SLAC-149
STAN-CS-72-277
UC-32
(MISC)

REGION BOUNDARIES ON A TRIANGULAR GRID

Charles T. Zahn

STANFORD LINEAR ACCELERATOR CENTER
STANFORD UNIVERSITY
Stanford, California 94305

PREPARED FOR THE U. S. ATOMIC ENERGY
COMMISSION UNDER CONTRACT NO. AT(04-3)-515
AND NSF GRANT NO. GJ687

May 1972

Printed in the United States of America. Available from National Technical Information Service, U. S. Department of Commerce, 5285 Port Royal Road, Springfield, Virginia 22151.
Price: Printed Copy $3.00; microfiche $0.95.

ABSTRACT

A simple graph model is developed for binary digital pictures on a triangular grid leading to consistent and intuitive definitions of connectivity and region boundaries as well as fast memory-efficient algorithms for computing boundaries and the "insidedness" tree. Boundary encodings are extremely compact and can be smoothed using a discrete implementation of the minimum-perimeter polygon methods of Montanari (JACM April 1970 and CACM January 1970) and Sklansky et al. (IEEE-TC March 1972). Details of implementation are briefly discussed and attempts to generalize the model to nontriangular grids explains the well-known "anomoly" associated with connectivity on the square grid.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Introduction

Much picture processing and pattern recognition research has been concerned with inputs encoded like Fig. 1. Following Rosenfeld [1, p. 2] we call these binary digital pictures. The present paper concerns several problems encountered in transforming the grid format into a more structured format suitable for describing and/or recognizing the shapes embodied in the picture. The first problem is that of defining a concept of connectivity among points of the digital picture so that the resulting connected components correspond well with one's intuitive sense of connectivity and separateness. This topic is treated at some length by Rosenfeld [3]. Having defined connectivity there is the problem of how to construct an efficient algorithm for labelling the separate connected components (or equivalently detecting what and where they are). Montanari [11] among others has suggested a method for this problem. Each connected component is separated from adjacent connected components of opposite color by boundary curves whose precise definition and computation is the third problem. Some have preferred to construct border curves through extremal points of a connected component but we feel boundaries which fall between adjacent pairs of opposite colored points have more intuitive appeal. Rosenfeld [3] and Zahn [10] represent different approaches to boundary curve construction, the latter being the forerunner of the approach developed here. Finally, it is of some interest to know which connected components are enclosed by which others and this information is nicely represented by an "insidedness tree" whose vertices may be components or boundary curves. Algorithms for constructing this tree are given by Montanari [4] and Buneman [5]. See Rosenfeld [1, pp. 135-139] or [2, p. 161] for brief discussions of these problems and excellent bibliographies.

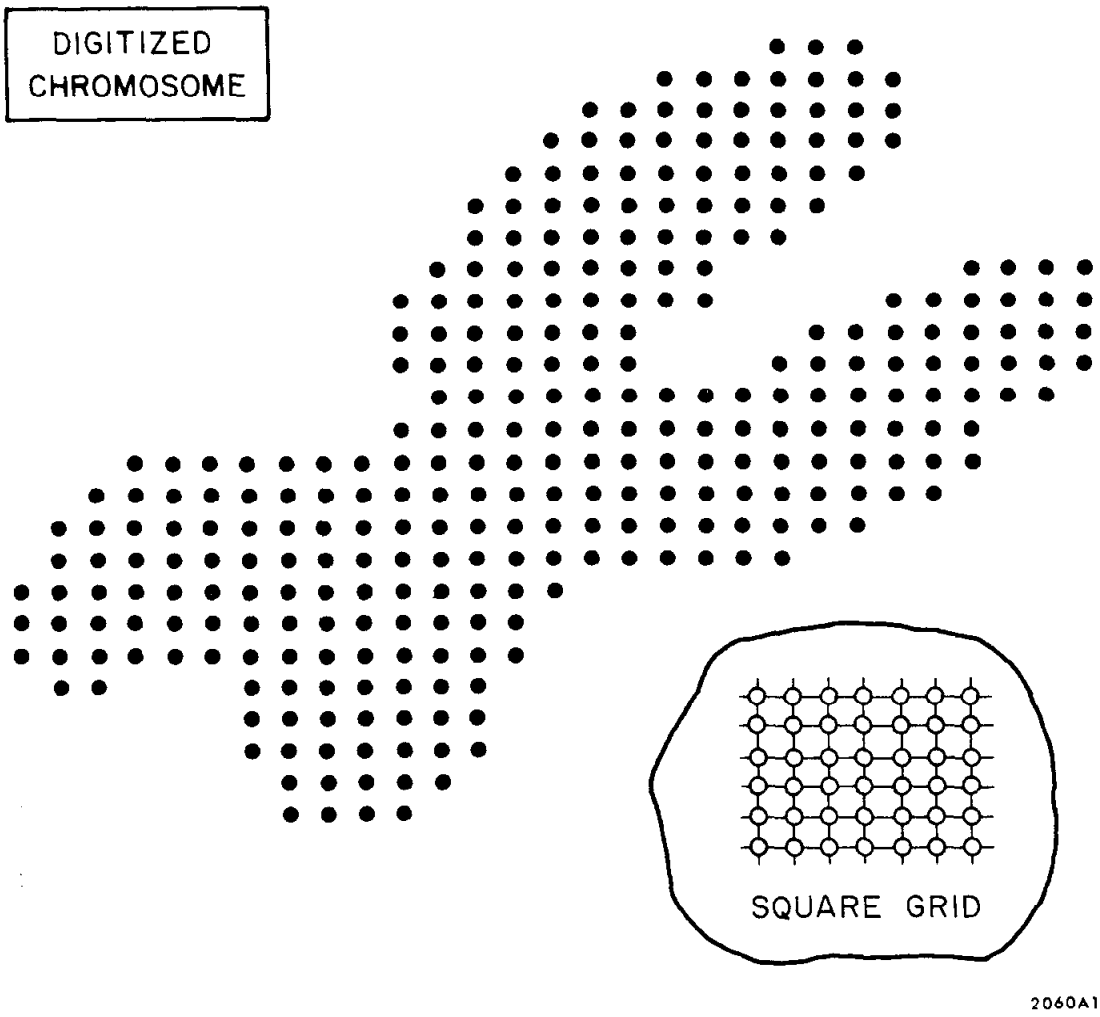

FIG. 1--Binary digital picture on square grid.

All methods cited above concern digital pictures on a square grid as exemplified by Fig. 1. In this paper we develop a theory and algorithms for the above problems assuming digital pictures on a triangular grid as exemplifed in Fig. 2. The triangular grid is also known as "hexagonal" [6] or "rhombic" [7]. Attempts to arrive at a consistent definition of connectivity on the square grid have encountered an "anomoly" which forces 8-connectivity for one color and 4-connectivity for the other [3]. This has the effect of treating two connected shapes differently even if they differ only in color. The anomoly runs somewhat deeper and has been independently acknowledged by a number of pattern recognizers. Golay [6] has emphasized the isotropic nature of the triangular grid and developed paralled picture processing operators for this grid. Gray [8] also shows an appreciation for the elegance of this grid. In earlier work [9] we recognized several important ways in which the triangular grid was at an advantage but until now we had not carried these hints to their logical conclusion.

In the following sections we develop a simple graph model for binary digital pictures on a triangular grid leading to consistent and intuitive definitions of connectivity and region boundaries and fast memory-efficient algorithms for computing boundaries and the insidedness tree. The boundary encodings can be smoothed using a discrete implementation of the minimum perimeter polygon methods of Montanari [4], [11] and Sklansky et al. [12]. The resulting data-structure represents connected regions (components) by their boundary curves so that shape comparisons can employ techniques developed for curves (e.g., [13], [14], [16] to cite a few).

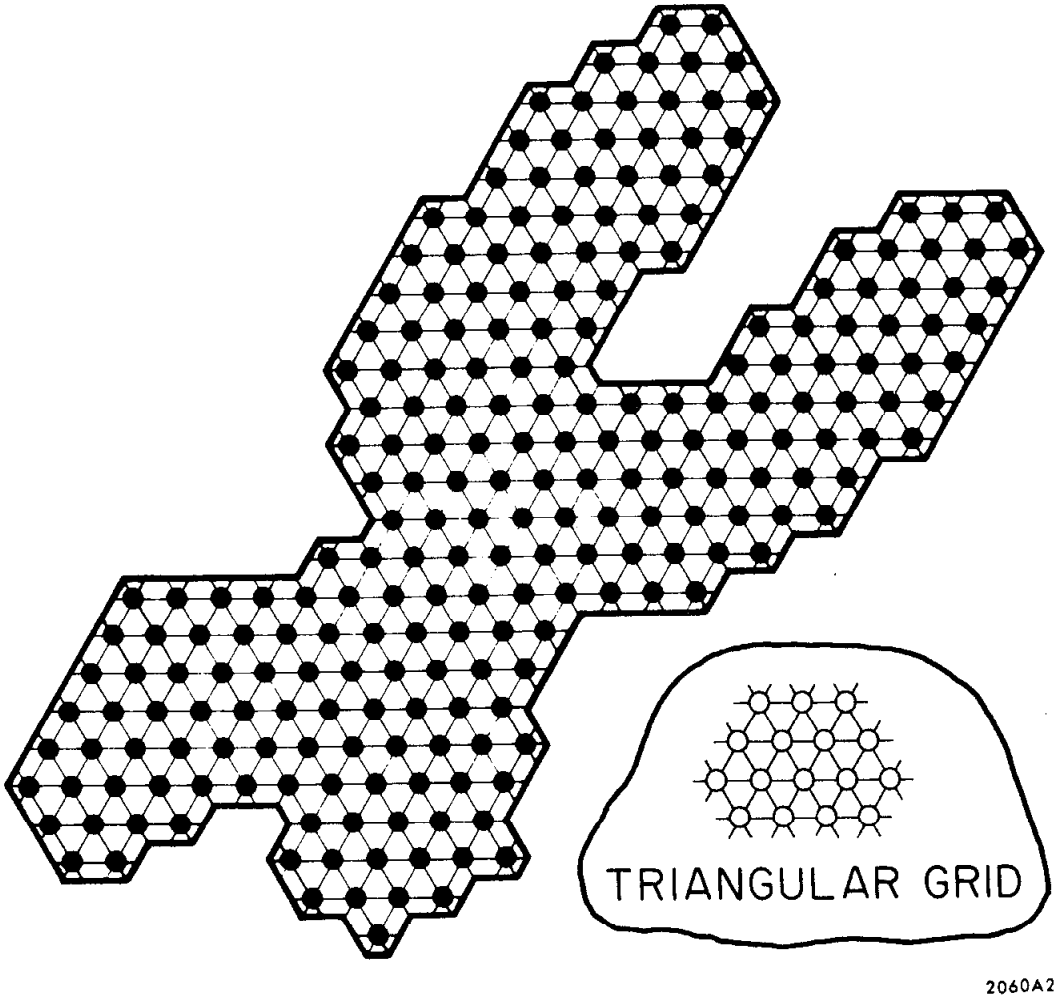

FIG. 2--Binary digital picture on triangular grid.

Triangular Picture Graphs

A triangular grid is an infinite planar graph each of whose faces is an equilateral triangle and each of whose vertices is incident to exactly six edges. A triangular picture graph is a triangular grid whose vertices are labelled 'black' or 'white' so that the set of black vertices is bounded. Figure 4 depicts a portion of a triangular picture graph as seen through a rectangular window; it is a discrete digitizing of the regions in Fig. 3. Although in practice all such pictures will be viewed through a bounded window, it is convenient for the theory which follows to consider infinite triangular picture graphs.

The connectivity graph C(T) of a triangular picture graph T is the subgraph of T consisting of all vertices of T and just those edges (level edges) whose end vertices have the same label. The heavy edges in Fig. 5 identify the connectivity graph for Fig. 4. The graph C(T) can be decomposed into connected components $\{C_K\}$ each bearing a label 'black' or 'white' inherited from its vertices. For each C_K let the connectivity region R_K be defined as a region of the plane containing all the edges in C_K. If a level face is one with only level edges then the faces included in R_K are all level. The level faces are shaded in Fig. 5.

A contrast edge of T is one with differently labelled end vertices and a contrast face is one with at least one contrast edge. The contrast edges are light in Fig. 5. It is easily seen that a contrast face has exactly two contrast edges. Each contrast face contains a unique contour segment which is a directed line segment joining the midpoints of the two contrast edges in such a way that the black vertex or vertices of the face lie to the right of the directed line. The contour segments for all (unshaded) contrast faces are depicted in Fig. 5. Some motivation for the term 'contour' is appropriate at this point. It is

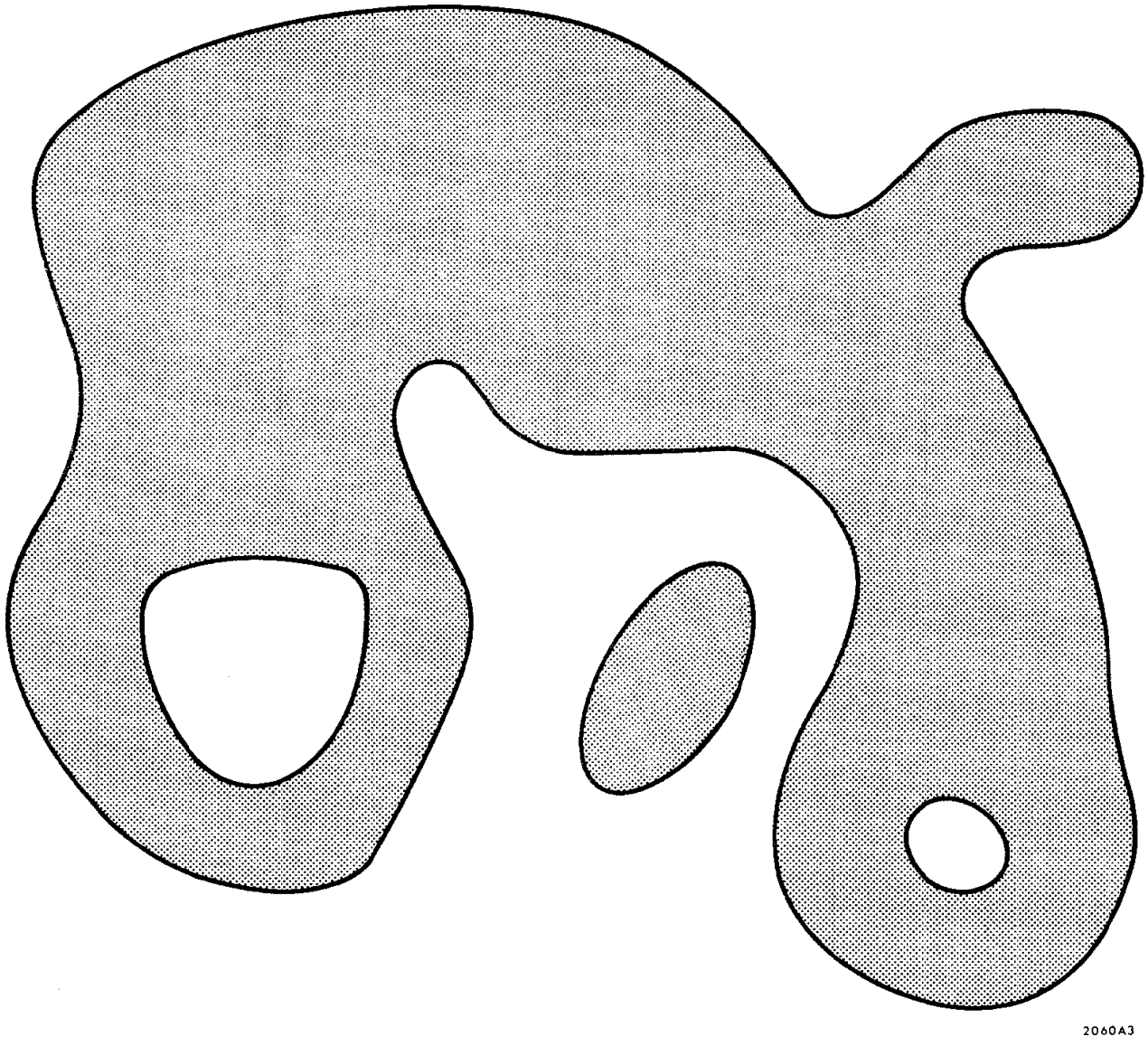

FIG. 3--Black and white planar regions with smooth boundaries.

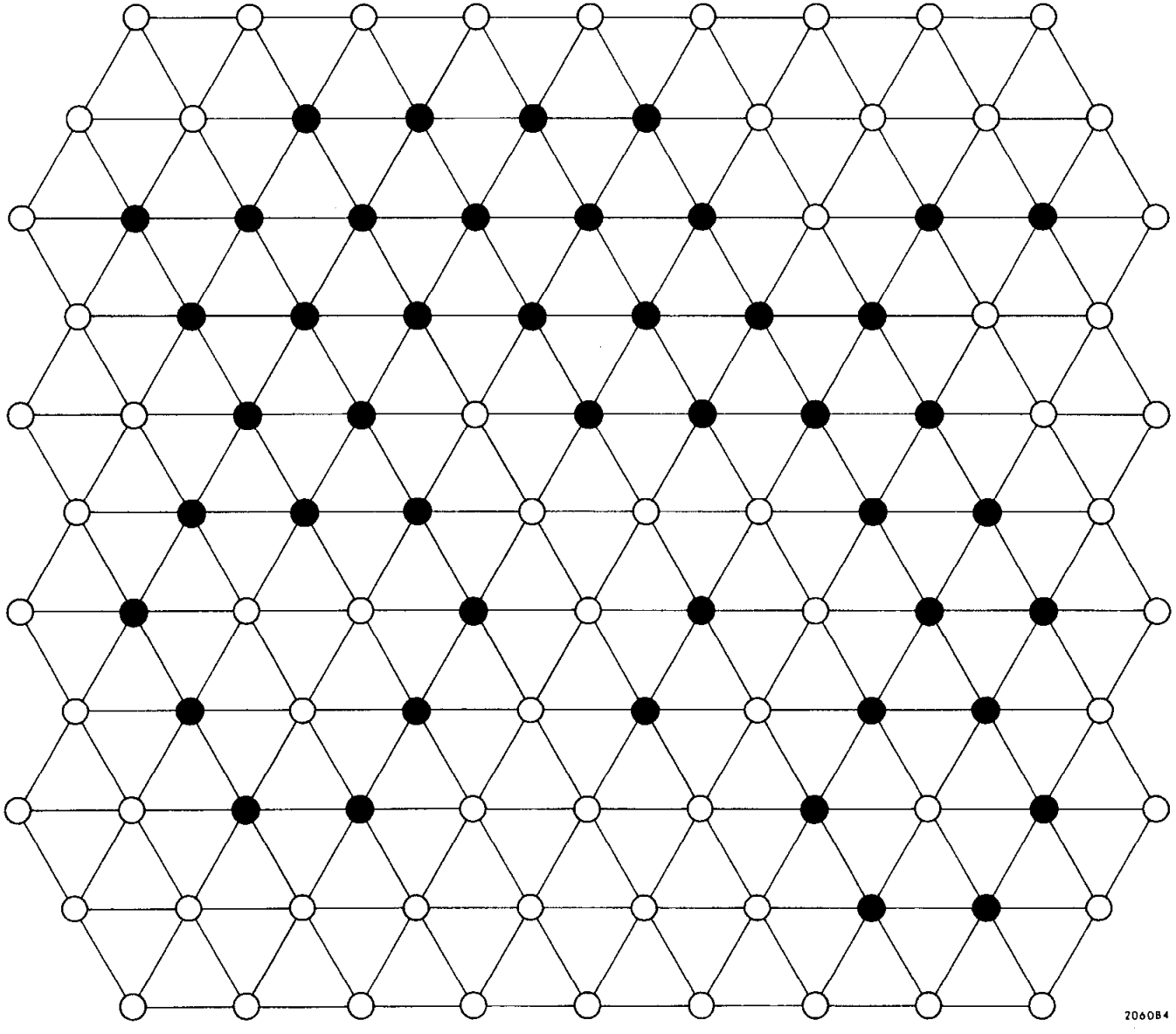

FIG. 4--Triangular picture graph derived from Fig. 3.

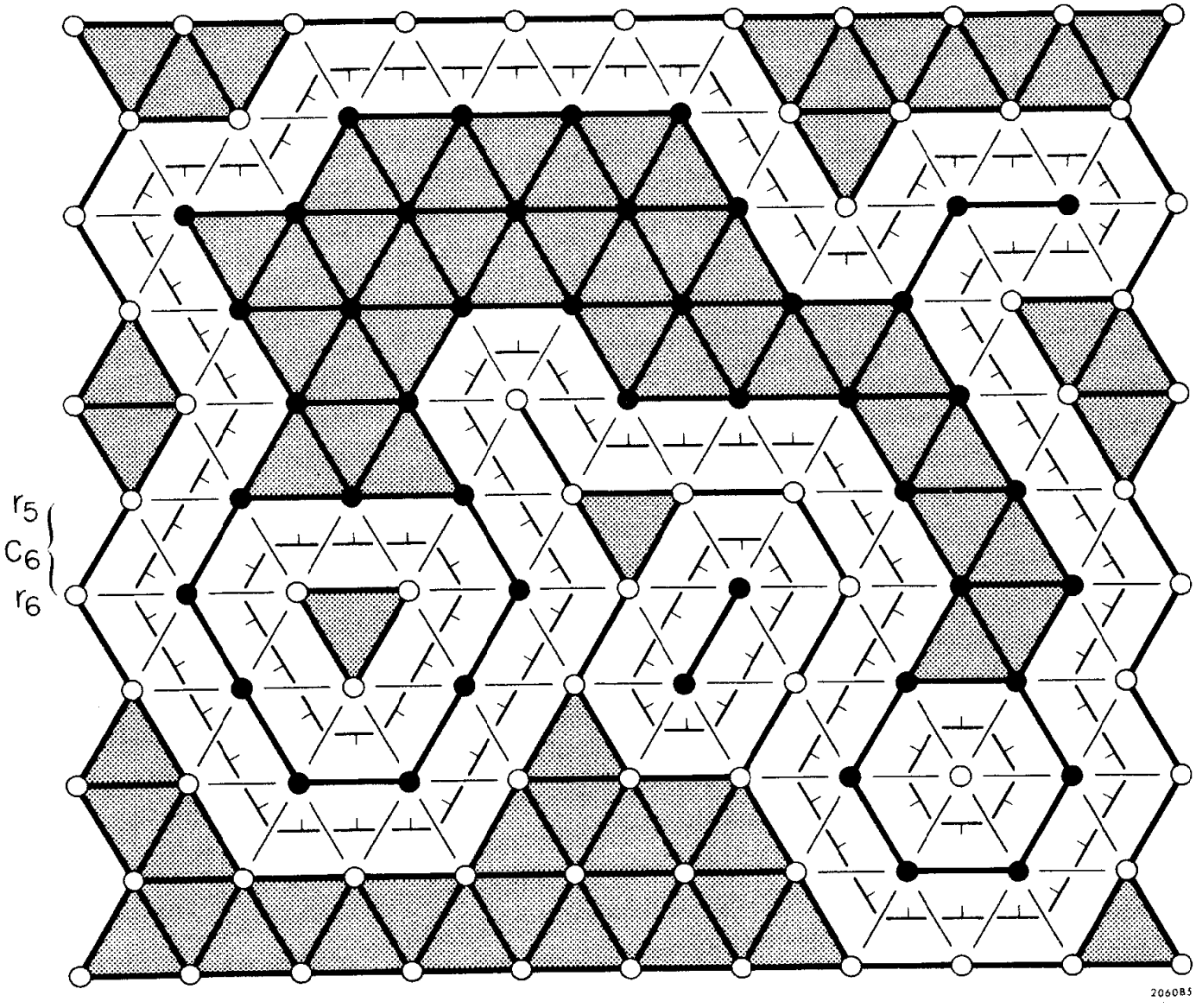

FIG. 5--Connectivity graph, contrast faces and contour segments for the triangular picture graph of Fig. 4.

possible to construct a continuous piecewise-linear function F_T defined on the plane so that $F_T(v) = 1$ at black vertices and $F_T(v) = 0$ at white vertices. This is accomplished very simply by defining F_T on each separate triangular face of T to be the unique linear function whose values at the three vertices of the face are as prescribed by the vertex labels. The contour set of F_T at value 1/2 (i.e., points p with $F_T(p) = 1/2$) is then a family of mutually nonintersecting simple closed curves which separate black and white areas of the plane. The union of all contour segments defined above constitutes the contour set for F_T.

We would like to define a graph based on T which corresponds to the contour curves of F_T. It is convenient to use the dual graph for this purpose. Every planar graph G has a dual graph D(G) constructed by making a vertex in D(G) for each face of G and connecting two vertices of D(G) by an edge if the corresponding faces of G share an edge in G. There is thus a one-to-one correspondence between the edge sets of G and D(G) and between the face set of G and the vertex set of D(G). Now let the dual graph D(T) of a triangular picture graph T inherit labelling structure from T as follows: Appropriate vertices of D(T) will be designated as contrast vertices and labelled with the contour segment from the corresponding contrast face in T. Furthermore, edges of D(T) will be directed from vertex v_1 to vertex v_2 if the corresponding edge in T is a contrast edge which has black on the right when crossing it from face f_1 to face f_2 (the correspondents of v_1 and v_2 respectively). The boundary graph B(T) is the subgraph of the labelled D(T) consisting of contrast vertices and directed edges. The appropriateness of this definition is demonstrated by the following theorem which shows the consistency between the connectivity graph C(T) and the boundary graph B(T).

Theorem 1

If $C(T) = \cup C_K$ is the connectivity graph of the triangular picture graph T with connected components C_K defining connectivity regions R_K and $B(T) = \cup B_i$ is the boundary graph of T with connected components B_i then

(a) Each B_i is a directed circuit and the set of contour segments labelling vertices of B_i forms a simple closed curve γ_i. The edge directions in B_i are compatible with the contour segment directions in the sense that when edge (v_1, v_2) belongs to B_i and $s_K = (p_K, q_K)$ is the contour segment label on vertex v_K, then $q_1 = p_2$.

(b) The curves $\{\gamma_i\}$ form a mutually nonintersecting family of simple closed curves which partitions the remainder of the plane into connected regions ρ_j each containing as a subset exactly one of the connectivity regions R_K.

Proof

(a) A vertex v of B(T) corresponds to a contrast face f in T and f has exactly two contrast edges e_1 and e_2. When viewed from the center of f one of these edges is a left-right pair (b, w) and the other a (w, b) pair. Suppose e_1 is the (b, w) and e_2 the (w, b). Then e_1 corresponds to an edge of B(T) directed into v and e_2 to an edge directed out from v. Hence any vertex of B(T) has exactly one in-directed and one out-directed edge; hence, the B_i are directed circuits. If directed edge (v_1, v_2) belongs to B(T) then T contains two adjacent contrast faces f_1 and f_2 with a common contrast edge e_{12} whose labelling is (w, b) as viewed from f_1 and (b, w) viewed from f_2. If $s_K = (p_K, q_K)$ is the contour segment for f_K then clearly q_1 and p_2 are both the midpoint of e_{12}. Each edge of B(T) indicates that two contour segments from

adjacent faces of T have a common endpoint and since each B_i is a directed circuit the set of contour segments labelling its vertices must form a closed curve γ_i. The γ_i are simple because a contrast face has one unique contour segment.

(b) It is clear that the family of closed curves $\{\gamma_i\}$ partitions the remainder of the plane into a set of maximal connected regions ρ_j. Each connectivity region R_K is entirely contained in one region $\rho_{j(K)}$ since otherwise a curve γ_i would intersect R_K and this is impossible since R_K is made up entirely of level edges and faces. If two vertices v_1 and v_2 are both in the region ρ_j then they can be connected by a curve δ_{12} lying entirely in ρ_j. It is an easy exercise to replace δ_{12} by a curve δ'_{12} which lies entirely within edges of T as well as being in ρ_j. The curve δ'_{12} is obtained by replacing each face-crossing subcurve of δ_{12} by a portion of the face boundary. In the case of a contrast face the subcurve lies within a triangular or trapezoidal subface but it can still be pulled over to the boundary of the face in an obvious way. The curve δ'_{12} can be transformed to Δ_{12} which consists of a sequence of edges of T simply by eliminating redundant loops in δ'_{12}. The curve Δ_{12} joins v_1 to v_2 inside ρ_j and consists of edges of T. Now Δ_{12} cannot contain a contrast edge for then a point on some γ_i would be inside ρ_j. Hence, Δ_{12} is a level path in T and so v_1 and v_2 belong to the same R_K. This means that two distinct R_K cannot be in the same region ρ_j. We have thus shown that there is a natural one-to-one correspondence between the connected regions ρ_K determined by the curves $\{\gamma_i\}$ and the connectivity regions R_K determined by the connected components C_K of C(T).

Endproof

Theorem 1 shows that the regions ρ_j and R_K come in natural pairs ($R_K \subseteq \rho_K$) so that there is no reason to distinguish the indexing symbols. The connection between ρ_K and R_K is actually stronger than what we have stated above. In fact, the part of region ρ_K not in R_K is restricted to narrow bands near the boundary curves for ρ_K, where narrow means no wider than one half the length of an edge of T. The difference in area between R_K and ρ_K is thus approximately a linear function of the boundary perimeter of ρ_K. With these remarks as justification we shall now restrict our attention to the connected regions ρ_K and no longer treat the connectivity regions R_K directly. Figure 6 depicts the relationships among graphs regions and curves in T.

There is some more structure relating the curves $\{\gamma_i\}$ and the regions $\{\rho_K\}$ which can be captured very naturally in a tree-structure. Each ρ_K except ρ_∞ (the sole unbounded region) has a unique outer boundary curve $\gamma_{i(K)}$ so that there is a one-to-one correspondence between the bounded ρ_K and the curves of γ_i. Henceforth, we use the same indexing symbol and assume γ_K is the outer boundary curve for region ρ_K. In addition to its outer boundary each ρ_K (even ρ_∞) has zero or more inner boundary curves separating ρ_K from its holes. We define the insidedness tree I(T) for a triangular picture graph T as a directed tree rooted at ρ_∞ having vertices ρ_K and edges corresponding to boundary curves γ_K. The pair (ρ_K, ρ_ℓ) is a directed edge of I(T) if the outer boundary curve γ_K for region ρ_K is also one of the inner boundary curves for ρ_ℓ. In this case the edge (ρ_K, ρ_ℓ) carries a label γ_K. An equivalent way to phrase the condition is that region ρ_K is a subset of one of the holes in region ρ_ℓ. The insidedness tree for Fig. 4 is shown in Fig. 7 where regions have been appropriately labelled as white or black and boundary curves are directed as clockwise or counter clockwise. We point out that region labels alternate as one

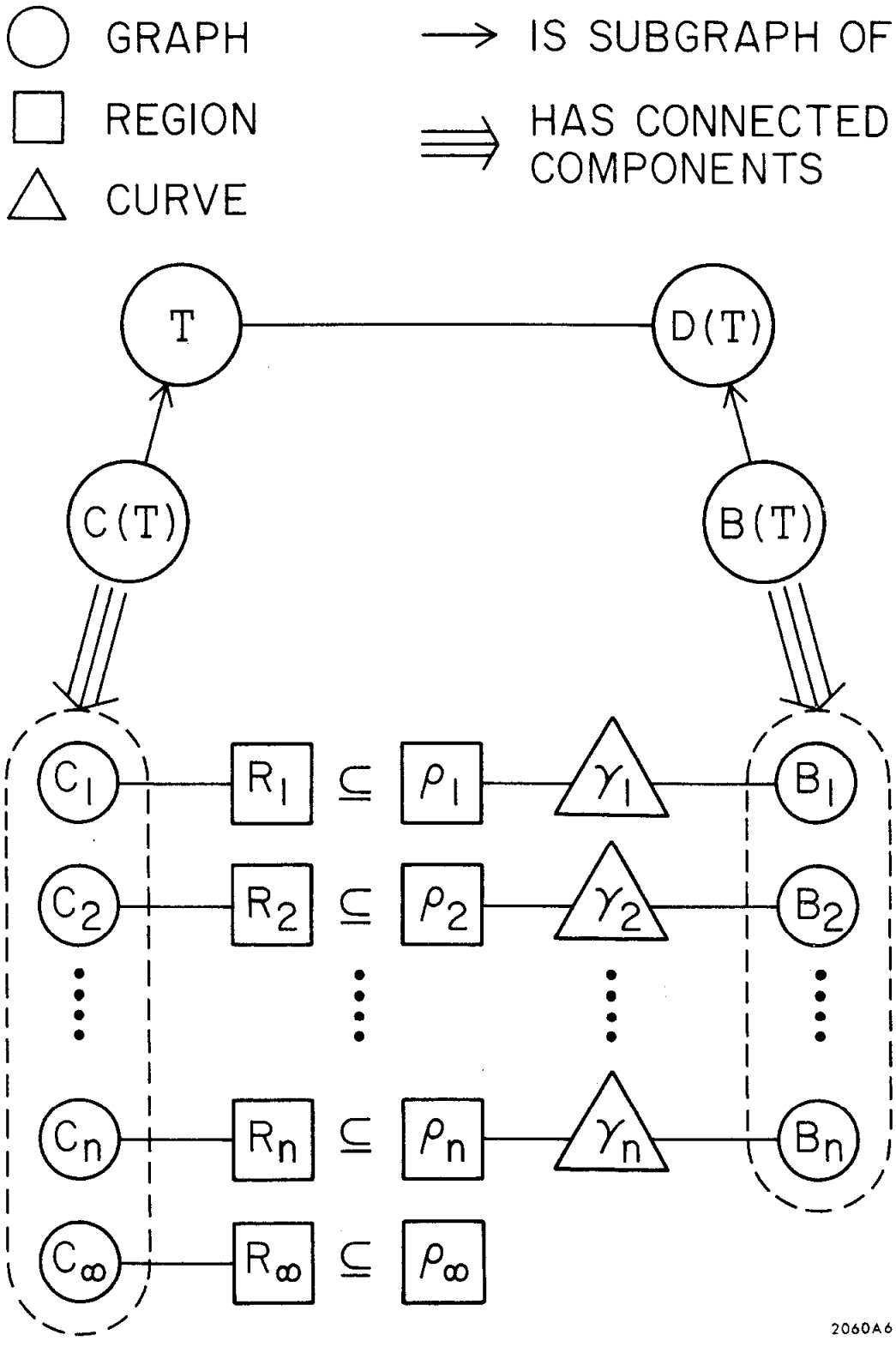

FIG. 6--General relationship among triangular picture graph T, dual graph D(T), connectivity graph C(T), boundary graph B(T), connectivity regions R_K, boundary curves γ_K and planar regions ρ_K formed by boundary curves γ_K.

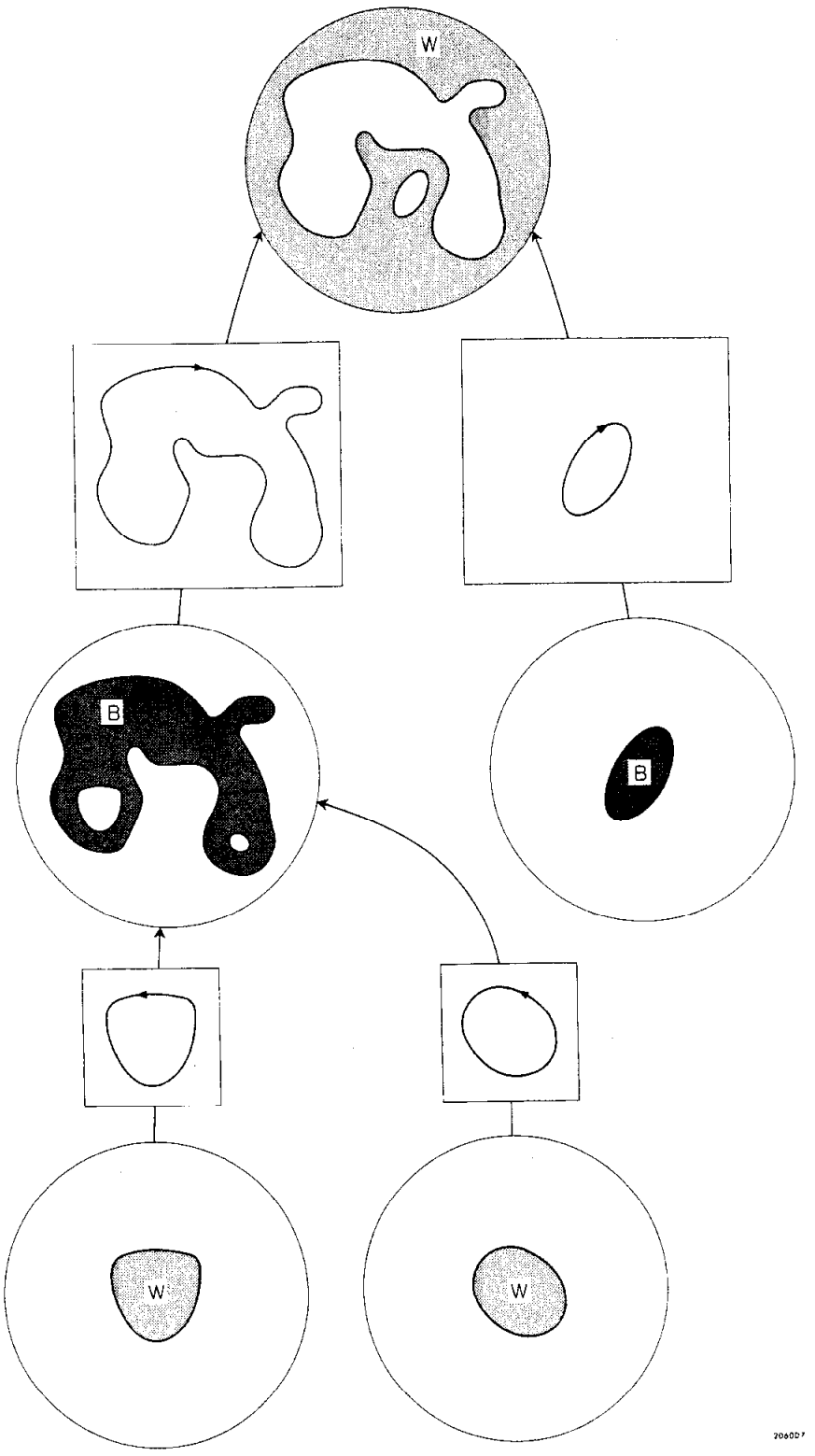

FIG. 7--Insidedness tree of regions and boundaries for the picture in Fig. 3.

passes up or down the tree and clockwise curves are just above black regions and vice versa for counter clockwise curves. The importance of this simple relationship is that in the next section we will show how to compute the boundary curves γ_K for a picture T without directly computing the connectivity graph C(T) and so all information about the shape of a region ρ_K and the label (b or w) of region R_K must be inferred from the curves $\{\gamma_K\}$.

Computing the Tree of Boundary Curves

Let a TV-scan triangular picture graph T* be a black-white labelling of a partial triangular grid formed by staggered rows of vertices in a rectangular window with the restriction that all vertex labels are white along the outer border adjacent to the unbounded region of the plane. Figure 4 is a TV-scan triangular picture graph. There is no loss of generality here since T* can always be extended to an (infinite) triangular picture graph T in the obvious way and any T can be transformed to a T* without loss of information because the set of black vertices is bounded. The only difference between T and T* for the previous theory is that C_∞ and R_∞ are infinite in T while they are connected to the outer border in T*. Concerning curves $\{\gamma_K\}$ and regions $\{\rho_K\}$ and ρ_∞ there is no difference at all.

Now we describe a method for computing the boundary curves $\{\gamma_K\}$ and the insidedness tree I(T*) for a TV-scan triangular picture graph T*. Let r_i denote the i^{th} row of vertices in T* reading from top to bottom with i in the range [0, n]; we shall usually think of each row r_i as a sequence of edges of T*. The sequence of triangular faces between r_{i-1} and r_i (ordered from left to right) we call the corridor C_i. The corridor sequence σ_i is obtained from C_i by replacing each face by its "undirected" contour segment value ($\cdot$ = level, / , \ , ——) and then eliminating entries representing level faces. Figure 8 depicts a portion of

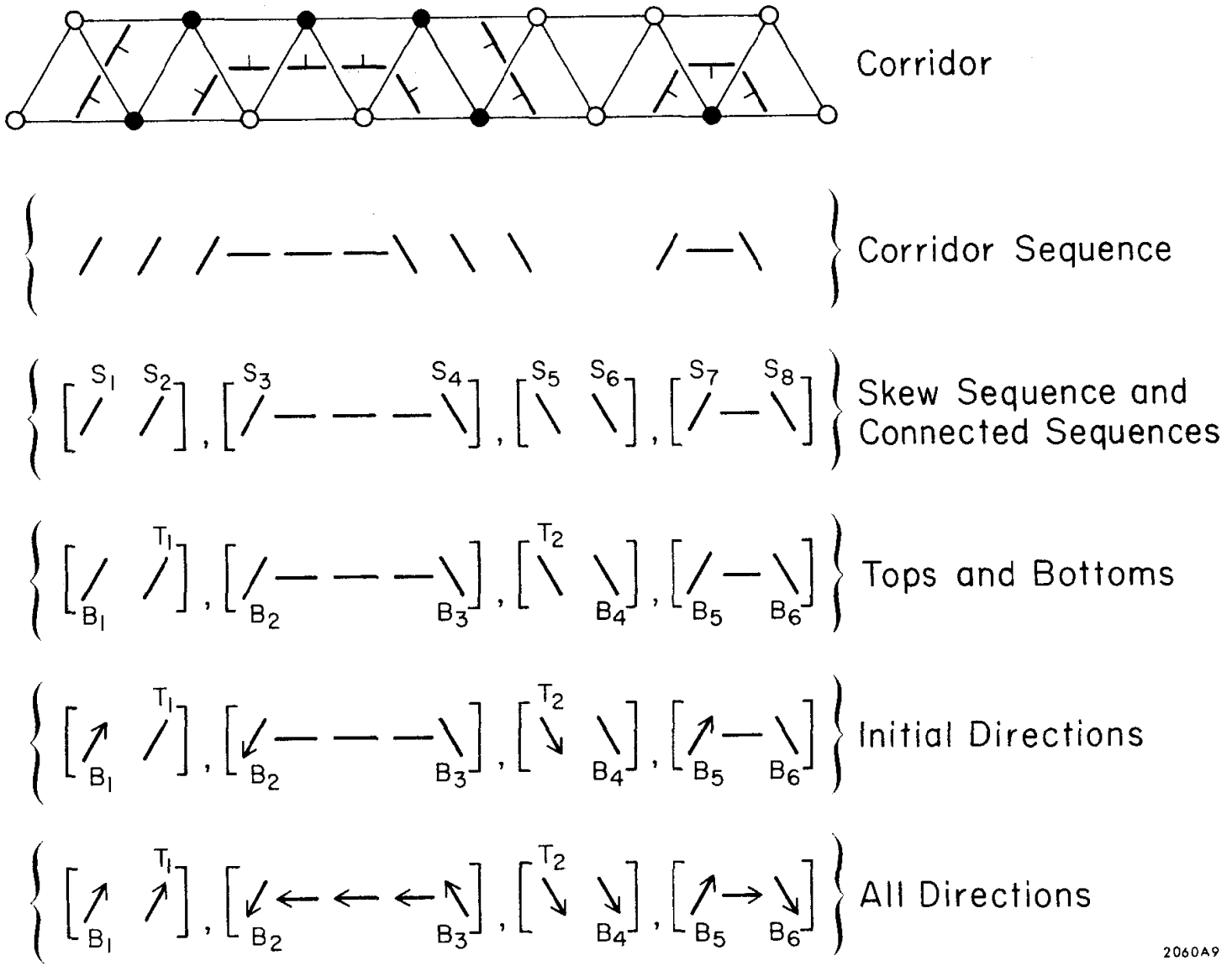

FIG. 8--Analysis of a corridor sequence.

corridor C_6 from Fig. 5 and demonstrates pictorially much of what we attempt to verbalize in the next few paragraphs. Let the skew sequence S_i be obtained from σ_i by deleting all flat (—) entries. Then $S_i = (s_{i,1}, s_{i,2}, \cdots, s_{i,2m_i})$ can be broken into m_i adjacent pairs where each pair along with the (possibly zero) flat segments in between constitute a connected piece of contour intersecting the corridor. Within the skew sequence some segments touch the top row r_{i-1} and the others touch the bottom row r_i and we call these two subsequences the top sequence τ_i and the bottom sequence β_i. The segments of τ_i are of the form (S_{odd}, \) or (S_{even}, /) and the segments of β_i are of the form (S_{odd}, /) or (S_{even}, \). This is because each connected subsequence $(s_{i,2K-1}, \cdots, s_{i,2K})$ of the full corridor sequence σ_i touches the adjacent rows at the leftmost endpoint of segment $s_{i,2K-1}$ and the rightmost endpoint of segment $s_{i,2K}$. The sequences τ_i and β_i are indexed from left to right as subsequences of S_i. Table 1 gives a BNF grammar for valid corridor sequences and exhibits the natural parse into connected subsequences, some of which cross the corridor and others of which simply dip in and out.

The following theorem shows how to reconstruct the directions for contour segments and how to relate connected subsequences from adjacent corridors. It provides the backbone of an algorithm for constructing the boundary curves $\{\gamma_K\}$ from the corridor sequences of a picture graph T*.

Theorem 2

A contour segment is directed upward (/ or \ ) if it has an odd index in τ_i or β_i and downward (/ or \ ) if it has an even index in τ_i or β_i. If C_i and C_{i+1} are adjacent corridors with $\beta_i = (b_1, b_2, \cdots, b_p)$ and $\tau_{i+1} = (t_i, t_2, \cdots, t_q)$ then p=q and each pair of segments (b_j, t_j) intersects at the midpoint of a contrast edge along row r_i.

Table 1

Grammar for Valid Corridor Sequences

⟨even-flat⟩ ← ⟨empty⟩ | ⟨odd-flat⟩ —

⟨odd-flat⟩ ← ⟨even-flat⟩ —

⟨bottom-bottom⟩ ← / ⟨odd-flat⟩ \

⟨bottom-top⟩ ← / ⟨even-flat⟩ /

⟨top-top⟩ ← \ ⟨odd-flat⟩ /

⟨top-bottom⟩ ← \ ⟨even-flat⟩ \

⟨corridor-crossing⟩ ← ⟨bottom-top⟩ | ⟨top-bottom⟩

⟨in-out⟩ ← ⟨bottom-bottom⟩ | ⟨top-top⟩

⟨part-sequence⟩ ← ⟨corridor-sequence⟩ ⟨corridor-crossing⟩

| ⟨part-sequence⟩ ⟨in-out⟩

⟨corridor-sequence⟩ ← ⟨empty⟩ ⟨corridor-sequence⟩ ⟨in-out⟩

| ⟨part-sequence⟩ ⟨corridor-crossing⟩

Proof

Consider the row r_i between adjacent corridors C_i and C_{i+1} as a sequence of edges of T*. Since the leftmost and rightmost vertices of r_i are on the border of T* they are labelled w. Eliminating the level edges of r_i we obtain a left-right sequence $(e_1, e_2, \cdots, e_{2u})$ of contrast edges such that e_j is labelled (w, b) or (b, w) according as j is odd or even. Now the bottom sequence β_i for corridor C_i contains exactly those contour segments of σ_i which touch (i.e., have an endpoint on) row r_i. Each such segment b_j must therefore have an endpoint which is the midpoint of some contrast edge in row r_i. On the other hand, each contrast edge e_j on row r_i is adjacent to a contrast face in corridor C_i and therefore its midpoint m_j is the endpoint of some segment b_j in the bottom sequence β_i. We have shown that $p=2u$ and that m_j the midpoint of contrast edge e_j is an endpoint of segment b_j in the bottom sequence for corridor C_i. An analogous argument leads to the conclusion that $q=2u$ and m_j is an endpoint of segment t_j in the top sequence for corridor C_{i+1}. The segment directions for b_j and t_j depend on the labelling of contrast edge e_j and since these labellings alternate (w, b), (b, w), $\cdots$ the first part of the theorem follows immediately.

Endproof

Figure 8 indicates that once the correct segment direction has been selected for the initial segment in each connected subsequence (i.e., the segments which have odd index in the skew sequence) then the remainder of the segments inherit directions in the obvious way.

We now outline algorithms to construct the boundary curves $\{\gamma_K\}$ in a TV-scan triangular picture graph T* and to link these curves into the insidedness tree I(T*). A later section discusses some implementation details and suggests a hardware-software combination for converting the raw binary array representation of T* into a tree I(T*) of boundary curves.

Algorithm B

(1) Pass over the picture T* in TV-scan order generating a corridor sequence for each pair of adjacent horizontal rows and inserting end-of-corridor codes. Add an end-of-picture code as the final symbol. Call this the full sequence of contour segments.

(2) Process the full sequence from left to right parsing it into corridor sequences and further into connected subsequences by identifying consecutive (odd, even) pairs in each skew sequence. For each segment of the skew sequence determine its direction and assign it to the top or bottom sequence for the corridor. Segments assigned to the top sequence are linked to the appropriate segment in the bottom sequence constructed during the processing of the previous corridor. Segments assigned to the bottom sequence are placed in a list for use by the next corridor. The direction of the initial segment of each connected subsequence is propagated through the remainder of the subsequence and appropriate links are made between adjacent segments.

(3) Pass once through the full sequence looking for the next untraced contour segment. When such a segment s is encountered then trace through the linked sequence of segments (marking them as traced) starting at s and returning to s. Give this new curve a name and place the name in a list of curves with a reference to segment s as the top

of the curve. The initial segment s will always be of type / and if it is linked to the next segment (——) in the full sequence then the curve encloses a black region; otherwise, it encloses a white region. This information is recorded with the curve name and reference to s. Then the search for an untraced segment resumes directly after s in the full sequence. This is repeated until the end-of-picture code is encountered.

Algorithm B determines the geometry (except position) of each boundary curve γ_K in the picture T* and also indentifies the label of the immediately enclosed region ρ_K. The following algorithm constructs a tree which is almost identical to I(T*); the difference is that the tree computed has vertices corresponding to boundary curves γ_K and is rooted at a fictitious curve at ∞ called γ_∞. As a simple consequence of the one-to-one correspondence $\{\gamma_K\} \leftrightarrow \{\rho_K\}$ this difference is of no consequence. It does seem more natural to construct I(T*) this way since the $\{\gamma_K\}$ are the objects which have been computed by algorithm B.

Algorithm I

(1) For each curve on the list of boundary curves obtain its top segment s and then find the segment previous to s in the same corridor sequence. If s is the initial segment in a corridor sequence then link its curve $\gamma(s)$ to the fictitious curve at ∞ called γ_∞. If segment t precedes s in the same corridor sequence then link $\gamma(s)$ to the curve $\gamma(t)$ containing segment t. There are some interesting problems of what computation and data structure is most efficient for determining $\gamma(t)$ but we are content to post a simple warning sign a la Bourbaki. The linkage symbolizes the fact that $\gamma(s)$ and $\gamma(t)$ can be connected by a curve which does not intersect any other boundary curves. If $\gamma(s)$ and $\gamma(t)$ enclose

regions with identical labels then the link implies that $\gamma(s)$ and $\gamma(t)$ are siblings in the insidedness tree (i.e., $\gamma(s)$ and $\gamma(t)$ have a common immediately enclosing curve). Otherwise $\gamma(s)$ will be a child of $\gamma(t)$.

(2) When each curve has been given a link it remains only to transform sibling links into the appropriate child link. This can be done by tracing sequences of sibling links until a child link is found and then letting all the intermediate siblings inherit this child link to their common parent. Since all the links point backwards in the full (TV-scan) sequence there will always be an end to sibling links. The child links thus determined define the insidedness tree of boundary curves.

The regions ρ_K are implicit in I(T*) in the sense that each vertex with its children represent the outer and inner boundary curves for some region ρ_K. The degree of multiple connectivity of ρ_K is given by the number of such boundary curves.

If the position of regions is not important compared to their shape, size, orientation and degree of multiple connectivity the algorithm B suffices. However, if the position of curves is required the following modifications to algorithm B will do the job.

(1a) Proceed as in step 1 of algorithm B with some additional information encoded into the corridor sequences. Before the first corridor sequence we enter the index of that corridor (i.e., the first corridor containing a contrast face). Subsequently, a corridor index is inserted before a new corridor sequence if and only if the previous corridor sequence contained no bottom segments. Within each corridor every adjacent pair of the form (/ −) will have inserted after it the horizontal position within the corridor of the segment /.

(3a) Proceed as in step 3 of algorithm B except keep track of the corridor index by incrementing it for each corridor sequence which has a bottom sequence and resetting it from the encoded index otherwise. The top of each curve is a pair of the form (/ −) so the position can be determined from the current corridor index and the encoded horizontal position.

Some remarks about data-structures are appropriate at this point. Algorithm B constructs one-way linked circuits of segments (i.e., the directed circuits B_K of $B(T^*)$) so each segment must have a pointer to indicate its successor. The full sequence is probably best accommodated as a simple one-dimensional static array and since each segment entry requires only two bits (/, \ , —, and E where E means "not a segment, code follows") there can be some considerable data packing. The references to the previous bottom sequence and the additions to the new bottom sequence can be implemented as one continuous bottom queue whose maximum size is certainly bounded by the length of the rows r_i. The elements of the top sequence are used immediately so no further lists are required for step 2. A simple list of curve headers is needed for step 3.

Algorithm I indicates a requirement for a pointer in each curve header to record the links between curves but also requires some additional data structure to allow the determination of the curve header for $\gamma(t)$ when t is an arbitrary segment and $\gamma(t)$ is the curve containing t. One way is (at step B3) to set a tag bit in the segment-to-segment link which arrives at the top of a curve; then that link is set to point to the curve header rather than the top. The curve header for $\gamma(t)$ can be gotten by tracing the curve from t back to the header

for $\gamma(t)$ using the tag bit to recognize the curve header from a segment. No further data-structure is required for algorithm I.

A final note of some interest is that the end-of-corridor codes stipulated in algorithm B are unnecessary for the correct linking together of contour segments and even for creating the curve-to-curve links used to construct the insidedness tree. The reason for the first statement is implicit in our earlier discussion of the data structure for the top and bottom sequences. The validity of using a single queue for the super-sequence of successive bottom sequences rests on the fact that no matter where the end-of-corridor codes fall in the full sequence, the K^{th} top segment encountered matches the K^{th} bottom segment encountered. The parsing of corridor sequences into parity pairs can clearly be done without reference to the end-of-corridor information.

In creating curve-to-curve "next to" links we linked curve γ_s to γ_∞ if the top segment s was the first segment of a corridor sequence. In the absence of knowledge about where corridors begin we can still link γ_s to the curve $\gamma(t)$ which corresponds to the segment t previous to s in the full sequence because if s begins a corridor, then t ends the previous one so both γ_s and $\gamma(t)$ bound holes in ρ_∞ and should be siblings. The topmost curve is of course linked to γ_∞.

Compacting and Smoothing Boundary Curves

Each closed curve γ_K computed by algorithm B is a sequence of vectors of constant length in one of six possible directions and hence can be represented by a Freeman chain-encoding [13]. We shall assume that the horizontal direction to the right is labelled zero (0) and the subsequent labels 1, 2, 3, 4, 5 represent counter-clockwise rotations of 60^o. The outer curve in Fig. 5 is encoded as (000000055550110005433455555544554443332221111222333322345555444433322222111122221001 1). This can be condensed by recording the

initial direction and thereafter the length of each run and the difference (+ or -) at each bend. The above encoding becomes (0/7-4+1+2-3-1-1-2+1+6-2+2-3-3 -3-4+3+4-2+1+1+4-5-3-5-4+4-1-2+2-) in which minus (-) indicates the curve bends to the right or clockwise by 60^{o} and plus (+) indicates a bend to the left. A careful consideration of the length-bend encoding shows that each odd length is between identical bends and each even length is between unlike bends. This is essentially due to the fact that the triangular faces of the triangular grid can be partitioned into two sets F_1 and F_2 based on orientation and adjacent faces always belong to different sets. This means that the bends are redundant and needn't be explicitly encoded. If the lengths are in binary format then the bend can be determined by the low order bit of the length and the type of the previous bend. The encoding can thus be further condensed to (-0/7412311216223334342 11453544122). The convention we have adopted is that the curve starts at the first bend encountered in a TV-scan of the picture. With this convention the initial direction is redundant and the label of the immediately enclosed region can be deduced directly from the initial bend; (-) implies 'black' with initial direction 0 whereas (+) implies 'white' with initial direction 4.

When discrete grid systems are used to represent curves in the plane there occurs the distasteful phenomenon known as quantization error, the most undesirable effect of which is that perfectly straight lines are represented by zigzag polygons. In an attempt to undo this mischief several authors [15], [4], [12], [14] have proposed methods for smoothing digitized curves. In particular, Montanari [11] and Sklansky et al. [12] define a minimum perimeter polygon which has the same digitization as a given curve. In the terminology of Montanari [11] the triangular grid is a complete convex digitization scheme (CCDS) and the normal digitization of a boundary curve is the sequence of

triangular contrast faces corresponding to the segments of γ_K. The minimum perimeter polygon (MPP) is then the shortest polygon which lies entirely in the same faces and encounters them in the same cyclic order; intuitively visualize a rubber band woven through the contrast faces of γ_K. Figure 9 shows the outer boundary curve from Fig. 5 and its MPP while Fig. 10 is the MPP for the chromosome in Fig. 2. Although Montanari [4] gives a general method for computing the MPP of a digitization, it involves the solution of a nonlinear programming problem and so any shortcuts are worthy of attention. For the special case of the triangular grid we have obtained some simple rules which can be used to generate a good approximation to the MPP of a digitized boundary curve. The rules are stated in terms of the edge-length encoding of the boundary and are derived from the following lemmas. Table 2 gives most of the rules we have derived.

Let us call the sequence of contrast faces for a curve γ_K a road and let α and β denote the outer and inner boundary curves for the annular region defined by the road. The boundary curves α and β are formed by the level edges of faces in the road, one curve being edges labelled black, the other white. Any curve in the road joining points on α and β we call a transversal and so contrast edges are transversals; it is clear that any curve which follows the annular road cuts every transversal.

Lemma 1

For every circular neighborhood of an MPP vertex the portion of the neighborhood on the concave side of the vertex contains a point not in the road.

Proof

Given a neighborhood of an MPP vertex whose concave sector is entirely in the road a line across this sector can be drawn to shorten the MPP.

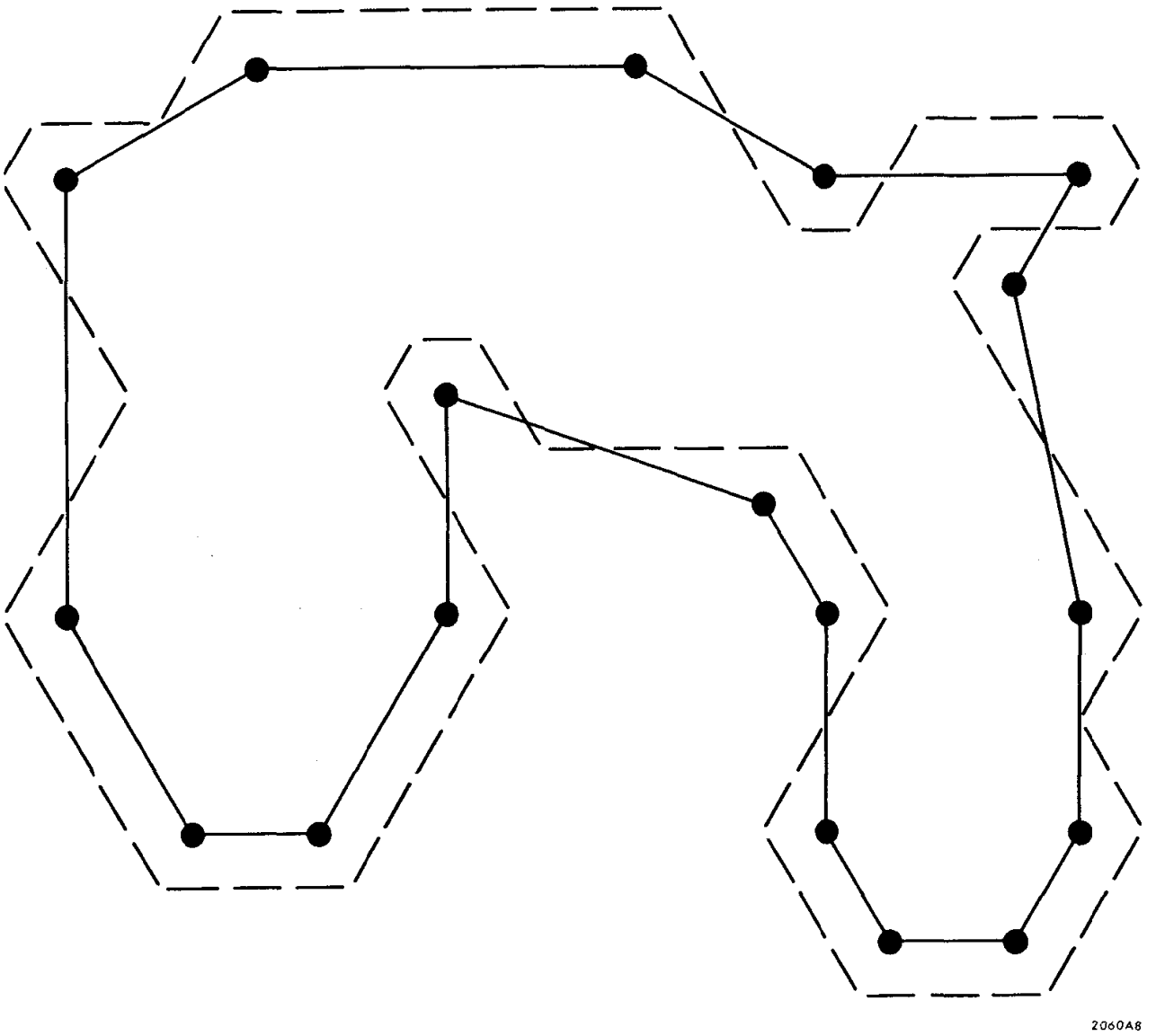

FIG. 9--Minimum perimeter polygon smoothing of longest boundary curve of Fig. 5.

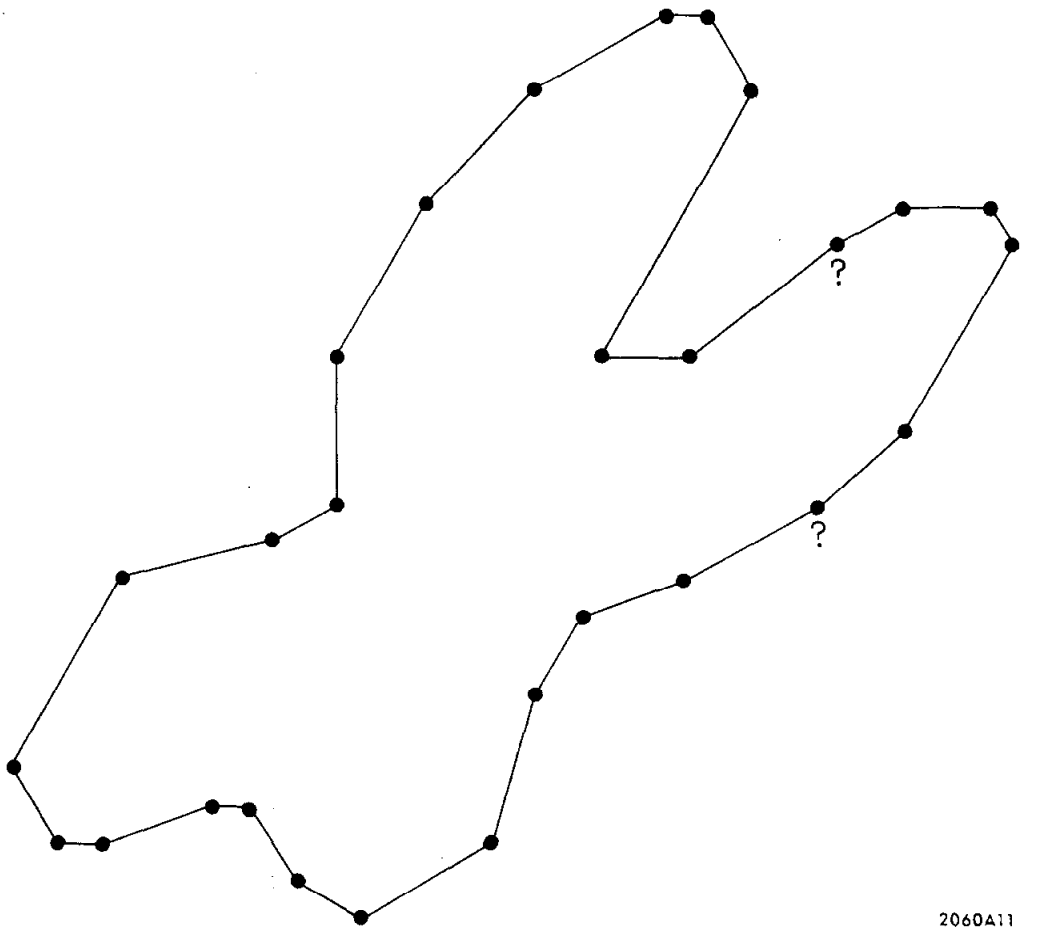

FIG. 10--Smoothed version of boundary of chromosome depicted in Fig. 2.

Table 2

	Condition	MPP	Vertex
1.	odd$\uparrow$	yes	
1.1.	odd$\uparrow(\neq 2)$	yes	yes
1.2.	odd$\uparrow$2 even	yes	yes
1.3.	odd$\uparrow$2 odd	yes	no
2.	2$\uparrow$2		no
3.	$3^+\uparrow 3^+$	yes	
4.	$3^+\uparrow 5^+$	yes	yes
5.	.4$\uparrow$(2 or 4)		no
6.	.2$\uparrow$		no
6.1.	.2$\uparrow$even		no
6.2.	.2$\uparrow$odd	yes	no
7.1.	$4^+\uparrow(22)^*4^+$	yes	
7.2.	$4^+\uparrow(22)^*$.	yes	
8.	$6^+\uparrow$22	yes	
8.1.	$6^+\uparrow(22)^*$.	yes	yes
9.	4$\uparrow$4.		no
10.	$8^+\uparrow$22	yes	yes
11.	.(2n)$\uparrow$2 (2m). $\lvert n-m\rvert\leq 1$		no
12.	.((2n$\uparrow$2$\uparrow$)* (2n) 2		no

$\uparrow$ bends to which conclusion applies.

n^+ any length $\geq n$.

. bend known to be on MPP.

α^* repeat α 1 or more total times.

(2n) refers to a single edge of length 2n in rules 11 and 12.

Corollary 1.1

No MPP vertex is interior to the road.

Corollary 1.2

No MPP vertex lies on a straight line segment of one of the boundary polygons α and β.

We shall call a vertex on α or β a concave bend or convex bend according as the concave or convex sector of the vertex is outside the road. This definition corresponds to local concavity or convexity of the road at the particular boundary point.

Corollary 1.3

No MPP vertex is at a convex bend of α or β.

If a bend transversal is a contrast edge across the road corresponding to a bend in the contour curve γ_K defining the road and if a bend vertex is the end-vertex of a bend transversal lying on the concave side of the contour bend then we have the following result.

Lemma 2

Each MPP vertex is a bend vertex (i.e., concave bend in road).

This lemma allows us to restrict our attention to bend vertices which correspond to contour bends. To discover which bend vertices are really MPP vertices stronger medicine is required--supplied in the following lemmas. Let a chord $\overline{PQ}$ be a line segment in the road with both P and Q on the same road boundary curve (α or β) and let the sector generated by chord $\overline{PQ}$ be the simply-connected subregion of the road formed by the chord.

Lemma 3

If S is a road sector generated by chord $\overline{PQ}$ then MPP $\cap$ S $\subseteq \overline{PQ}$.

Proof

If the MPP intersects the sector S then it crosses into and out of S via the chord $\overline{PQ}$. It cannot intersect the interior of S because that would imply an unnecessary MPP vertex.

Lemma 4

If P and Q are points on the MPP and $\overline{PQ}$ lies in the road then $\overline{PQ} \subseteq$ MPP.

As an example of the use of these lemmas we prove part of condition 1.1 in table 2. We demonstrate that a contour bend between an odd edge and an edge of length 1 generates a bend vertex of the MPP. Referring to Fig. 11 we apply lemmas 3 and 4 to sector (BFEC) which shows that $\overline{PQ} \subseteq$ MPP because MPP must cross transversals $\overline{EQ}$ and $\overline{PC}$. Since the MPP doesn't intersect the interior of sector (ADCB) it follows that P is an MPP vertex. Other conditions are proved in a similar fashion.

The MPP in Fig. 10 contains 28 vertices out of 58 contour bends in Fig. 2. All but two of these vertices were clearly established as MPP vertices or not; the two which were found to be on the MPP but not known to be vertices are marked with a '?' in Fig. 10. The first five conditions in table 1 appear to be the most useful. For example, the 58 contour bends of the boundary curve in Fig. 2 represent only 53 bend vertices because there are five 1s in the edge length sequence; Of these 53, 23 are established as MPP vertices and 18 as not MPP vertices using just conditions 1, 1.1, 1.2, 1.3, 2. That leaves only 12 vertices to be tested by other means.

Implementation Considerations

First we must consider how to generate a TV-scan triangular picture graph T*. Most image scanners that are currently in use transform a

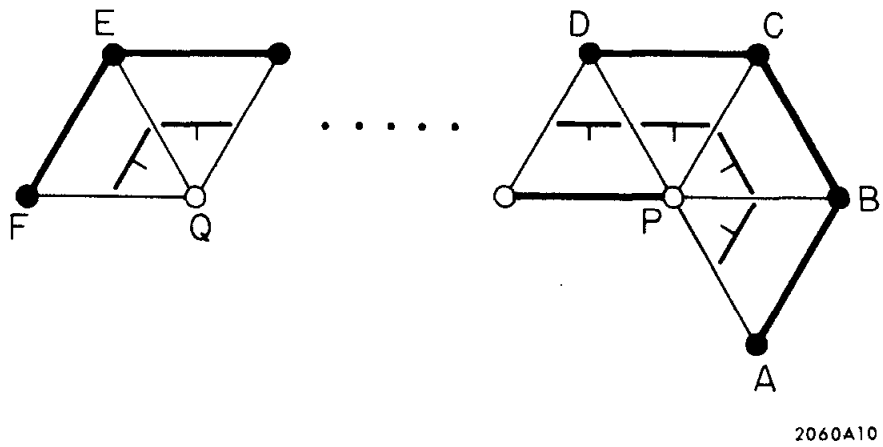

FIG. 11--Section of annular road straddling an odd length segment followed by a unit length segment.

continuous two-dimensional image intensity distribution into a set of quantized sample intensity values at the vertices of a square grid as depicted in Fig. 1. If image intensity is quantized into two values (0,1) then we have binary digital pictures as discussed by Rosenfeld [1],[2]. These square grid digital pictures are generated top-to-bottom and left-to-right in the familiar TV-scan or raster scan. It is natural to think of a picture as consisting of rows of sample values spaced Δj units apart along each scanned line where the distance between successive lines is Δi. For the square grid we have $\Delta i = \Delta j$ and the sample points are of the form $(i\Delta i, j\Delta j)$ if the northwest corner of the picture is the origin. It is not hard to see that a TV-scan triangular picture graph T* also consists of equally spaced (Δi) rows of equally spaced (Δj) sample points with two differences. First, the ratio $(\Delta i/\Delta j) = \sqrt{3}/2 \doteq .866$ rather than 1 and second, the rows are staggered so that the sample points are of the form

$$(i\Delta i,\ j\Delta j + \{\ \underline{\text{if}}\ i \equiv 0\ \underline{\text{mod}}\ 2\ \underline{\text{then}}\ \Delta j/2\ \underline{\text{else}}\ 0\ \}\ ).$$

Neither of these differences represent radical departure from the overall design of image scanners.

There is also the possibility of simulating a triangular grid scanner when the actual hardware produces square-grid pictures. The technique will be approximate in the sense that triangular faces will have a height-to-width ratio of $(\Delta i/\Delta j) = 1$ rather than the equilateral ratio $\sqrt{3}/2 \doteq .866$. The transformation from square-grid to triangular-grid is suggested by the staggered row phenomenon. Each row r_i with even index $i = 2K$ remains as is while rows with odd index $i = 2K + 1$ are replaced by rows containing sample values at $j\Delta j + \Delta j/2$, the new values being averages between adjacent values in the original row. The interpolation is clearly more meaningful if the original square-grid picture

is quantized at more than 2 levels and the reduction to binary (via a threshold) occurs after the required two-point interpolations. As far as the triangular face distortion is concerned it is an entirely systematic one which can probably be adjusted for; although this is a real nuissance, we shall see in the next section that the theory of connectivity and boundaries in triangular picture graphs does not depend on the equilateral property of triangles in the usual triangular grid. Hence, the material of previous sections is still applicable with minor adjustments whenever actual lengths and distances are involved. Notice, for example, that there are still two varieties of triangular face based on orientation!

The second problem is that of generating corridor sequences from pairs of successive rows of the triangular picture graph. To illustrate we will generate the corridor sequence σ_6 between rows r_5 and r_6 in Fig. 5. The derivation is depicted in Fig. 12. It is clear from theorem 2 that the contrast edges of r_5 and r_6 play a central role in the structure of corridor sequence σ_6. We therefore immediately define a binary difference vector Δr_K for each row r_K as

$$\Delta r_K = r_K \oplus SR1(r_K),$$

where SR1 means shift right one place and $\oplus$ is exclusive or bit-by-bit. The difference vector Δr_K is a selector for the contrast edges in row r_K. Each 1 in Δr_K represents the midpoint of a contrast edge in row r_K and there is a natural staggering effect from row to row as shown in Fig. 12. Next, the difference vectors of two adjacent rows are interleaved in a perfect shuffle and this shuffled vector is replaced by an equivalent index sequence of its 1s. Because of the perfect shuffle, the parity of each index identifies which row the contrast edge lies on. The index sequence can be partitioned into successive

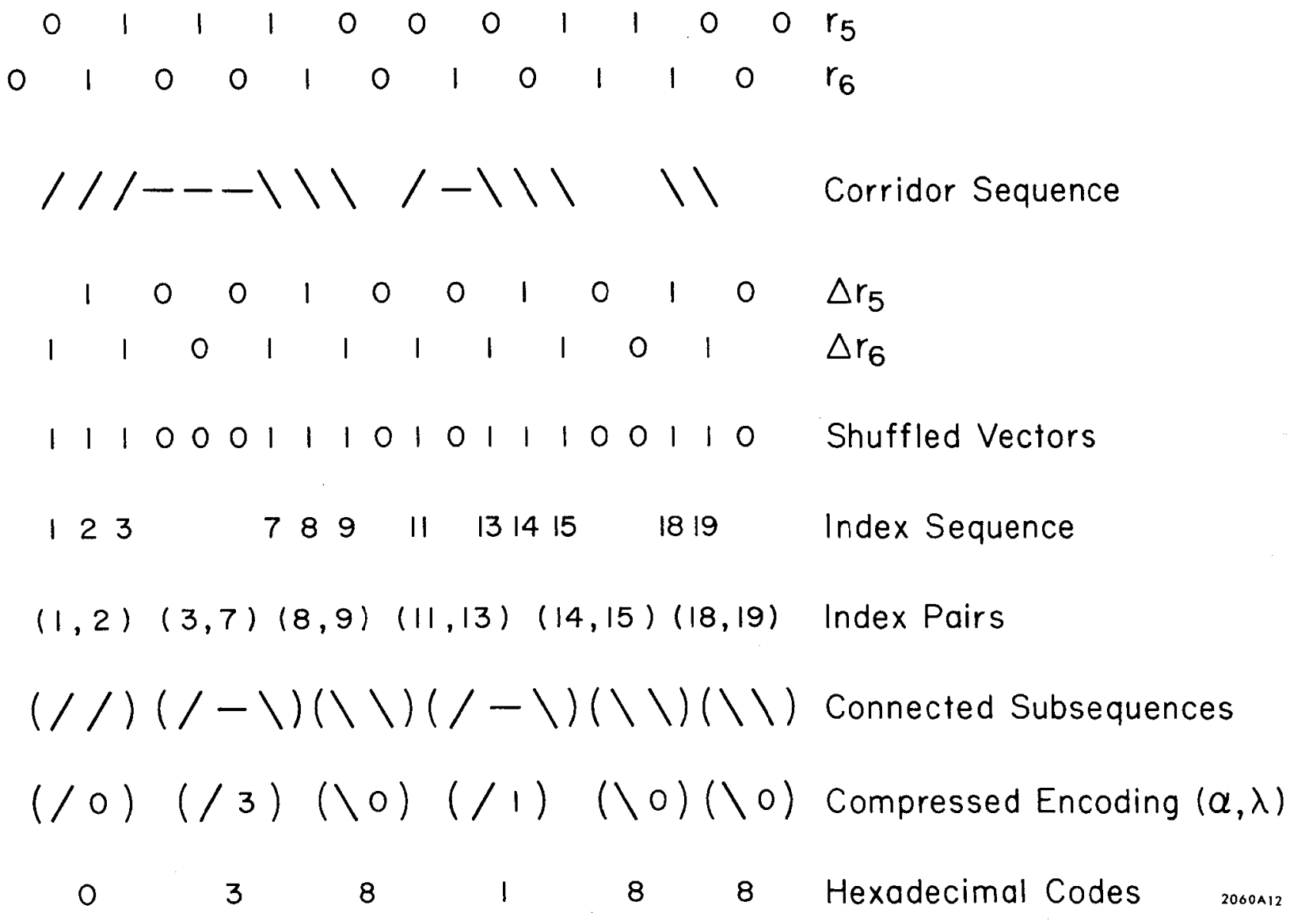

FIG. 12--Computing and encoding the corridor sequence σ_6 from adjacent binary rows (r_5, r_6) of the triangular picture graph in Fig. 5.

pairs as shown in Fig. 12. Each index pair (x,y) corresponds to a unique connected subsequence of the corridor sequence σ_6. If we use the notation (α, λ, β) to denote connected subsequences where $\alpha \in (/, \backslash)$, $0 \leq \lambda$ is an integer denoting the number of flat (—) contour segments, and $\beta \in (/, \backslash)$ then α, β, λ can be determined from x and y by the following formulas:

$$\alpha \leftarrow \underline{\text{if odd}}\ (x)\ \underline{\text{then}}\ /\ \underline{\text{else}}\ \backslash\ ;$$

$$\beta \leftarrow \underline{\text{if odd}}\ (y)\ \underline{\text{then}}\ /\ \underline{\text{else}}\ \backslash\ ;$$

$$\lambda \leftarrow (y - x - 1);$$

We know from previous considerations that the value of β is redundant, being a function of α and λ given by

$$\beta \leftarrow \underline{\text{if odd}}\ (\lambda)\ \underline{\text{then opposite}}\ (\alpha)\ \underline{\text{else}}\ \alpha;$$

where opposite interchanges / and \ . This formula shows that the low order bit δ of the binary representation for λ is equivalent to the boolean expression $(\alpha \neq \beta)$. Because of this redundancy we encode each connected subsequence by a pair (α, λ) as shown in Fig. 12. It is possible to encode all subsequences (α, λ) for $0 \leq \lambda \leq 6$ by a 4-bit hexadecimal code, leaving one code 7 for end-of-corridor and another F for an escape code. After an escape code an 8-bit code is used for an (α, λ) with $\lambda > 6$. Each end-of-corridor (EOC) after a corridor sequence with no bottom segments is followed by an 8-bit row-index increment. An initial EOC is followed by the index of the initial non-empty corridor sequence and code FFF indicates end-of-picture. The entire Fig. 5 can thus be encoded by the hexadecimal sequence (701F0772837892781870 8C87 038188788000000789090187B8907BFFF). The high order bit of each code (except 7 and F) represents α with $0 \equiv /$ and $1 \equiv \backslash$. This encoding into connected

sequences of corridor sequences involves a partial completion of step 2 in algorithm B and leads to an efficient table lookup based on the 16 hexadecimal codes.

The transformation from two binary rows to the hexadecimal digit sequence can be implemented in hardward employing a shift register to store one row or in software using shifts, exclusive or and a left justify instruction (if available).

Generalizations

The theory developed for triangular picture graphs extends in some ways to picture graphs defined on grid systems which are less regular or whose faces are not restricted to triangles. The extent to which each of these systems fails to generalize the previous theory reveals the essential reasons why the triangular grid is emphatically the best grid. We begin by defining a planar picture graph to be any binary vertex-labelled planar graph with the set of black vertices bounded. Figure 13 depicts such a picture graph; it is irregular and has triangular, quadrilateral and pentagonal faces. A connectivity graph can be defined exactly as before using the ideas of level and contrast edge. The level edges of the connectivity graph are heavy in Fig. 13. Let a planar picture graph be called convex if each face is convex. Now every contrast face of a convex planar picture graph has an even number of contrast edges but that number can be larger than 2 if the face is not triangular. This suggests an advantage for triangular-faced picture graphs but we can still try to define boundary curves as segments across contrast faces joining midpoints of contrast edges. There are several ways to do this pairing when a face has more than 2 contrast edges. One way is to join the two contrast edges bounding a sequence of level black edges around the face. These coutour segments are

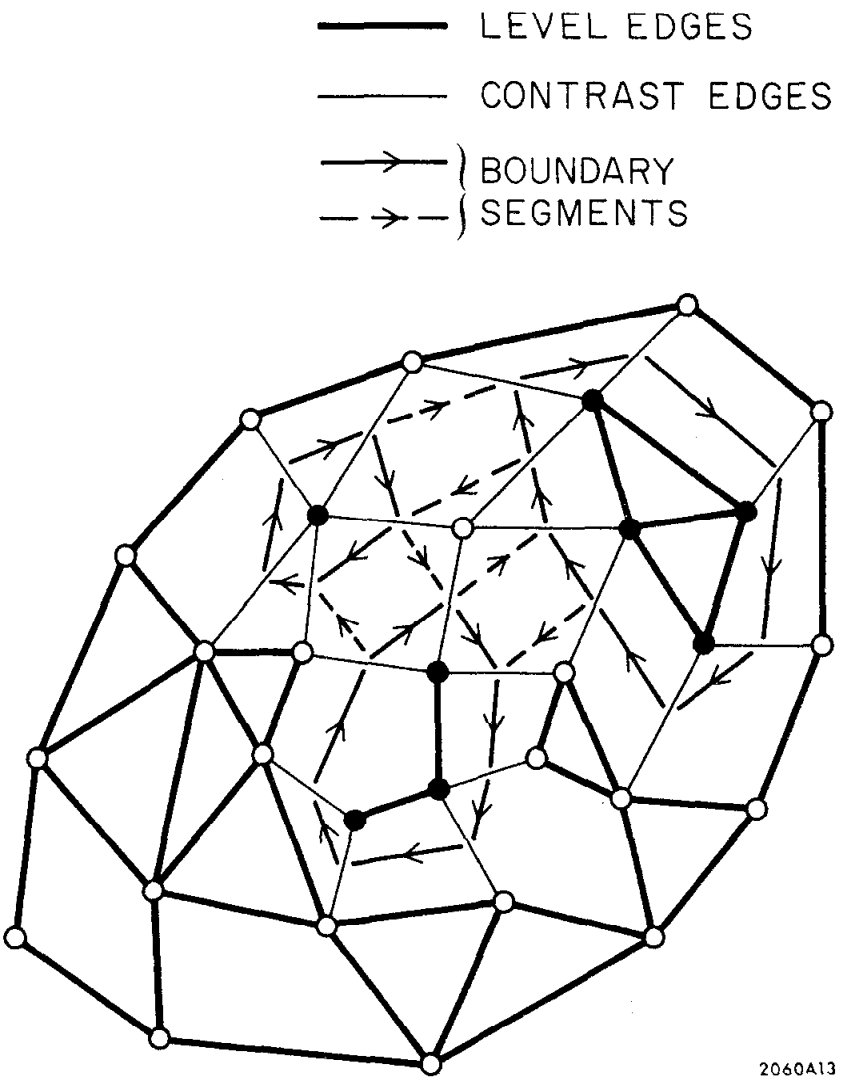

FIG. 13--A convex planar picture graph.

shown as solid arrows in Fig. 13. Another is similar but the segments span white sequences. These segments (when different) are depicted as dashed. Either choice allows a consistent definition of boundary curves which form a disjoint family of simple closed curves. The problem is that neither definition is compatible with the connectivity graph in the sense of an earlier section. Notice that Fig. 13 contains a white vertex which is isolated in the connectivity subgraph but is not separated from the rest of the white subgraph by any boundary segments using the first boundary choice (solid). There is an isolated black vertex to play a similar role for the second boundary choice (dashed if two). The most we can say is that the solid boundary is compatible with an asymmetric definition of connectivity in which black vertices must share an edge but white ones need only share a face. The other boundary choice corresponds to the reverse situation. The well-known problem of connectivity on a square grid [3] is a special case of this phenomenon. It is known that 4-connectedness for black and 8-connectedness for white (or vice versa) must be used on the square grid to obtain reasonable notions of boundary. But 8-connectedness is easily seen to be equivalent to face-connectedness!

We casually used the term contour segments for this more general case but that part of the theory does not generalize to non-triangular faces except by rules for triangulating these faces depending on the vertex-labels of the face. On the other hand, any planar picture graph composed of triangular faces admits compatible definitions of connectivity and boundaries formed by contour segments. The triangular grid happens to be the only regular planar tesselation by triangles and also is relatively isotropic.

References

1. A. Rosenfeld, Picture Processing by Computer (Academic Press, New York, 1969).
2. A. Rosenfeld, "Picture Processing by Computer", Computing Surveys Vol.1, No. 3, September 1969.
3. A. Rosenfeld, "Connectivity in Digital Pictures", Journal of the Association for Computing Machinery, Vol. 17, No. 1, January 1970, pp. 146-160.
4. U. Montanari, "A note on Minimal Length Polygonal Approximation to a Digitized Contour", Communications of the ACM, Vol. 13, No. 1, January 1970, pp. 41-46.
5. O. Buneman, "A Grammar for the Topological Analysis of Plane Figures", Machine Intelligence 5 (edited by B. Metzer and D. Michie), American Elsevier, New York, 1970, pp. 383-393.
6. M. Golay, "Hexagonal Parallel Pattern Transformations", IEEE Trans. on Computers, Vol. C-18, No. 8, August 1969, pp. 733-740.
7. B. McCormick, "The Illinois Pattern Recognition Computer--ILLIAC III", IEEE Trans. on Computers, Vol. EC-12, December 1963, pp. 791-813.
8. S. Gray, "Local Properties of Binary Images in Two Dimensions", IEEE Trans. on Computers, Vol. C-20, No. 5, May 1971, pp. 551-561.
9. C. Zahn, "Two-Dimensional Pattern Description and Recognition via Curvaturepoints", Report No. SLAC-70, Stanford University, December 1966.
10. C. Zahn, "A Formal Description for Two-Dimensional Patterns", Proc. International Joint Conference on Artificial Intelligence, Washington, D. C., May 1969 (also available as SLAC-PUB-538).

11. U. Montanari, "On Limit Properties in Digitization Schemes", Journal of ACM, Vol. 17, No. 2, April 1970, pp. 348-360.

12. J. Sklansky, R. Chazin, and B. Hansen, "Minimum-Perimeter Polygons of Digitized Silhouettes", IEEE Trans. on Computers, Vol. C-21, No. 3, March 1972, pp. 260-268.

13. H. Freeman, "On the Digital Computer Classification of Geometrical Line Patterns", Proc. Nat. Electron Conf., October 1962, pp. 312-324.

14. C. Zahn and R. Roskies, "Fourier Descriptors for Plane Closed Curves", IEEE Trans. on Computers, Vol. C-21, No. 3, March 1972, pp. 269-281.

15. H. Freeman and M. Glass, "On the Quantization of Line Drawing Data," IEEE Trans. on Systems Science and Cybernetics, Vol. SSC-5, No. 1, 1969, pp. 70-79.

16. U. Montanari, "Continuous Skeletons from Digitized Images", Journal of ACM, Vol. 16, No. 4, October 1969, pp. 534-549.