A Method for Obtaining Skeletons Using a Quasi-Euclidean Distance

U. MONTANARI*

Politecnico di Milano,† Italy

ABSTRACT. The problem of obtaining the skeleton of a digitized figure is reduced to an optim policy problem. A hierarchy of methods of defining the skeleton is proposed; in the more con plicated ones, the skeleton is relatively invariant under rotation. Two algorithms for compu ing the skeleton are defined, and the corresponding computer programs are compared. A cı terion is proposed for determining the most significant skeleton points.

KEY WORDS AND PHRASES: feature recognition, pattern recognition, skeleton, digitized figur optimal policy, optimal path, reticular networks, discrete wave propagation

R CATEGORIES: 3.63

1. *Introduction*

In pattern recognition, methods for extracting the main features of a picture ; needed. When all the information is contained in the contour of a figure, a "skeleto method may be useful. In the skeleton method, the contour is replaced by a ge eralized axis of symmetry, the "skeleton" (Blum [1]: medial axis), together wi associated values of a parameter. In order to obtain the skeleton it is necessary compute the distance of every point of the plane from the set of points of the figu From the skeleton, the contour of the figure can easily be regenerated, so the amoı of information involved is the same. However, the skeleton seems to emphas some properties of the picture; for instance, curvature properties of the cont correspond to topological properties of the skeleton [1]. The concept of a skele defined in the real plane was first proposed by Blum [1]. A mathematical theory then developed by Kotelly [2] and Calabi [3, 4].

In applications, it is often necessary to process pictures given in digitized fo It may therefore be convenient to define a skeleton transformation directly in discrete case. This discrete skeleton should, if possible, have properties very sim to those of the continuous one. Such a transformation was defined by Rosenfeld Pfaltz [5, 6], who also found a very simple algorithm for obtaining it. However, distance they considered is substantially not Euclidean, so, for instance, this skel is not invariant under rotation. Moreover, no improvement is obtained even if number of points is increased.

In this paper it is suggested that the problem of obtaining a discrete skeleton be reduced to the determination of optimal paths through a reticular network. minimal path length between two vertices in the network provides an approxima

* Present address: Istituto per l'Elaborazione dell' Informazione, Consiglio Nazionale Ricerche, Pisa, Italy.
† Istituto di Elettrotecnica ed Elettronica. This research was supported in part by Com di Ingegneria, Consiglio Nazionale delle Richerche.

of the distance between the corresponding points in the real plane. By increasing the complexity of the network, we can obtain a "distance" that approximates Euclidean distance as closely as we want. In this sense we speak of "quasi-Euclidan" distances. The skeleton obtained using the simplest network coincides with Rosenfeld's skeleton.

In Section 2 several definitions of the skeleton in the real plane are given. One of those is emphasized, because it can be easily extended to the discrete case. In Section 3, the various reticular networks are specified in terms of Farey sequences, and in Section 4 the minimal path problem which corresponds to skeleton determination is described. The solution of this problem for an infinite reticulum allows us to derive, in Section 5, analytical expressions for the minimal path length between two points of the network for the various methods. In Sections 6 and 7, two algorithms are described that take advantage of the regularity of the network to compute simultaneously the lengths of the minimal paths from all the points of the network to the figure. In order to make it easier to apply these results for pattern recognition purposes, in Sections 9 and 10 a method of eliminating the less significant skeleton points is described. In Section 11, FORTRAN IV programs which implement the two algorithms are briefly described, and their results are compared.

2. *The Skeleton in the Real Plane*

There are many equivalent definitions of the skeleton in the real plane. We give four of them, proposed in [1] and [3], in order to introduce some concepts that arise in the discrete case as well.

(a) Let us interpret the contour as an initial wavefront and then let it propagate with the constraint that no point can be excited more than once. In this situation, there will exist points where the wavefront "intersects itself" and then is extinguished. This locus, together with the corresponding set of times, is the skeleton [1] (see Figure 1(a)). One could ask whether the contour propagates "inside" or "outside" the figure. In what follows, we assume it propagates outside, so only the external skeleton is considered. To obtain the internal one, it suffices to complement the figure.

(b) We define a function at every point P of the plane whose value is the distance of P from the contour:

$$d(P) = \min d(P, Q), \qquad Q \in \mathfrak{F}, \tag{1}$$

where $\mathfrak{F}$ is the set of points of the figure, and $d(P, Q)$ is the usual Euclidean distance between two points. Let us visualize this function as a surface; the "ridges" of the surface, namely the points where we cannot define a tangent plane, together with the corresponding distances, constitute the skeleton [1].

(c) Let us define the set

$$\mathfrak{I}(P) = \{R \mid d(P, R) = d(P), \qquad R \in \mathfrak{F}\}.$$

If $\mathfrak{I}(P)$ contains more than one point, P is a skeleton point. In Figure 1(b), $\mathfrak{I}(P_1) = \{R_1\}$, so P_1 is not a skeleton point, but $\mathfrak{I}(P_2) = \{R_2', R_2''\}$, so P_2 is a skeleton point [3].

(d) Given a point P, a minimal path from P to the contour is a straight-line segment PR where $R \in \mathfrak{I}(P)$. Evidently, P is a skeleton point if it does not belong

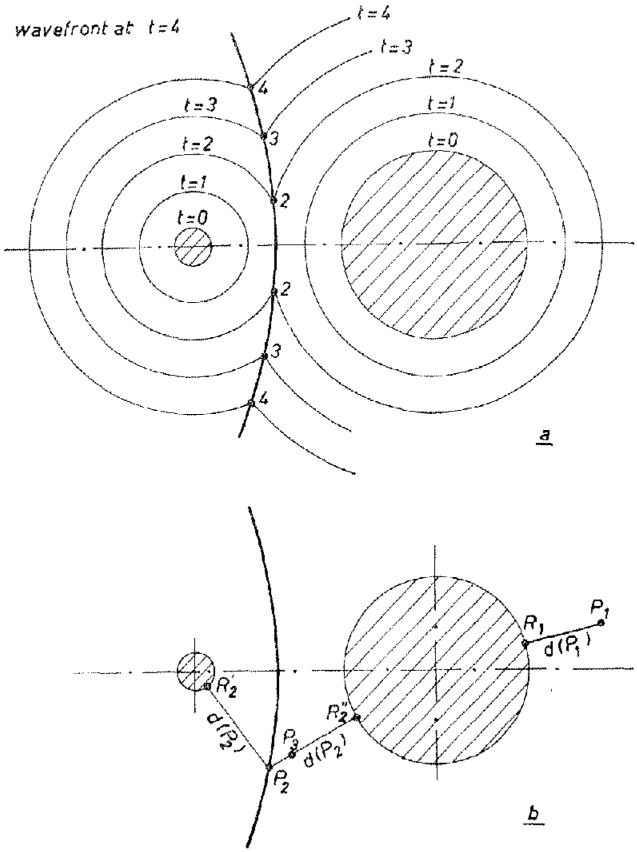


FIG. 1. Examples of application of different definitions of skeletons

to the minimal path of any other point. In Figure 1(b), P_3 belongs to the minimal path P_2R_2'', so P_3 is not a skeleton point; but P_2 does not belong to any minimal path, so P_2 is a skeleton point.

This last definition was introduced in a slightly different manner by Calabi [3]. It can be easily extended to the discrete case, since the concept of minimal path is well defined in the discrete case too.

3. *Reticular Networks as Approximations of the Real Plane*

In this section we define the networks with which we approximate the real plane. Usually, a digitized picture is given in the form of a rectangular array of elements a_{ij} $(i = 1, \cdots, r;\ j = 1, \cdots, s)$. We can interpret (i, j) as the coordinates (x_p, y_p) of a point P in the real plane. We connect some pairs of these points with straight-line segments. In Figures 2(a)–(c) we show all the connections for the methods, which will be numbered 0, 1, 2, respectively. In Figures 3(a)–(c) an "elementary cell" (that is, all the points directly connected with a given point), and the corresponding segments with their lengths, are shown for methods 0, 1, 2. Note that in method 0, every point is directly connected only to the four nearest points, while in method n $(n \neq 0)$, the connections are the simplest ones such

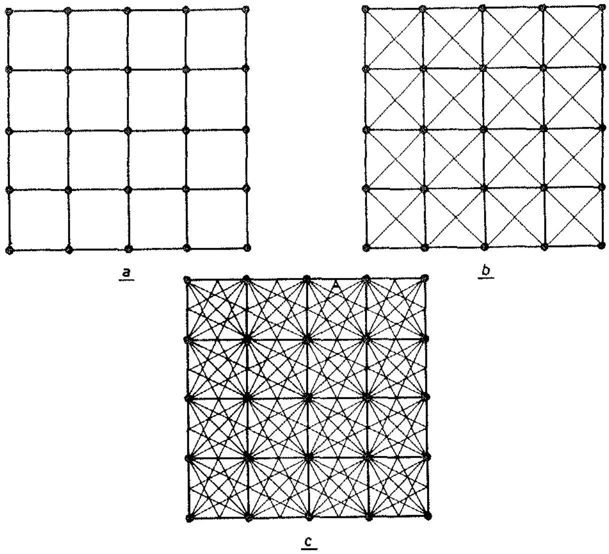


FIG. 2. Reticular networks for methods 0, 1, 2

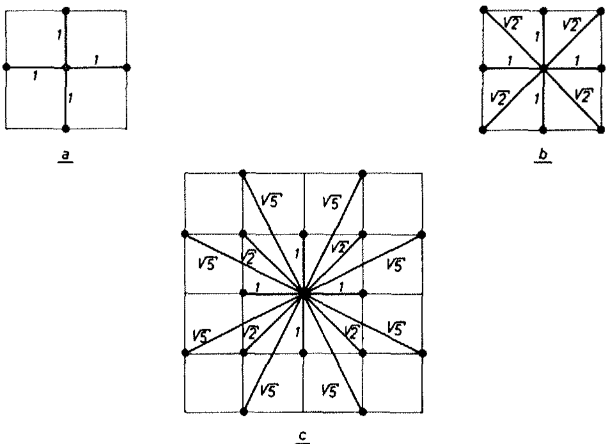


FIG. 3. Elementary cells for methods 0, 1, 2

that every point P_i is connected (directly or not), by a straight line, with every point P_j belonging to the square centered at P_i and with side length $(2n + 1)$.

We can easily see that the slope of every segment, relative to the axes, is represented by a rational number. For symmetry, we can represent the elementary cell by giving only the segments with slopes t such that $0 \leq t \leq 1$. See Figures 4(a)–(d) for methods 0, 1, 2, 3, respectively.

In method n ($n \neq 0$), these slopes constitute the Farey sequence of order n (Table I), that is, the ordered sequence of all rational numbers between 0 and 1 with denominators less than or equal to n (see, for example, [7]). The directions having those slopes can be thought of as allowed directions for the wavefront.

Let us now define a reticular network whose vertices and arcs are in one-to-one correspondence with the picture points and connections, respectively, so that to any path through the network there corresponds a polygonal arc on the real plane.

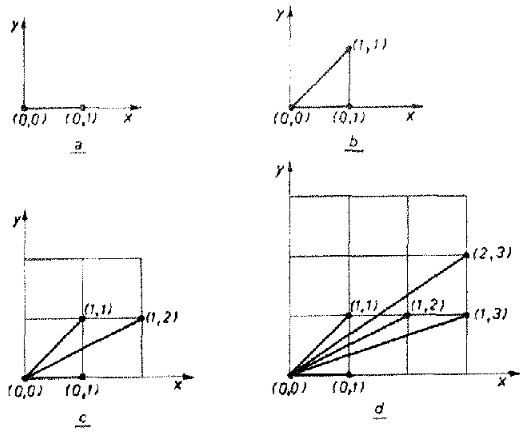


FIG. 4. Connections in the elementary cells correspond to terms of Farey sequences. Methods 0, 1, 2, 3

TABLE I

n	*Farey sequence of order n*
1	0, 1
2	0, $\frac{1}{2}$, 1
3	0, $\frac{1}{3}$, $\frac{1}{2}$, $\frac{2}{3}$, 1
4	0, $\frac{1}{4}$, $\frac{1}{3}$, $\frac{1}{2}$, $\frac{2}{3}$, $\frac{3}{4}$, 1
5	0, $\frac{1}{5}$, $\frac{1}{4}$, $\frac{1}{3}$, $\frac{2}{5}$, $\frac{1}{2}$, $\frac{3}{5}$, $\frac{2}{3}$, $\frac{3}{4}$, $\frac{4}{5}$, 1
6	0, $\frac{1}{6}$, $\frac{1}{5}$, $\frac{1}{4}$, $\frac{1}{3}$, $\frac{2}{5}$, $\frac{1}{2}$, $\frac{3}{5}$, $\frac{2}{3}$, $\frac{3}{4}$, $\frac{4}{5}$, $\frac{5}{6}$, 1

To every such arc, a length is assigned equal to the sum of the lengths of the associated connections. Given two points, P_i and P_j, and the two corresponding vertices P_i' and P_j', let us define two functions: $d = d(P_i, P_j)$ is the Euclidean distance between the points P_i and P_j, and $T = T(P_i', P_j')$ is the length of the minimal path through the network between the vertices P_i' and P_j'. Evidently, we have

$$T(P_i', P_j') \geq d(P_i, P_j). \tag{2}$$

4. *A Minimal Path Problem*

We now prove the equivalence of the skeleton problem with an optimal policy problem. This problem is a well-known one, and its solution is equivalent to the solution of a system of functional equations whose unknowns are minimal path lengths.

Let us define the following sets:

$U = \{P' \mid 1 \leq x_p \leq r; 1 \leq y_p \leq s; x_p, y_p$ integers$\}$, the set of all the vertices.

$I = \{P' \mid a(P) =$ .TRUE.$\}$, the set of vertices corresponding to the points of the figure.

$E = \{P' \mid a(P) =$ .FALSE.$\} = U - I$, the set of vertices corresponding to the points of the picture which are external to the figure.

Let us take an extra vertex P_N' and connect it with all the vertices $P' \in I$ using arcs whose length is assumed to be zero. For every vertex P_i', we can find a minimal path to P_N'.

We define an external point P_i to be a skeleton point if and only if $P_i' \in E$ and

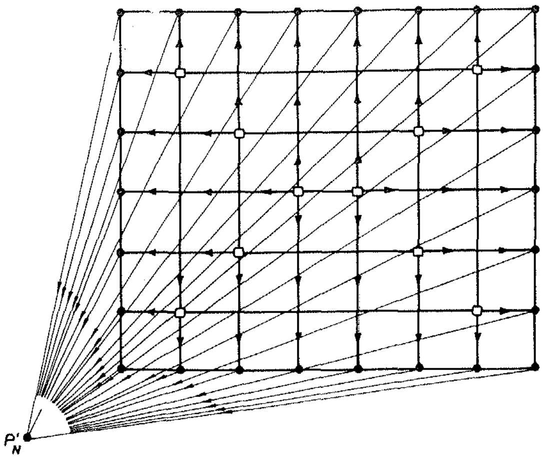


FIG. 5. An application of the definition of discrete skeleton. The points marked with the dot are internal points; the points marked with the square are skeleton points. The length of every arc is 1.

it does not belong to a minimal path from any other vertex to P_N'. To construct the skeleton, we determine for every vertex P_i' the "optimal policy" (or policies), namely the arc (or arcs) connected to P_i' belonging to a minimal path from P_i' to P_N', and countersign it as going out from P_i'. P_j is a skeleton point if and only if $P_j' \in E$ and there is no arc connected to P_j' and countersigned as going in. (See, for example, Figure 5.)

The determination of the optimal policy is a well-known problem. Bellman [8] has shown the solution of this problem to be equivalent to the solution of the following system of functional equations:

$$\begin{aligned} T_i &= \min (t_{ij} + T_j), \qquad i = 1, 2, \cdots, N-1, \qquad j = 1, \cdots, N, \qquad j \neq i, \\ T_N &= 0, \end{aligned} \tag{3}$$

where, if the arc connecting the vertices P_i' and P_j' exists, t_{ij} is equal to the length of this arc; otherwise, $t_{ij} = \infty$.

T_i in (3) is the length of the minimal path to the figure, that is, $T_i = T(P_i, P_N)$. For every vertex P_i', the "optimal policy" is given by the arc (or arcs) t_{ik} for which

$$T_i = t_{ik} + T_k. \tag{4}$$

Then P_k is not a skeleton point if and only if we have $T_k = T_i - t_{ik}$ for some P_i'. To construct the skeleton, we must thus:

(a) determine for every node P_i' the distance T_i to the figure;

(b) construct the set

$$S = \{P_k \mid P_k' \in E; \quad \text{for every} \quad P_i', \quad T_k \neq T_i - t_{ik}\};$$

(c) associate with every point $P_k \in S$ the parameter T_k.

5. *A Quasi-Euclidean Distance*

In this section it is proved that we can always find a minimal path between every pair of vertices of our reticulum (assumed infinite) such that the corresponding

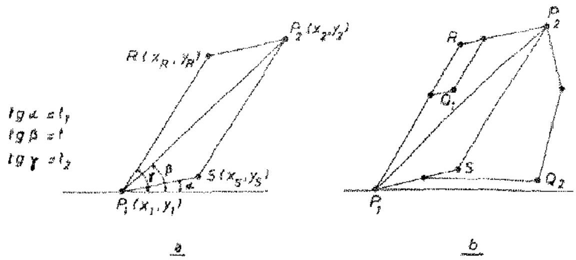


FIG. 6. Minimal paths from P_2 to P_1 in an infinite reticulum

polygonal arc consists of just two straight-line segments. In consequence, an analytical expression for the minimal path length between any pair of vertices is obtained, and the errors in the various methods as compared with Euclidean distance are computed.

To compute the length of the minimal path between the pair of vertices (P_1', P_2'), let t be the slope of the segment $\overline{P_1P_2}$. By symmetry, it suffices to assume that $0 \le t \le 1$. Let

$$P_1 = (x_1, y_1), \qquad \Delta x = x_2 - x_1, \qquad t = \frac{\Delta y}{\Delta x};$$

$$P_2 = (x_2, y_2), \qquad \Delta y = y_2 - y_1, \qquad 0 \le t \le 1.$$

THEOREM 1. *Suppose given an infinite reticular network of type n.*

(a) If the rational number t belongs to the Farey sequence of order n, then only one minimal path between P_1' and P_2' exists. The corresponding polygonal arc connecting P_1 and P_2 is a single straight-line segment. The length of the minimal path is thus

$$T(P_1', P_2') = \Delta x(1 + t^2)^{\frac{1}{2}}. \tag{5}$$

(b) Suppose that the rational number t does not belong to the Farey sequence of order n. Let t_1 and t_2 be successive terms of this sequence such that $t_1 < t < t_2$. In this case we always have more than one minimal path between P_1' and P_2'. However, we can always find two minimal paths such that each of the corresponding polygonal arcs consists of just two straight-line segments, having slopes t_1 and t_2 (see Figure 6(a)). Therefore, the length of any minimal path is easily found to be

$$T(P_1', P_2') = \Delta x \left(\frac{t - t_1}{t_2 - t_1} \cdot (1 + t_2^2)^{\frac{1}{2}} + \frac{t_2 - t}{t_2 - t_1} (1 + t_1^2)^{\frac{1}{2}} \right), \qquad t_1 < t < t_2. \tag{6}$$

PROOF. In case (a), the proof is trivial, and eq. (2) holds with equality. We now turn to case (b), where we prove that (i) these paths exist, and that (ii) they are minimal.

(i) Let us consider the points R and S (Figure 6), the vertices of the parallelogram with diagonal $\overline{P_1P_2}$. If we represent the rational numbers t_1, t_2, t as lowest term fractions,

$$t_1 = \frac{a_1}{b_1}, \qquad t_2 = \frac{a_2}{b_2}, \qquad t = \frac{\Delta y}{\Delta x},$$

we have

$$x_R - x_1 = \frac{b_1 b_2 \Delta y - a_1 b_2 \Delta x}{a_2 b_1 - a_1 b_2}, \qquad x_S - x_1 = \frac{a_2 b_1 \Delta x - b_1 b_2 \Delta y}{a_2 b_1 - a_1 b_2};$$

$$y_R - y_1 = \frac{a_2 b_1 \Delta y - a_1 a_2 \Delta x}{a_2 b_1 - a_1 b_2}, \qquad y_S - y_1 = \frac{a_1 a_2 \Delta x - a_1 b_2 \Delta y}{a_2 b_1 - a_1 b_2}.$$

But $a_2 b_1 - a_1 b_2 = 1$ because a_1/b_1 and a_2/b_2 are two successive terms of a Farey sequence [7]. Therefore R and S have integer coordinates, i.e. they correspond to vertices.

(ii) We first show that the function $T(P_1', P_2')$ defined in (6) is a "distance," i.e.:

(I) $T(P_1', P_2') \geq 0$; $T(P_1', P_2') = 0$ iff $P_1' = P_2'$.

(II) $T(P_1', P_2') = T(P_2', P_1')$.

(III) $T(P_1', P_2') \leq T(P_1', Q') + T(Q', P_2')$).

(I) and (II) are obvious.

Remember now that the length of a path is equal to that of the corresponding polygonal arc. We can then easily see (Figure 6(b)) that in (III) equality holds for all points Q internal to the parallelogram $P_1 R P_2 S$; otherwise, strict inequality holds. We call the function $T(P_1', P_2')$ the q.E. (quasi-Euclidean) distance.

Suppose now that we have found the q.E. distance from any vertex P_2' to the vertex P_1'; this distance will satisfy a system of functional equations of type (3). Because the solution of (3) is unique, it suffices to prove that the distance defined in (6) satisfies (3)—in other words, that

$$T(P_1', P_2') = \min_{Q'} (t_{P_2',Q'} + T(P_1', Q')),$$

where Q' is any vertex directly connected to P_2'. But this follows immediately by applying (III) to the vertices P_1', Q', and P_2'.

The function

$$\epsilon = \frac{T(P_1', P_2') - d(P_1, P_2)}{T(P_1', P_2')},$$

where ϵ is given in percent, is shown in Table II for methods 0, 1, 2, 3.

Note that method 2 should normally be satisfactory, since its quasi-Euclidean distance is very close to Euclidean distance. Figure 7(a)–(c) shows the locus of points having a given q.E. distance from a given point for methods 0, 1, 2, respectively. This locus ia a polygon which can be inscribed into the circle consisting of the points having the given Euclidean distance. This can be shown by putting T = constant in (6).

6. *An Iterative Algorithm To Find the Distance From a Figure*

For the solution of the functional system (3) an iterative algorithm has been developed. In the simplest case (method 0), this algorithm coincides exactly with Rosenfeld's. Because of the regularity of the network, it converges in two iterations only, and only half of the vertices connected to each vertex need to be examined in each iteration.

A general method can be outlined as follows (see [9]):

(a) The initial condition may be, for instance, $T_i^0 = t_{iN}$.

(b) The iterative formula is

$$T_i^k = \min_{P_j' \in E_{i1}^k,\ P_r' \in E_{i2}^k} (t_{ij} + T_j^k, t_{ir} + T_r^{k-1}), \qquad i = 1, 2, \cdots, N - 1, \quad (7)$$

where E_{i1}^k is the set of the vertices P_j' for which we have already computed the value T_j^k in the kth iteration, and $E_{i2} = U - E_{i1} \{P_i'\}$. The order in which we

TABLE II

α	$t = tg\alpha$	ϵ percent for method number			
		0	*1*	*2*	*3*
0	0.	0.	0.	0.	0.
2°30′	0.04366	4.267	1.712	0.935	0.613
5°	0.08749	8.335	3.230	1.677	1.034
7°30′	0.13165	12.197	4.551	2.226	1.263
10°	0.17633	15.846	5.674	2.580	1 299
12°30′	0.22169	19.273	6.595	2.739	1.142
15°	0.26795	22.475	7.313	2.702	0.793
17°30′	0.31530	25.442	7.827	2.470	0.251
20°	0.36397	28.171	8.136	2.043	0.157
22°30′	0.41421	30.656	8.239	1.421	0.252
25°	0.46631	32.893	8.136	0.607	0.157
27°30′	0.52057	34.876	7.827	0.251	0.088
30°	0.57735	36.603	7.313	0.792	0.193
32°30′	0.63707	38.069	6.595	1.142	0.107
35°	0.70021	39.273	5.673	1.298	0.200
37°30′	0.76733	40.212	4.551	1.263	0.437
40°	0.83910	40.883	3.229	1.038	0.482
42°30′	0.91633	41.287	1.712	0.613	0.337
45°	1.	41.421	0.	0.	0.

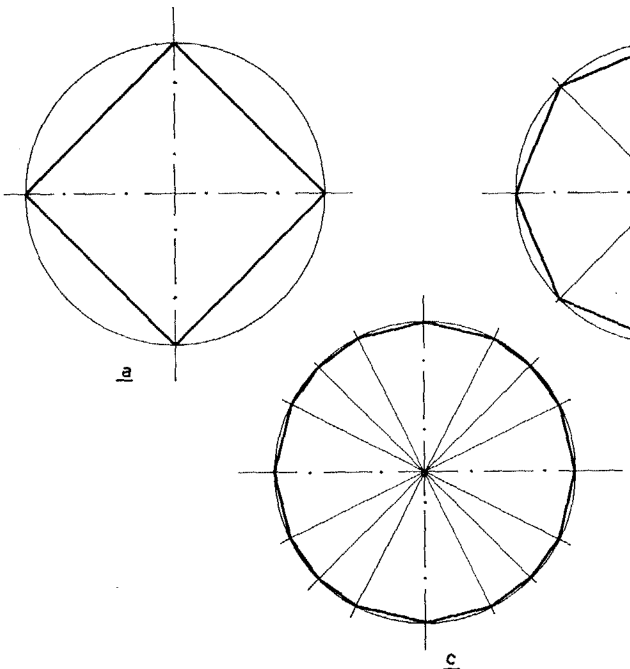


FIG. 7. Loci of points having a given q.E. distance from a given point for methods 0, 1, 2

compute the values T_i^k is arbitrary and may depend on the iteration. However, the convergence rate of the method can be greatly increased by a clever choice of the order (see [9, 10]).

(c) The method stops when we have, for all P_i', $T_i^{m+1} = T_i^m$. In fact, in this case the values T_i^m satisfy the system (3), so we can write

$$T_i^{m+1} = T_i^m = T_i . \tag{8}$$

It is interesting to observe that the values T_i^k form a monotone nonincreasing sequence, i.e.

$$T_i^0 \geq T_i^1 \geq \cdots \geq T_i^m \geq T_i . \tag{9}$$

LEMMA 1. *Assume that we have performed j iterations, and have obtained the sequence*

$$A = (P'_{i_{1,1}}, \cdots, P'_{i_{1,N-1}}, \cdots, P'_{i_{j,1}}, \cdots, P'_{i_{j,N-1}}),$$

where $P'_{i_{r,s}}$ is the s-th vertex for which we computed the value in the r-th iteration. Consider an optimal path from P_N' to P_i', say the sequence

$$(P_N' = P'_{i_0}, P'_{i_1}, \cdots, P'_{i_h} = P_i') = B,$$

where P_{i_k} $(k = 0, \cdots, h-1)$ is directly connected to $P_{i_{k+1}}$. If the points $P_{i_1}, \cdots, P_{i_h}$ appear in order in the sequence A (we will then say that the sequence B is a "subsequence" of the sequence A) then we have

$$T_{i_h}^j = T_{i_h} . \tag{9a}$$

For the proof of this lemma, see Appendix 1.

THEOREM 2. *In our specific case, the points P_i are arranged in a rectangular array. Compute the values T_i^1 in forward raster sequence and T_i^2 in backward raster sequence.*[1] *We then have $T_j^2 = T_j$ for all P_j', i.e. the method converges in two iterations only.*

We can also obtain other simplifications:

(a) We need consider only vertices $P_i' \in E$ (since for $P_i' \in I$, $T_i = 0$).

(b) In computing (7) in the first iteration, there is no need to consider vertices $P_r' \in E_{i2}^1$, since in this case we always have $T_r^0 = \infty$.

(c) We have $E_{11}^1 = \phi$, so also $T_1^1 = \infty$.

(d) We have $E_{i2}^2 = E_{i1}^1 = E_{i1}$ and $E_{i2}^1 = E_{i1}^2 = E_{i2}$, so we can make use of T_i^1 in the second iteration.

The algorithm is then as follows: Let us give to each node P_i' an index corresponding to the forward raster sequence, and:

(a) Let

$$T_i^1 = 0 \quad \text{if} \quad P_i' \in I, \qquad T_1^1 = \infty \quad \text{if} \quad P_1' \in E,$$

$$T_i^1 = \min_{P_j' \in E_{i1}} (t_{ij} + T_j^1) \quad \text{if} \quad P_i' \in E; \qquad E_{i1} = \{P_j' \mid j < i\}.$$

(b) Let

$$T_i^2 = \min_{P_j' \in E_{i2}} (t_{ij} + T_j^2, T_i^1), \qquad i = N-1, \cdots, 1 \quad \text{if} \quad P_i' \in E,$$

$$E_{i2} = \{P_j' \mid j > i\},$$

[1] See [5].

$$T_i^2 = 0 \quad \textit{if} \quad P_i' \in I,$$
$$U = E \cup I = E_{i1} \cup E_{i2} \cup \{P_i'\}.$$

(*c*) *Let*

$$S = \{P_k \mid P_k' \in E; \ \textit{for every } P_i' \textit{ directly connected to} \quad P_k',\, T_k^2 \neq T_i^2 - t_{ik}\}.$$

(*d*) *Associate to every point* $P_k \in S$ *the parameter* T_k^2.

PROOF. Let us consider a vertex P_i' and a minimal path from P_i' to P_N' (or from P_N' to P_i'; it is the same since we are concerned with a nondirected graph). In this path, the vertex directly connected to P_N' will be $Q' \in I$. But between P_i' and Q' there exist two minimal paths (a minimal path) as described in Theorem 1b (a). In fact, even though in this case the network is not infinite, the polygonal arcs corresponding to these minimal paths cannot be cut by the edges of the array, because we always have "allowed directions" parallel to the edges.

Let us divide the allowed directions into two sets D_1 and D_2 (see Figure 8). Suppose that there is more than one minimal path from Q' to P_i'. The two directions of the polygonal arcs corresponding to these minimal paths can belong:

(a) both to D_1,
(b) both to D_2,
(c) one to D_1 and the other to D_2.

In case (a), it is easy to see how the sequence corresponding to the minimal path from P_N' to P_i' is a "subsequence" of the sequence constructed examining the nodes in forward raster sequence; we can therefore apply Lemma 1, and we have reached the solution at the first iteration.

In case (b) we similarly obtain the solution at the second iteration.

In case (c) we can consider the minimal path from P_N' to P_i', of the two available ones, in which we first meet the arcs corresponding to the direction contained in

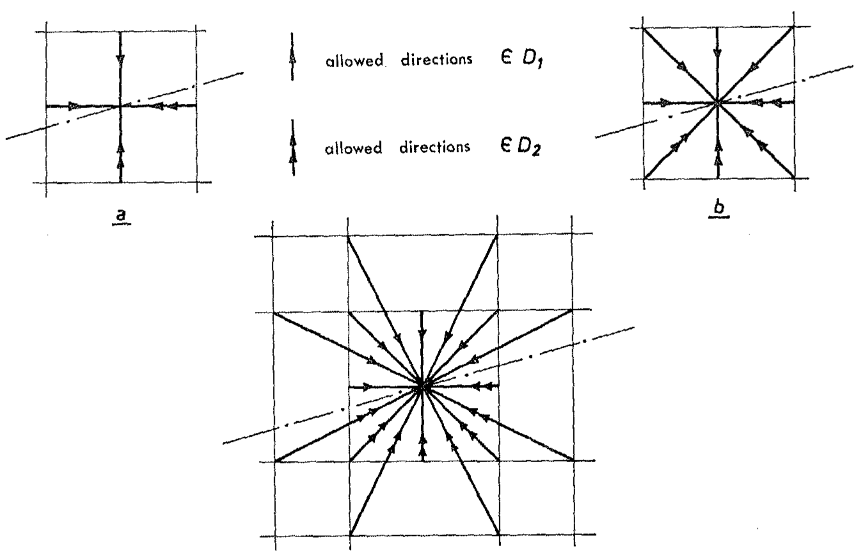
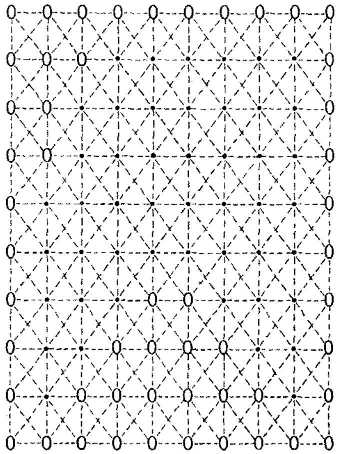


FIG. 8. Allowed directions belonging to D_1 and D_2 are considered in the first and second iteration of the iterative algorithm respectively



(a)

00	00	00	00	00	00	00	00	00	00
00	00	00	10	10	10	10	10	10	00
00	00	10	14	20	20	20	20	14	00
00	00	10	20	28	30	30	28	14	00
00	10	14	24	34	40	40	28	14	00
00	10	20	28	38	48	42	28	14	00
00	10	20	30	00	00	10	20	14	00
00	10	20	00	00	00	00	10	14	00
00	10	00	00	00	00	00	00	10	00
00	00	00	00	00	00	00	00	00	00

(b)

00	00	00	00	00	00	00	00	00	00
00	00	00	10	10	10	10	10	10	00
00	00	10	14	20	20	20	20	10	00
00	00	10	20	28	30	30	20	10	00
00	10	14	24	20	20	24	20	10	00
00	10	20	14	10	10	14	20	10	00
00	10	14	10	00	00	10	14	10	00
00	10	10	00	00	00	00	10	10	00
00	10	00	00	00	00	00	00	10	00
00	00	00	00	00	00	00	00	00	00

(c)

00	00	00	00	00	00	00	00	00	00
00	00	00	10					10	00
00	00	10		20			20		00
00	00		20	28	30	30			00
00	10		24	20		24	20		00
00		20					20		00
00	10	14	10	00	00	10	14	10	00
00	10	10	00	00	00	00	10	10	00
00	10	00	00	00	00	00	00	10	00
00	00	00	00	00	00	00	00	00	00

(d)

FIG. 9. An example of application of the iterative algorithm: method 1

D_1 and then the arcs corresponding to the direction contained in D_2. In this way, we obtain a suitable sequence B. Finally, suppose that there is only one minimal path from Q' to P_i'; then either case (a) or case (b) holds.

Thus by utilizing both iterations we can apply Lemma 1 to all the P_i' and obtain the solution in all cases.

Figure 9 shows an example of the application of this algorithm. In Figure 9(a) we have the reticular network corresponding to method 1, where the vertices directly connected to P_N', i.e. corresponding to some $P_i' \in I$, are countersigned. In Figure 9(b) the values of T_i^1 are shown; in Figure 9(c) the values of $T_i^2 = T_i$ are shown. In Figure 9(d) we see the skeleton points with the corresponding parameters. Here and in the following figures, we show only the integer part of the distance, assuming the unit length to be 10.

7. *A Simplified Dantzig Algorithm To Obtain the Skeleton*

In this section, a simplified Dantzig algorithm for obtaining the skeleton directly is presented. Each vertex is considered only once. The distances are computed in order, beginning from the smallest ones, so that we can interpret this algorithm as a discrete wavefront propagation. In what follows, the general Dantzig method for optimal path determination is described (see [9]).

(1) In the first step, we look for the vertex (or vertices) P_k' such that $t_{kN} = \min t_{jN}$. We can immediately write $T_N = 0$, $T_k = t_{kN}$. We then define the sets $E_1^2 = \{P_N\}$, $E_2^2 = U - E_1^2$.

(2) We have to compute the values

$$T_h^m = \min_{P_x' \in E_1^m} (t_{hx} + T_x) \quad \text{for every} \quad P_h' \in E_2^m. \tag{10}$$

Dantzig proved that, for the vertex (or vertices) P_k' such that $T_k^m = \min_{P_h' \in E_2^m} T_h^m$, we have already obtained the minimal length, i.e.

$$T_k^m = \min_{P_h' \in E_2^m} T_h^m = T_k. \tag{11}$$

The sets E_1 and E_2 for the next step are given by $E_1^{m+1} = E_1^m + \{P_k'\}$, $E_2^{m+1} = E_2^m - \{P_k'\}$. That means that the set E_1^m is the set of vertices for which, at the beginning of the mth step, we have already computed the distance.

(3) The algorithm stops at the step r such that E_2^{r+1} is empty. With this algorithm, vertices are processed according to their distances; that is, if at the mth step we have $P_i \in E_1^m$ and $P_j \in E_2^m$, i.e. if P_j will be accepted after P_i, we can conclude at once that

$$T_i < T_j. \tag{12}$$

In our application, it is interesting to point out how, with this algorithm, we also determine the optimal policies in quite a natural manner. In fact, to find the optimal policy (or policies) for the vertex (or vertices) P_k' accepted in the mth step, it suffices to remember the vertex (or vertices) P_x' for which $(t_{kx} + T_x)$ is the least in (10). The optimal policy is (or policies are), the arc (or arcs) connecting P_k' and P_x'. Therefore, we can countersign all the vertices P_x' that we meet. When the algorithm stops, the noncountersigned vertices belonging to the set E form the skeleton.

We now examine in detail this algorithm for method 0. In this case, the distances can assume integer values only.

THEOREM 3. *In method* 0, (*i*) *the vertices P_k' accepted in the m-th step are all those belonging to the set E_2^m and directly connected with some vertex belonging to E_1^m;* (*ii*) *the distance of all these vertices is m.*

PROOF. Let P_{k_m}' be the vertex (or vertices) accepted in the mth step.

(i) From inequality (12) we can write $T_{k_{m-1}} = \max(T_x)$, $\quad P_x' \in E_1^m$. Then from (10),

$$T_h^m \leq 1 + T_{k_{m-1}} \tag{13}$$

for every $P_h' \in E_2^m$ and directly connected to a node $P_x' \in E_1^m$, but from (11),

$$T_{k_m} \leq T_h^m. \tag{14}$$

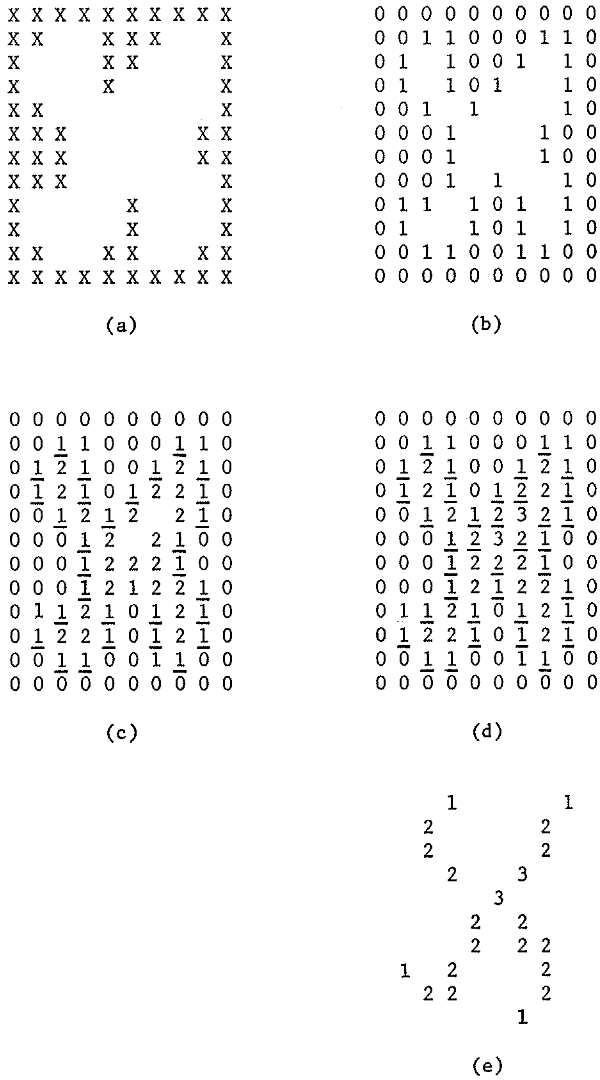


FIG. 10. An example of application of the Dantzig algorithm: method 0

Therefore from (13) and (14), $T_{k_m} \leq 1 + T_{k_{m-1}}$, but from (12), $T_{k_m} > T_{k_{m-1}}$. But the values T can only be integers; therefore from the last two inequalities we obtain

$$T_{k_m} = T_{k_{m-1}} + 1. \tag{15}$$

But from (13) and (14), $T_{k_m} = T_h{}^m$, that is, all the nodes $P_h{}'$ are accepted.

(ii) We have $T_{k_1} = 1$. Therefore from the finite-difference equation (15) we have $T_{k_m} = m$.

This theorem allows us to obtain easily the distance of every vertex for method 0. It suffices to find the vertices belonging to the boundary of the set[2] E, erase them from the set E, and iterate the procedure: the points erased in the mth iteration correspond to nodes whose distance is m. (See the example in Figure 10.)

[2] This result can be accomplished, for instance, with the algorithm described by Ledley in [11].

A simpler algorithm can be found for methods greater than 0 as well, with the aid of the following theorem. Let CE_2^m be the "boundary" of the set E_2^m, i.e.

$$CE_2^m = \{P_i' \mid P_i' \in E_2^m; \exists Q', T(P_i', Q') = 1 \text{ .AND. } Q' \notin E_2^m\}.$$

THEOREM 4. *Assume that the Dantzig general algorithm has been applied through the $(m - 1)$-st step. Then*

(*a*) *All the points accepted at the m-th step belong to CE_2^m.*

(*b*) *The distances of two nodes, both belonging to some CE_2^m, differ by less than* 1.

(*c*) *For the vertices P_h' belonging to CE_2^m and not accepted at the m-th step we can write*

$$T_h^m = T_h. \tag{16}$$

For the proof of this theorem, see Appendix 2.

The simplified Dantzig algorithm is then:

(1) $E_1^1 = I, \quad E_2^1 = E.$

(2) At the beginning of the mth step we already have the sets E_1^m and E_2^m. Therefore we can (a) construct the set CE_2^m, for instance using Ledley's algorithm [11]. We then (b) have to compute the values

$$\min_{P_x' \in E_1^m} (t_{hx} + T_x) = T_h^m = T_h$$

for the vertices P_h' belonging to $CE_2^m - (CE_2^m \cap CE_2^{m-1})$ (we can assume $CE_2^0 = \phi$). In fact, we remember the values $T_h^{m-1} = T_h^m = T_h$ for the nodes P_h' belonging to $CE_2^m \cap CE_2^{m-1}$ from the previous iterations. (c) We also have to countersign the vertices P_x' for which $(t_{hx} + T_x)$ is the least, i.e. for which $T_x = T_h - t_{hx}$. Among the vertices $P_h'(d)$ we look for the vertices P_k' such that $T_k = \min T_h$ where $P_h' \in CE_2^m$ (and not $P_h' \in E_2^m$, as in the general Dantzig algorithm). We can now construct the sets

$$E_1^{m+1} = E_1^m + \{P_k'\}, \qquad E_2^{m+1} = E_2^m - \{P_k'\}$$

that will be needed in the next iteration.

(3) When $E_2^r = \phi$, the noncountersigned vertices P_S' form the set S; the corresponding distance is T_S (see Figure 11).

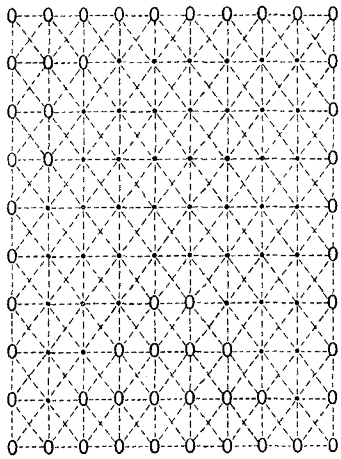
With this algorithm, each vertex is examined only once, since as soon as it is reached by the boundary we can compute the definitive distance value.

The property proved in Theorem 4, part (b) means that the set CE_2^m can be seen as a wavefront. Therefore this algorithm realizes the discrete propagation of the figure contour.

8. *The Inverse Transformation: How To Obtain the Figure, Given the Skeleton*

Blum [1] proved that it is possible to achieve the following result: If one excites every skeleton point at a negative time equal to the corresponding parameter, at $t = 0$ the wavefront is the contour. In the discrete case too it is possible to prove that given the skeleton, the points of the figure can be determined by the solution of a minimal path problem. Moreover, for the solution of this problem, the same iterative algorithm described in Section 6 can be used.

We first prove the following almost obvious lemma.



(a)

00	00	00	00	00	00	00	00	00	00
00	00	00	10	10	10	10	10	10	00
00	00	10						10	00
00	00	10						10	00
00	10							10	00
00	10			10	10			10	00
00	10		10	00	00	10		10	00
00	10	10	00	00	00	00	10	10	00
00	10	00	00	00	00	00	00	10	00
00	00	00	00	00	00	00	00	00	00

(b)

00	00	00	00	00	00	00	00	00	00
00	00	00	10	10	10	10	10	10	00
00	00	10	14	20	20	20	20	10	00
00	00	10	20				20	10	00
00	10	14		20	20		20	10	00
00	10	20	14	10	10	14	20	10	00
00	10	14	10	00	00	10	14	10	00
00	10	10	00	00	00	00	10	10	00
00	10	00	00	00	00	00	00	10	00
00	00	00	00	00	00	00	00	00	00

(c)

00	00	00	00	00	00	00	00	00	00
00	00	00	10	10	10	10	10	10	00
00	00	10	14	20	20	20	20	10	00
00	00	10	20				20	10	00
00	10	14	24	20	20	24	20	10	00
00	10	20	14	10	10	14	20	10	00
00	10	14	10	00	00	10	14	10	00
00	10	10	00	00	00	00	10	10	00
00	10	00	00	00	00	00	00	10	00
00	00	00	00	00	00	00	00	00	00

(d)

00	00	00	00	00	00	00	00	00	00
00	00	00	10	10	10	10	10	10	00
00	00	10	14	20	20	20	20	10	00
00	00	10	20	28	30	30	20	10	00
00	10	14	24	20	20	24	20	10	00
00	10	20	14	10	10	14	20	10	00
00	10	14	10	00	00	10	14	10	00
00	10	10	00	00	00	00	10	10	00
00	10	00	00	00	00	00	00	10	00
00	00	00	00	00	00	00	00	00	00

(e)

00	00	00	00	00	00	00	00	00	00
00	00	00	10					10	00
00	00	10		20			20		00
00	00		20	28	30	30			00
00	10		24	20		24	20		00
00		20					20		00
00	10	14	10	00	00	10	14	10	00
00	10	10	00	00	00	00	10	10	00
00	10	00	00	00	00	00	00	10	00
00	00	00	00	00	00	00	00	00	00

(f)

FIG. 11. An example of application of the Dantzig algorithm: method 1. P' belongs to E_1^{m+1}; $\not{P}'$ is countersigned

Lemma 2. *Let G be an undirected graph, and let $T(P_i', P_j')$ be the minimal path length between the vertex P_i' and the vertex P_j'. Now adjoin a new vertex P_k' and compute in the new graph G^k the minimal length $T^k(P_i', P_j')$. Then*

(a) $T^k(P_i', P_j') \leq T(P_i', P_j')$;

(b) If P_i' belongs in G^k to a minimal path between P_k' and P_j', all the minimal paths between P_i' and P_j' in G and G^k coincide.

Proof. (a) G is a subgraph of G^k.

(b) A minimal path between P_i' and P_j' certainly does not contain P_k'. Therefore a corresponding path exists in G, and (a) holds with equality. Conversely, a minimal path in G must be minimal in G^k too, since otherwise equality does not hold in (a).

Let be C an integer constant such that $C > \max_{P_i \in S} T_i$, where S, as usual, is the skeleton point set. Now connect an extra vertex P_M' to every skeleton node P_S' with an arc of length $t_{SM} = C - T_S$. Let G^N, G^M, G^{MN} be the networks with extra nodes P_N', P_M', and $P_N' \; P_M'$, respectively. Let $T^N(P_i', P_j')$, $T^M(P_i', P_j')$, $T^{MN}(P_i', P_j')$ be the minimal path lengths between P_i' and P_j' computed in G^N, G^M, G^{MN}, respectively.

Theorem 5. *Let $T_i = T^N(P_i', P_N')$, $T_i' = T^M(P_i', P_M')$. Then:*

(a) If $P_i' \in E$ we have $T_i' = C - T_i$.

(b) If $P_i' \in I$ we have $T_i \geq C$.

From the skeleton we can thus obtain the sets E, I again. (We have introduced the additive constant C, since otherwise we would have arcs of negative length.)

Proof. We compute $T^{MN}(P_M', P_N')$. P_M' is directly connected with skeleton vertices only. Therefore a skeleton vertex P_r' will belong to an optimal path between P_N' and P_M':

$$T^{MN}(P_M', P_N') = t_{rM} + T^{MN}(P_r', P_N').$$

We can apply Lemma 2(b):

$$T^{MN}(P_r', P_N') = T^N(P_r', P_N') = T_r .$$

Therefore we have

$$T^{MN}(P_M', P_N') = t_{rM} + T_r = C - T_r + T_r = C.$$

But a path from P_N' to P_M' of length C touches every skeleton vertex P_S':

$$T_S + t_{SM} = C \quad \text{for every} \quad P_S' \in S.$$

Therefore every skeleton vertex belongs to a minimal path between P_N' and P_M', while every nonskeleton vertex contained in E belongs to a minimal path in G^N from some $P_S' \in S$ and P_N', by definition of the skeleton. But minimal paths between every $P_S' \in S$ and P_N' coincide in G^N and G^{MN} by Lemma 2(b). Therefore every vertex contained in E belongs to a minimal path from P_N' to P_M'. We can write

$$C = T^{MN}(P_M', P_N') = T^{MN}(P_M', P_i') + T^{MN}(P_i', P_N').$$

Applying Lemma 2(b) twice,

$$C = T^M(P_M', P_i') + T^N(P_i', P_N') = T_i' + T_i .$$

If $P_i \in I$, we have $t_{iN} = 0$. Therefore $T^{MN}(P_i', P_M') = T^{MN}(P_N', P_M') = C$; and from Lemma 2(a), we have $T^M(P_i', P_M') \geq C$.

To compute the values T_i' it suffices to apply the iterative algorithm of Section 6 with initial values $T_i^0 = C - T_S$ if $P_i \in S$, $T_i^0 = \infty$ otherwise.

9. *Extracting the Most Important Skeleton Branches*

In our skeleton, nonsignificant points can appear that might make it difficult to utilize this method for pattern recognition purposes. In this section, a criterion is proposed for determining the most significant skeleton branches. This criterion depends upon a parameter that is essentially a threshold. It can be applied, with slight modifications, to the algorithms presented in Sections 6 and 7.

Let us consider a skeleton in the continuous real plane. Often we can single out "branches," that is, pieces of skeleton having the form of continuous curves. In this case, we can take the derivative along the curve:

$$g(P) = \lim_{P_1 \to P} \left| \frac{d(P_1) - d(P)}{\Delta S} \right|,$$

where $d(P)$ and $d(P_1)$ are the skeleton parameters at P and P_1, and ΔS is the arc length of skeleton branch P_1P. If we consider the skeleton as the intersection of propagating wavefronts (Section 2), $v(P) = 1/g(P)$ has the meaning of the speed at which, in P, the given skeleton branch is generated.

Blum [1] suggested that the most significant skeleton branches, from a perceptual point of view, might be the ones generated with the highest speed. The extrema of skeleton branches, where the branches stop or are connected with other branches, and the isolated points, are also very important. A limiting speed can thus be fixed, and one can accept only the detached points, and the skeleton branch pieces with speeds greater than the threshold value.

For example, Figure 12 shows the same case as in Figure 1(a), together with the values of $v(P)$. We can see that this parameter is infinite at the vertex of the hyperbola, and decreases asymptotically toward 1. In fact, we tacitly assumed the wave-

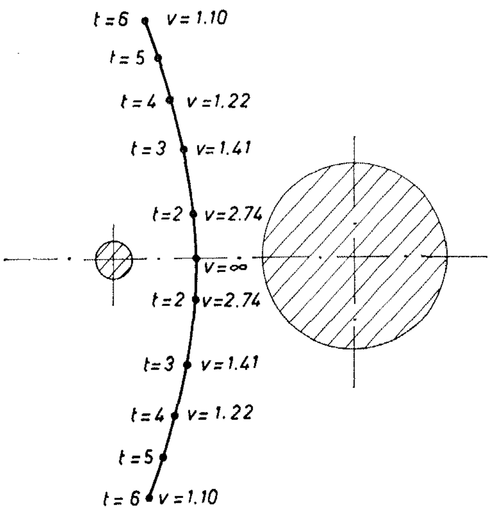


FIG. 12. A skeleton branch and its propagation speed

front propagation velocity to be one. It is clear that the most important part of this skeleton is the central one.

In order to extend this criterion to the discrete case it is convenient to give another equivalent definition of $g(P)$. Let us compute for every point P the distance $d(P)$ from the figure, and the directional derivative

$$f(P, \alpha) = \lim_{P_1 \to P} \frac{d(P_1) - d(P)}{\Delta l},$$

where Δl is the length of the segment $\overline{PP_1}$, and where the oriented segment PP_1 forms an angle α with a reference axis. Let

$$f(P) = \max_{\alpha} f(P, \alpha).$$

It can be seen that:

(a) If P does not belong to the skeleton, $f(P) = 1$ and conversely, so that the function $f(P)$ could be used to define the skeleton.

(b) If P is a "departure" point (i.e. a nearest point to the figure), or a midpoint of a skeleton branch, $f(P)$ is equal to the inverse $1/v(P)$ of the speed defined above.

(c) If P is an isolated point or an "arrival" point (i.e. a farthest point from the figure) of a skeleton branch, $f(P)$ is negative.

In the discrete case, we replace the distances by minimal path lengths, and the derivatives by finite difference ratios:

$$f_i = \max_{P_j} \frac{T_j - T_i}{t_{ij}};$$

P_j directly connected to P_i, where T_j, T_i, t_{ij} have the usual meaning. From (3) it follows that $-1 \leq f_i \leq 1$.

Instead of the definition of skeleton given in Section 4, we now propose the following:

Definition. The point P_i is a skeleton point if and only if $P_i' \in E$ and $-1 \leq f(P_i) < K \leq 1$, where K is an assigned threshold value.

In order to avoid in computing as many divisions as network arcs, the following equivalent definition is preferable:

$$S = \{P_i \mid P_i' \in E; \text{for every } P_j', T_j - T_i < K \cdot t_{ij}\}. \tag{17}$$

If $K = 1$ this definition coincides with the definition given in Section 4. In order to obtain this reduced skeleton with the iterative algorithm, it suffices to replace step (c) of the final algorithm described in Section 6 by definition (17). To obtain the same result with the simplified Dantzig algorithm described in Section 7, we must change step 2(c). Instead of countersigning the vertices for which $T_x = T_h - t_{hx}$, we must countersign the vertices for which $T_x \leq T_h - K \cdot t_{hx}$.

In Figure 13 we show the skeleton of a digitized circle:

(a) Method 1 without reduction ($K = 1$).

(b) Method 1 with elimination of the less significant skeleton points using the threshold $K = 0.90$.

(c) Same as (b), but with $K = 0.80$.

(d) Same as (b), but with $K = 0.70$.

```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 S . . . S 0 0 0 0 0
0 0 0 S . . S . S . . S 0 0 0
0 0 S . . . . . . . . . S 0 0
0 0 . . . . . . . . . . . 0 0
0 S . . . . S . S . . . . S 0
0 . S . . S S . S S . . S . 0
0 . . . . . . S . . . . . . 0
0 . S . . S S . S S . . S . 0
0 S . . . . S . S . . . . S 0
0 0 . . . . . . . . . . . 0 0
0 0 S . . . . . . . . . S 0 0
0 0 0 S . . S . S . . S 0 0 0
0 0 0 0 0 S . . . S 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

(a)

```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 S . . . S 0 0 0 0 0
0 0 0 S . . S . S . . S 0 0 0
0 0 S . . . . . . . . . S 0 0
0 0 . . . . . . . . . . . 0 0
0 S . . . . S . S . . . . S 0
0 . S . . S . . . S . . S . 0
0 . . . . . . S . . . . . . 0
0 . S . . S . . . S . . S . 0
0 S . . . . S . S . . . . S 0
0 0 . . . . . . . . . . . 0 0
0 0 S . . . . . . . . . S 0 0
0 0 0 S . . S . S . . S 0 0 0
0 0 0 0 0 S . . . S 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

(b)

```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 S . . . S 0 0 0 0 0
0 0 0 S . . . . . . . S 0 0 0
0 0 S . . . . . . . . . S 0 0
0 0 . . . . . . . . . . . 0 0
0 S . . . . . . . . . . . S 0
0 . . . . . . . . . . . . . 0
0 . . . . . . S . . . . . . 0
0 . . . . . . . . . . . . . 0
0 S . . . . . . . . . . . S 0
0 0 . . . . . . . . . . . 0 0
0 0 S . . . . . . . . . S 0 0
0 0 0 S . . . . . . . S 0 0 0
0 0 0 0 0 S . . . S 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

(c)

```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 . . . . . 0 0 0 0 0
0 0 0 . . . . . . . . . 0 0 0
0 0 . . . . . . . . . . . 0 0
0 0 . . . . . . . . . . . 0 0
0 . . . . . . . . . . . . . 0
0 . . . . . . . . . . . . . 0
0 . . . . . . S . . . . . . 0
0 . . . . . . . . . . . . . 0
0 . . . . . . . . . . . . . 0
0 0 . . . . . . . . . . . 0 0
0 0 . . . . . . . . . . . 0 0
0 0 0 . . . . . . . . . 0 0 0
0 0 0 0 0 . . . . . 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

(d)

FIG. 13. The skeleton of a digitized concave circle. Successive thresholds reduce the skeleton to the center only

10. *Nonsignificant Skeleton Points Close to Matrix Boundary*

Often on the matrix boundary there are elements external to the figure. In this case, some elements near the matrix boundary usually become meaningless skeleton points. For example, the external skeleton of a triangle in the real plane is empty, as usual for a convex figure [1]. In Figure 14 are shown the distances and the external skeleton of a discretized triangle (method 1). In this case, the skeleton is not empty. The vertices corresponding to points $P_i \in S$ on the matrix boundary in Figure 14 do not belong to a minimal path of any other vertex, only because the network is not infinite.

In this section a method of eliminating the skeleton noise points close to the matrix boundary is described. In Figure 14, all the skeleton points in the first and last row and in the first and last column have no meaning. In the general case, suppose a digitized picture given in the form of a rectangular matrix of elements a_{ij} $(i = 1, \cdots, r;\ \ j = 1, \cdots, s)$. Interpret (i, j) as the coordinates x_P and y_P of a point P in the real plane, and construct the skeleton S in the usual manner. Now add to the reticulum, as external vertices, the vertices corresponding to all the remaining points of the plane with integer coordinates, and compute the skeleton S^* in this

14	10	10	14	24	34	42	52
10	0	0	10	20	28	38	48
10	0	0	10	14	24	34	42
10	0	0	0	10	20	28	38
10	0	0	0	10	14	24	34
10	0	0	0	0	10	20	28
10	0	0	0	0	10	14	24
10	0	0	0	0	0	10	20
14	10	10	10	10	10	14	24

(a)

S	S	S	.	.	S	.	S
S	0	0	.	.	.	.	S
S	0	0	S	.	.	.	S
S	0	0	0	.	.	.	S
S	0	0	0	S	.	.	S
S	0	0	0	0	.	.	S
S	0	0	0	0	S	.	S
S	0	0	0	0	0	.	S
S	S	S	S	S	S	.	S

(b)

FIG. 14. The distance transformation and the skeleton of a digitized convex triangle

infinite reticulum. One could ask what is the difference between the skeleton S and the subset RS^* of the skeleton S^* included in the original bounds of the matrix. The answer is given in the following theorem.

THEOREM 6. (*a*) *The distances of all the vertices corresponding to points included in the original matrix are the same in both cases.*

(*b*) $RS^* \subseteq S$, *and all the points of the set* $S - RS^*$ *can belong only to a frame* B_n *of the original matrix, whose width is* 1 *in the case of method* 0, *and is* n *in method* n ($n \neq 0$).

PROOF. Let us define the following sets:

$$U = \{P' \mid 1 \le x_p \le r;\ 1 \le y_p \le c;\ x_p,\ y_p \text{ integers}\}$$
$$I = \{P' \mid a(P) = .\text{TRUE.}\}$$
$$E = \{P' \mid a(P) = .\text{FALSE.}\} = U - I$$
$$S = \{P_k \mid P_k' \in E;\ \text{for every } P_i',\ T_k \neq T_i - t_{ik}\}$$
$$U^* = \{P' \mid x_p,\ y_p \text{ integers}\}$$
$$I^* = I$$
$$E^* = U^* - I$$
$$S^* = \{P_k \mid P_k' \in E^*;\ \text{for every } P_i',\ T_k^* \neq T_i^* - t_{ik}\}$$
$$RS^* = \{P \mid P \in S^*;\ P' \in U\}$$
$$B_n = \{P \mid 1 \le x_p \le m \text{ .OR. } r - m < x_p \le r \text{ .OR. } 1 \le y_p \le m \text{ .OR. } c - m < y_p \le c;\ \text{if } n = 0 \text{ then } m = 1 \text{ else } m = n;\ P' \in U\}$$

We must prove that

(a) $T_i = T_i^*$ for every $P_i' \in U$;

(b) $S \cap (U - B_n) = RS^* \cap (U - B_n), \quad S \cap B_n \supseteq RS^* \cap B_n$.

(i) Suppose given for P_i' a minimal path to the figure as described in Theorem 1, computed in the infinite reticulum. Let Q' be a vertex contained in I^* belonging to this path. If $P_i' \in U$, this is a minimal path in the finite case as well. In fact, in this case both P_i' and Q' belong to the finite reticulum ($I = I^*$), and therefore the whole minimal path belongs to the finite reticulum, since in every method we have connecting segments parallel to the edges of the matrix.

(ii) In determining S and RS^* apply the definitions: if $P_k' \in U - B_n$, the vertices connected to P_k' are the same in both the infinite and the finite reticulum, as are the distances by (i); hence the result is the same. If $P_k' \in B_n$, the vertices connected to P_k' are fewer in the finite reticulum than in the infinite one, so if the relation "$T_k \neq T_i - t_{ik}$ for every P_i" holds in the infinite reticulum, it certainly holds in the finite one too, but not conversely.

In order to obtain the skeleton RS^* using the algorithms of Sections 6 and 7, it suffices (a) to enlarge the given matrix by adding a frame of width 1 in the method 0, and n in the method n (with a_{ij} = .FALSE.), (b) to compute the skeleton in the usual manner, and (c) to erase the frame.

Let us now return to Figure 14. We see that we have skeleton noise points not only on the boundary of the matrix, but inside too, along the contour of the figure. The reason for this unwanted result is very simple. In the real plane, a point on the contour where we cannot define the tangent line to the contour is always a point where the skeleton touches the contour. In the discrete case, a similar situation is realized whenever the contour points do not lie in a line, since in a digitized figure curves are impossible. Note that this does not happen in the method 0; in fact, in this case the discretized contour does not introduce noise, since it looks smooth with the metric used.

In order to eliminate these skeleton noise points for methods greater than zero, it usually suffices to use the method described in Section 9 with a threshold value $K \leq 0.70$. (See, for instance, Figure 13.)

11. *Two Computer Programs: Description and Comparison*

In this section, a short description is given of FORTRAN IV programs which realize the two algorithms described in Sections 6 and 7 and modified in Sections 9 and 10, for the methods 0, 1, 2, 3. These programs are available at the Istituto di Elettrotecnica ed Elettronica, Politecnico di Milano, Milano, Italy.

Programming the iterative method of Section 6 is very simple, since a raster sequence can be obtained by simply using two "DO" statements internal one to the other.

In the Dantzig algorithm, we accept, at every step, only the vertices not yet accepted which have minimal distance. In this case we should therefore have a major reordering problem. In our program, reordering is automatically performed by a list structure. At the beginning of the mth step, the list structure contains the names and the distances of all the vertices $I^m = CE_2^{m-1} \cap CE_2^m$, i.e. belonging to CE_2^{m-1} and not accepted at the $(m - 1)$-st step. (Here the names could be, for instance, the coordinates of the corresponding points.) Usually, many vertices belonging to CE_2^m have the same distance. The names of all the vertices having the same distance are arranged in a list, while the distance is stored in the head of the list. The heads, ordered according to the distance, are then arranged in a two-way ring list. In this new list, a head of the heads with a permanent, known address is the successor of the element corresponding to the minimal distance, and, at the same time, the predecessor of the element corresponding to the maximal one. The vertices belonging to E_1^m, I^m, $E_2^m - I^m$ are distinguished by different marks on the original array.

In order to obtain the set CE_2^m, it suffices to add to the list structure the vertices contained in $E_2^m - I^m$ directly connected with an arc of length 1 to a vertex accepted at the $(m - 1)$-st step. By Theorem 4, the distances of these vertices can be computed using (10). If we want to put one of these vertices into this structure, we must traverse the ring list beginning from the farthest element, until we find a head with a distance equal to the distance of our vertex; the name of this vertex is then added to this list. If such a head is not found, a new list is initialized, and its head inserted into the ring list in the proper position. In this manner, all the points to be accepted in the nth step are in the list whose head is the predecessor of the head of the heads. This list and its head can then be erased, and the corresponding memory positions added to the free list. The list is then ready for the $(m + 1)$-st step. The computing times required by the two programs, for the same method, are comparable. The Dantzig algorithm is somewhat faster, especially with method 0, for figures having a long perimeter and a small area.

Increasing the method by 1 increases the computing time by a factor of about 1.7. If we consider the improvement in the approximation to Euclidean distance (Section 5), it appears that the best methods are 0, 1, 2.

12. *Conclusion*

A definition of the skeleton of a digitized figure is given, which permits a close approximation of the properties of the skeleton of a continuous picture in the real plane, especially invariance under rotation. Two algorithms are developed; one of them is especially suitable for long-perimeter, small-area figures such as chromosomes. A quantitative definition of "significant" skeleton points is given, depending upon a parameter. With a suitable choice of this parameter, only the most important figure features are preserved in the skeleton, so that its application to pattern recognition can be made easier. An algorithm for performing the inverse transformation is also developed. Using this algorithm, a prototype of the figures belonging to a given class can be obtained from the reduced skeleton.

APPENDIX 1. *Proof of Lemma* 1

We use induction to prove (9a), (i) for $h = 1$, (ii) for $h = l$ assuming it holds for $h < l$.

(i) The sequence $(P_{i_0} = P_N, P_{i_1})$ is an optimal path; that is $t_{i_1,N} = T_{i_1}$. But the initial condition is $T_{i_1}^0 = t_{i_1,N}$, so for (9) we have $T_{i_1}^0 = \cdots = T_{i_1}^j = T_i$.

(ii) Let $P'_{i_{k,s}}$ be the element belonging to the sequence A corresponding to the element $P'_{i_l} \in B$, and $P'_{i_{u,v}}$ that corresponding to $P'_{i_{l-1}}$. $k \geq u$; if $k = u$ then $s > v$ because, for hypothesis, $P'_{i_{u,v}}$ is considered before $P'_{i_{k,s}}$ in the iterative process. Using (9a) for $h = l - 1$ we obtain $T_{i_{l-1}}^u = T_{i_{l-1}}$. After having applied (7) in the kth iteration for the vertex $P'_{i_{k,s}} = P'_{i_l}$ we have $T_{i_l}^k \leq T_{i_{l-1}} + t_{i_l,i_{l-1}}$. But the sequence B corresponds to an optimal path, so $T_{i_l} = T_{i_{l-1}} + t_{i_l,i_{l-1}}$; consequently $T_{i_l}^k = T_{i_l}$.

APPENDIX 2. *Proof of Theorem* 4

(a) Let P_j' be a vertex belonging to E_2^m but not to CE_2^m. Let

$$T_j^m = \min_{P_x' \in E_1^m} (t_{jx} + T_x) = t_{ji} + T_i, \qquad P_i' \in E_1^m.$$

Let the coordinates of the corresponding points P_j and P_i in the real plane be

(x_j, y_j) and (x_i, y_i) and, as usual,

$$t = \frac{y_j - y_i}{x_j - x_i}, \qquad 0 \leq t \leq 1, \qquad x_j > x_i.$$

The case $t = 0$ is impossible, because we have assumed that P_j' does *not* belong to CE_2^m.

Let $x_k = x_j - 1$, $y_k = y_j$; then $P_k' \in E_2^m$, since otherwise P_j' would belong to CE_2^m. Moreover P_j' is directly connected to P_i', since otherwise $t_{ji} = \infty$. This means that P_j is inside the square of side $(2n + 1)$ centered at P_i (where we are using method n). It can be easily seen that P_k is also inside, and therefore

$$T(P_i', P_k') = ((x_j - x_i - 1)^2 + (y_j - y_i)^2)^{\frac{1}{2}} < ((x_j - x_i)^2 + (y_j - y_i)^2)^{\frac{1}{2}} = t_{ij},$$

since all the points within the square have Euclidean distances from P_i. If P_i and P_k are directly connected, we have $T(P_i', P_k') = t_{ik}$, so $T_k^m \leq T_i + t_{ik} < T_i + t_{ij} = T_j^m$, and P_j is not accepted in the mth iteration. But if P_i and P_k are not directly connected, they are connected in a straight line; let $P_r' \in E_1^m$ and $P_s' \in E_2^m$ be two directly connected points which lie on the straight-line segment connecting P_i and P_k. Then

$$T_s^m \leq T_r + t_{rs} \leq T_i + T(P_i', P_r') + t_{rs}$$
$$= T_i + T(P_i', P_s') \leq T_i + T(P_i', P_k') < T_i + t_{ij} = T_j^m$$

and P_j is not accepted in any case.

(b) Let $$T_i^m = \max_{P_h' \in CE_2^m} T_h^m, \qquad T_j^m = \min_{P_h' \in CE_2^m} T_h^m.$$

If $P_i' \in CE_2^m$, there will be a vertex $P_k' \in E_1^m$ such that $t_{ik} = 1$. Therefore $T_i^m \leq T_k + 1$; but through the $(m - 1)$-st step we have used the complete Dantzig method, so we can write (12) $T_j^m > T_k$, and we have $T_i^m - T_j^m < 1$.

(c) If a vertex $P_h' \in CE_2^m$ is not accepted in the mth step, we have

$$T_h^m = \min_{P_x' \in E_1^m} (t_{hx} + T_x), \qquad P_h' \in CE_2^m$$

$$T_h^{m+1} = \min_{P_x' \in E_1^{m+1}} (t_{hx} + T_x), \qquad P_h' \in CE_2^{m+1},$$

but

$$E_1^{m+1} = E_1^m + \{P_k\} \quad \text{where} \quad T_k = T_k^m = \min_{P_h' \in CE_2^m} T_h^m,$$

so we can write $T_h^{m+1} = \min(T_h^m, T_k + t_{hk})$. For (b) we have $T_h^m < T_k + 1 \leq T_k + t_{hk}$, so $T_h^{m+1} = T_h^m$; and if P_h' is accepted in the $(m + r)$-th step, then $T_h = T_h^{m+r} = T_h^{m+r-1} = \cdots = T_h^m$.

ACKNOWLEDGMENT. The author is indebted to Professor Azriel Rosenfeld for his valuable work on the submitted manuscript.

REFERENCES

1. BLUM, H. A transformation for extracting new descriptors of shape. In Wathen-Dunn, Weiant (Ed.), Symp. on Models for the Perception of Speech and Visual Form (Boston, Nov. 1964), M.I.T. Press, Cambridge, Mass.

2. Kotelly, J. A mathematical model of Blum's theory of pattern recognition. Air Force Cambridge Res. Labs. Res. Rep. 63-164, April 1963 (AD 412437).
3. Calabi, L. On the shape of plane figures. Air Force Cambridge Res. Labs. Res. Rep. SR-1-60429, March 1965.
4. ——. A study of the skeleton of plane figures. Air Force Cambridge Res. Labs. Res. Rep. SR-2-60429, June 1965.
5. Rosenfeld, A., and Pfaltz, J. L. Sequential operations in digital picture processing. *J. ACM 13*, 4 (Oct. 1966), 471–494.
6. Pfaltz, J. L., and Rosenfeld, A. Computer representation of planar regions by their skeletons. *Comm. ACM 10* (1967), 119–125.
7. Rademacher, H. *Lectures on Elementary Number Theory*. Blaisdell, New York, 1964.
8. Bellman, R., and Dreyfus, S. *Applied Dynamic Programming*. Princeton U. Press, Princeton, N. J., 1962, pp. 229–231.
9. Biondi, E., Debenedetti, A., and Luccio F. Cammini ottimi. In *Identificazione e ottimizzazione: Problemi e metodi, Collana di automatica, Vol. 2*, Consiglio Nazionale delle Ricerche, Roma, 1967.
10. Luccio, F. On some iterative methods for the determination of optimal paths through a network. *Calcolo 3*, Fasc. 1 (1966), 31–48.
11. Ledley, R. S. High-speed automatic analysis of biomedical pictures. *Science 143* (1964), 219–233.

RECEIVED DECEMBER, 1967; REVISED MAY, 1968