetics and IBM. In 1965 he worked for the Center for Research, University of Kansas, as a Research Engineer and, in 1969, when he completed his Ph.D. at the University of Kansas, he joined the faculty of the Electrical Engineering Department where he is now a Professor. He has done research in pattern recognition, multiimage processing, remote sensing, texture analysis, data compression, clustering, artificial intelligence, and general systems theory. He has been responsible for the development of the Kansas Digital Image Data System (KANDIDATS), a multiimage processing package which runs on a minicomputer system.

Dr. Haralick is a member of the Association for Computing Machinery, Sigma Xi, the Pattern Recognition Society, and the Society for General Systems Research.

A Description Method of Handprinted Chinese Characters

TAKESHI AGUI AND HIROSHI NAGAHASHI

Abstract—**A description method of handprinted Chinese characters is presented. In the method, a Chinese character is composed of some partial patterns which are constructed using the concatenate relation, cross relation, and near relation. The relations of relative location among partial patterns are used for categorization of the partial patterns. A Chinese character is expressed from the results of categorization.**

Index Terms—**Categorization of patterns, geometrical feature points, handprinted Chinese characters, matrix for relative locations of blocks, relative location.**

I. Introduction

It is difficult to recognize Chinese characters because of the variety of the characters compared with numerals and other letters. Several investigations concerned with computer recognition of printed and handwritten Chinese characters have been made in recent years [1]-[4].

Manuscript received October 17, 1977; revised April 11, 1978.
The authors are with the Imaging Science and Engineering Laboratory, Tokyo Institute of Technology, Midori-ku, Yokohama, Japan.

A Chinese character is drawn by some simple strokes. Rankin analyzed the characters from the viewpoint of linguistic aspects and reported a syntactic method of modeling of the characters [5]. Moreover, some processings of descriptions in which concatenation of strokes and their arrangements are used also have been reported [6].

In this paper we report a descriptive method of handwritten Chinese characters, defining relative locations among their partial patterns that are constructed stably.

II. Representation of Relation

Let a handwritten Chinese character be described by a 32×32 binary matrix and be thinned. Geometrical feature points are extracted from the information of nodes and angles between strokes. The direction of a stroke started from a feature point is quantized to one of the eight directions, and from the feature points and their directions, the character pattern is decomposed with four kinds of lines, as shown in Fig. 1. Then, defining a concatenate relation among them,