Thinning Algorithms on Rectangular, Hexagonal, and Triangular Arrays

E.S. Deutsch
University of Maryland

In this report three thinning algorithms are developed: one each for use with rectangular, hexagonal, and triangular arrays. The approach to the development of each algorithm is the same. Pictorial results produced by each of the algorithms are presented and the relative performances of the algorithms are compared. It is found that the algorithm operating with the triangular array is the most sensitive to image irregularities and noise, yet it will yield a thinned image with an overall reduced number of points. It is concluded that the algorithm operating in conjunction with the hexagonal array has features which strike a balance between those of the other two arrays.

Key Words and Phrases: thinning algorithms, rectangular, hexagonal, triangular arrays, image processing, skeleton

CR Categories: 3.64

Author's address: University of Maryland, Computer Science Center, College Park, MD 20742.

Introduction

In much of the literature concerned with thinning or skeletonizing operations on digital images, the most common type of array used is the rectangular one. All of the thinning operations proposed, however diverse, make use of a small rectangular subarea, centered on each point in the picture, within the confines of which the skeletonizing operations are performed. This is not surprising, for conventional scanning techniques, operating in a line-by-line fashion, at once suggest this type of array and the subsequent form of processing.

This paper proposes to examine additional types of arrays: not only rectangular, but also hexagonal and triangular. (See Figure 1.) These arrays correspond to the two-dimensional mosaics. The fact that none of these arrays, with the exception of the first, follows a strict row-and-column arrangement is of no real consequence, for given a fixed rectangular array, the others can always be derived from it.

In comparing algorithms associated with each of the three types of arrays, we use the same basic approach to each algorithm, i.e. a thinning algorithm developed for rectangular arrays. This thinning algorithm was first proposed by Rutovitz [1] and subsequently modified by the author [2]. An algorithm similar to that in [1] appears in [3].

The use of a generalized approach, rather than the development of a different type of algorithm for each specific type of array, is deliberate; it's advantage is that the relative merits of each type of array can be assessed in an easier manner. Otherwise, factors pertaining to the individuality of each approach would have to be taken into account. Suppose, for example, we found that an adaptive approach to thinning on a rectangular array was best. Then it would be unfair to compare the results with those obtained using, say, an averaging technique

Fig. 1.

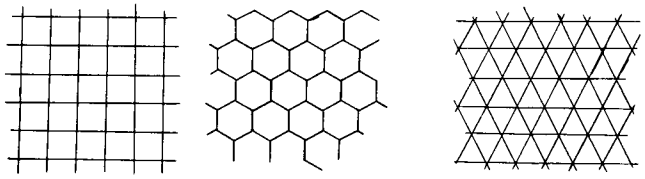


Fig. 2.

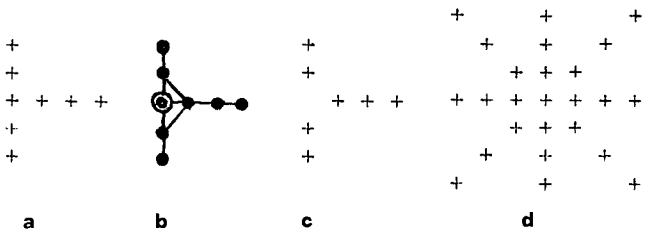
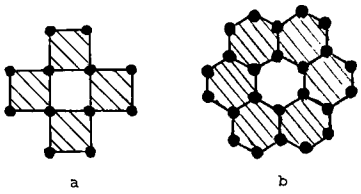


Fig. 3.



on an hexagonal array. The approach adopted here is based on notions of connectivity.

Of interest will be the relative "efficiency" of each type of array. Clearly, as the structure becomes more involved, so might its corresponding thinning algorithm. However, it will be useful to determine what advantages, if any, a particular array has—e.g. the resulting image reduction and processing time in each case.

The main motivation for thinning, especially in character recognition, is to obtain a line drawing of an originally-thick image: the fewer pattern points to be considered in the recognition stage the better. Skeletonizing an image implies a data reduction operation, which, if no portions of the image are lost—that is, if no discontinuity arises in the process—can make the recognition task easier. Once a line drawing representation of an image is available, its contours can be coded, and this code in turn can be recognized.

A partial aim of thinning is to reduce a connected image to a subset of points, the majority of which can be treated as nodes of order 2. This yields a line drawing. However, this aim can conflict with the designer's goal and lead to an erroneous recognition. As an example, consider the pattern on the rectangular array shown in Figure 2(a). The pattern's connectivity graph is shown in Figure 2(b). For connected image reduction, the point encircled could be removed, yielding the pattern shown in Figure 2(c). Yet for character recognition, such an action may present difficulties, for instance in distinguishing between the numeral 8 and the letter B. A discriminating feature lies in the mid-left portion of the B and the central portion of the 8. This requirement therefore overrides the connectivity reduction aim and is reflected in the algorithms presented. Coding algorithms for the handling of such and similar situations have been developed in which the apparent connectivity anomaly presents no difficulties [5]. Similar arguments apply to the pattern shown in Figure 2(d). Clearly the pattern represents a set of lines at 45° to each other, regardless of connectivity interpretations.

This paper does not attempt to represent a unified theory of thinning. While such a theory is clearly desirable, our aim here is to show how a thinning algorithm can be developed for each type of array of retinal points. Each of the three arrays are dealt with in turn, and in so doing, the connectivity criteria of the pattern (black) vis-à-vis that of its background (white) are discussed first. The rules of the thinning algorithm are then developed. Proof of continuity preservation for the first two algorithms can be found in [4].

The thinning algorithms appropriate to each array clearly show the increasing difficulties one encounters as the connectivity of the arrays becomes more complex. Combinatorially the hexagonal array is the easiest to deal with, whereas the triangular array presents quite a problem in view of the multiplicity of connectivity possibilities. Accordingly, the development of thinning rules for this array relies greatly upon those developed for the other two arrays.

Finally, it is pointed out that all the operations to be described are parallel; that is, we assume the operations on each element in the array take place simultaneously.

Rectangular Arrays

Connectivity Criteria

The skeletonizing algorithm described below is partially described in [1] and [2]. It is presented here in full. In dealing with patterns on rectangular arrays it has been pointed out in [5] that there is a choice between two connectivity situations. The pattern may be either 4-neighbor connected—using axial neighbors only—or 8-neighbor connected—using any of the eight peripheral neighbors. The choice is left to the user. Correspondingly, the complement or background must, mathematically, be either 8-neighbor or 4-neighbor connected.

Verification of this statement is presented below and follows the argument in [5]. We shall see that not all tesselations offer a connectivity choice. The absence of such a choice renders the design of a thinning algorithm easier since fewer connectivity situations have to be accounted for.

Consider the pattern shown in Figure 3(a). Assuming 4-neighbor connectivity for both the pattern and the complement, the number of vertices V in the pattern is 16,[1] the number of edges E is 16, and the number of faces F is 4.

Application of the Euler formula $V - E + F$ to the pattern should give its genus. Thus, by the above formula the genus is $16 - 16 + 4 = 4$. However, the pattern has four components and the background has two, so that the genus (the number of pattern components minus the number of background components + 1) is $4 - 2 + 1 = 3$. A similar disagreement in the value of the genus arises when 8-neighbor connectivity is assumed. For then the number of vertices in Figure 2 is 12, the number of edges 16, and the number of faces 4. Thus, by Euler's formula, the genus is 0, whereas in fact it should be 1.

However, if 4-neighbor connectivity is assumed for the pattern and 8-neighbor connectivity for the background then, by Euler's formula, the genus is 4. Since the number of background components is now 1 (not 2), the value of the genus obtained by counting the number of components is also 4. Similarly, when we assume the pattern to be 8-neighbor connected and the background to be 4-neighbor connected, both methods of calculating the value of the genus, which is 0, agree.

The Thinning Algorithm

For any element $a_{i,j}$ in row i and column j of the matrix, let $\gamma(1) \cdots \gamma(8)$ be the value of its eight neighbors 0 or 1 starting from $a_{i,j+1}$, in counterclockwise order, as shown. Let the patterns be 8-neighbor connected. The crossing number, χ, is defined as

$$\chi = \sum_{k=1}^{8} | \gamma(k+1) - \gamma(k) |,$$

4	3	2
5		1
6	7	8

where k has a period of 8 and χ indicates the number of distinct 4-neighbor connected groups of black and white (pattern and background) elements around $a_{i,j}$. In order for an element to be deleted from the pattern, all of the following conditions must hold:

1. $\chi = 0$, 2 or 4
2. $\sum_{k=1}^{8} \gamma(k) \neq 1$ (i.e. the element must have no, or at least two, neighbors in the pattern)
3. $\gamma(1) \wedge \gamma(3) \wedge \gamma(5) = 0$
4. $\gamma(1) \wedge \gamma(3) \wedge \gamma(7) = 0$
5. If $\chi = 4$ then in addition either condition (a) or (b) must hold:

a. $\{\gamma(1) \wedge \gamma(7) = 1\}$
&
$\{\gamma(2) \vee \gamma(6) = 1\}$
&
$\{\gamma(3)\&\gamma(4)\&\gamma(4)\&\gamma(5)\&\gamma(8) = 0\}$
Where & stands for the word *and*.

b. $\{\gamma(1) \wedge \gamma(3) = 1\}$
&
$\{\gamma(4) \vee \gamma(8) = 1\}$
&
$\{\gamma(2)\&\gamma(5)\gamma(6)\&\gamma(7) = 0\}$

The deletion operations continue until no further change occurs.

Briefly, the function of rule 2 is twofold. When the sum of neighbors is zero, the point in question is an isolated one and can be safely deleted. When the sum has the value 1, it prevents the ends of already thinned components from vanishing. Rules 3 and 4 preserve the 4-way connectivity in the top and right-hand positions within the rectangular window. Unless the pattern component is diagonal, erasure can only take place if there is only one peripheral pattern component ($\chi = 2$, rule 1). A special case arises when the pattern is a diagonal line, in which case $\chi = 4$. Thinning also takes place if the diagonal line is two elements thick (rule 5).

Rules 1 through 4 apply equally to a 4-way connected pattern; rule 5, however, applies only if 8-neighbor connectivity is used. The expression for χ in rule 1 can still be used.

A further rule can be applied to each element in the image after the last pass, provided the elimination of rectangular curves is not considered harmful: if

$$\gamma(k) \wedge \gamma(k+2) = 1 \text{ for } k = 1, 3, 5, 7,$$

then $a_{i,j}$ can be deleted.

It should be noted that the crossing number can be redefined in a simpler way:

$$\chi = \sum_{k=1,3,5,7} | \gamma(k+2) - \gamma(k) |.$$

However, the value of χ now gives the number of 8-way connected components surrounding $a_{i,j}$. The previous definition of χ yielded the number of such four-way connected components. The latter expression for χ is not used here.

The algorithm in its present form has a contraction bias in one direction; that is, the skeleton will tend not to lie centrally along the original pattern's components. In order to render it "isotropic"[2] the following rules should be used:

6. $\gamma(3) \wedge \gamma(5) \wedge \gamma(7) = 0$

[1] Since the image is 4-connected, the four faces are disconnected. In effect the nodes of adjoining faces are counted twice.

[2] For patterns which are only moderately thick—two or three picture units across—a nonisotropic algorithm may suffice, since the directional bias of the resulting skeleton will not be pronounced. When the pattern has thicker portions, an isotropic algorithm is required if the skeleton is to be centrally oriented.

7. $\gamma(5)\wedge\gamma(7)\wedge\gamma(1) = 0$
8. *if* $\chi = 4$, then either condition (a) or (b) must apply:

a. $\{\gamma(5)\wedge\gamma(3) = 1\}$
&
$\{\gamma(6)\vee\gamma(2) = 1\}$
&
$\{\gamma(1)\&\gamma(4)\&\gamma(7)\&\gamma(8) = 0\}$

b. $\{\gamma(7)\wedge\gamma(5) = 1\}$
&
$\{\gamma(8)\vee\gamma(4) = 1\}$
&
$\{\gamma(5)\&\gamma(6)\&\gamma(7)\&\gamma(2) = 0\}$

Note that the last set of rules consists of rules 3, 4, and 5 "rotated" through 180°. After the first pass using rules 1 through 5, rules 1, 2, 6, 7 and 8 are applied on a second pass, thus completing one cycle. See [4] for proof of connectivity maintenance.

Hexagonal Arrays

Connectivity Criteria

The first thing to note concerning hexagonal arrays is that there is no choice of connectivity for either the pattern or the background; both must be six-neighbor connected. This is verified by using the Euler formula on the pattern shown in Figure 3(b).

Assuming 6 neighbor connectivity, the number of vertices V is 24, the number of edges E is 30 and the number of faces F is 6. Using the Euler formula, the genus is equal to 0. Since the number of components of the pattern is 1 and the number of components of the background is 2, the genus has the value $1 - 2 + 1 = 0$. Thus, both values of the genus agree.

The Thinning Algorithm

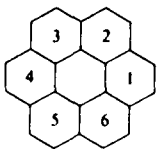
In developing an algorithm for hexagonal arrays, rules 1 and 2 pertaining to the rectangular array apply here too. Specifically, application of rule 2 is required to prevent already thinned lines from disappearing.

The crossing number can take the values 0, 2, 4, and 6. However, unlike the rectangular case, as soon as its value exceeds 2 the central element cannot be deleted; for in such cases the peripheral pattern (black) elements form two separate components which will be disconnected unless the center element is retained.

We thus have the first two rules:

$$\sum_{k=1}^{6} \gamma(k) \neq 1 \quad \text{and} \quad \chi = 2$$

both of which must apply if the element is to be deleted. Let the six-neighbor arrangement be as shown.



Here too we require a set of rules similar to rules 3 and 4 above. In Figure 3(b) let the neighboring element immediately to the right of the center element be labeled $\gamma(1)$, and the remaining ones be labeled counterclockwise, $\gamma(2)$ through $\gamma(6)$. At first sight one is led to stipulate that the following three connectivity conditions must be obeyed if the element is to be deleted:

a. $\gamma(1)\wedge\gamma(2)\wedge\gamma(3) = 0$
b. $\gamma(1)\wedge\gamma(2)\wedge\gamma(6) = 0$
c. $\gamma(1)\wedge\gamma(5)\wedge\gamma(6) = 0$

These rules seem reasonable since they guarantee connectivity between any set of three consecutive peripheral elements. For example, if rule (a) is not obeyed, the center element is not deleted, and the connectivity between $\gamma(1)$ and $\gamma(3)$ is maintained via the center element. Any set of three consecutive elements not explicitly written down forms the mirror image of one of the sets in (a), (b), or (c), and is taken care of when an earlier or a later element is considered.

However consider the simple pattern shown in Figure 4. In each pair of elements marked 1 and 2, one of the elements must be retained if the pattern is not to break up. By rule (c) the elements marked 1 are retained, while those labeled 2 are removed. Now consider the effect rules (a), (b), and (c) would have on the line pattern shown in Figure 5(a). The initial thinning stages are shown in Figures 5(b) and (c). Evidently rules (a) and (c) are incomparable: they cannot operate together in their present form, so one of them must be changed.

Taking the mirror image of rule (c) in order to preserve connectivity of any three consecutive elements, i.e. the rule:

$\gamma(2)\wedge\gamma(3)\wedge\gamma(4) = 0$

overcomes the difficulty encountered with the pattern in Figure 5(a); but if this rule is to be used then the pattern shown in Figure 5(d) will eventually vanish.

However, the connectivity between $\gamma(1)$ and $\gamma(5)$ (and between $\gamma(2)$ and $\gamma(4)$ when $\gamma(3)$ is the center element) will be maintained if the following two rules are substituted for rule (c):

$\gamma(1) = 1$
$\chi_{\gamma(1)} \neq 2$

That is, the crossing number at neighbor $\gamma(1)$ (the latter must also belong to the pattern) must not be 2. For then if $\gamma(5)$ belongs to the pattern, connectivity between $\gamma(1)$ is maintained via the center element.

The rules constituting the hexagonal thinning algorithm are summarized below:

1. $\sum_{k=1}^{6} \gamma(k) \neq 1$
2. $\chi = 2$
3. $\gamma(1)\wedge\gamma(2)\wedge\gamma(3) = 0$
4. $\gamma(1)\wedge\gamma(2)\wedge\gamma(6) = 0$
5. $\gamma(1) = 1$ and $\chi_{\gamma(1)} \neq 2$

Once again, in order to render the algorithm "isotropic," the additional rules given below can be used:

6. $\gamma(4)\wedge\gamma(5)\wedge\gamma(6) = 0$
7. $\gamma(3)\wedge\gamma(4)\wedge\gamma(5) = 0$

Fig. 4.

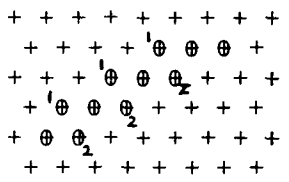


Fig. 5.

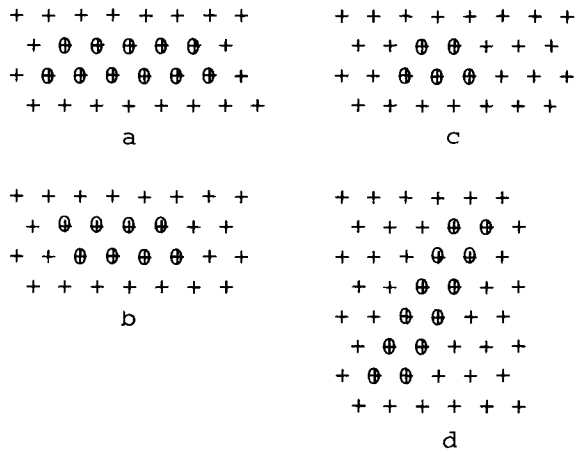
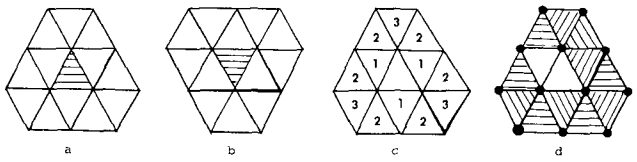


Fig. 6.



8. $\gamma(4) = 1$ and $\chi_{\gamma(4)} \neq 2$

Proof of continuity of this algorithm is given in [4]. A shrinking algorithm is also described in [7], but connectivity is not maintained in the process.

Triangular Arrays

Connectivity Criteria

The structure of the triangular array is much more complicated than either structure discussed above. The first thing to notice about this array is that the triangular elements have alternating orientations, so the arrangement of peripheral elements will vary accordingly. (See Figure 6.)

Here too it will be necessary to establish the neighborhood connectivity of both the pattern and its background. Before doing so we first show in Figures 6(a) and 6(b) the two different nearest-neighbor arrangements for the triangular array. For ease of reference, the central element in Figure 6(a) will be referred to as $\triangle$, and that of Figure 6(b) as $\triangledown$.

It will be observed from the figure that the nearest neighbors of both $\triangle$ and $\triangledown$ can be divided into three sets such that all the elements within each set are equidistant from the central element, distances being measured from centroids. The first set contains the triangular neighbors whose centroids are at a distance of 2 units away from the central element's centroid (where the median of each triangle is of length 3 units). The second set contains the six elements each centroid of which is at a distance of $2\sqrt{3}$ units away, and the third contains the three elements which are furthermost away, at a distance of 4 units. Note that included in this count are all the elements which share either an edge or a vertex with either $\triangle$ or $\triangledown$. This suggests a choice of neighborhood connectivity; 3, 9, or 12. See Figure 6(c) where the elements around $\triangle$ have been numbered according to the sets to which they belong.

It will be found that the Euler equation quoted above is satisfied if the pattern is 12-neighbor connected. As a result the background must be 3-neighbor connected. As a verification consider the configuration of elements in Figure 6(d), where the shaded elements constitute the pattern. The number of vertices V—if 12-neighbor connectivity is assumed—is 12; the number of edges E is 23; and the number of faces F in the pattern is 11. Accordingly, the genus is given by:

$$12 - 23 + 11 = 0.$$

The genus is in fact 0 because the numbers of components and holes in the configuration are both 1.

For the pattern to be 12-way connected, the background must be 3-neighbor connected, because now the only way the chain of triangles surrounding $\triangle$ can be broken, to connect the "hole" in the center with the "outside," is for two neighboring elements with a common edge to belong to the background too—that is, the background must be 3-way connected.

By the same token, if the pattern in Figure 6(d) is 3-way connected, then the number of vertices it contains is 13 (there are in effect two vertices at A), the number of edges is 23, and the number of faces is 11. Thus the genus, by Euler's formula, is:

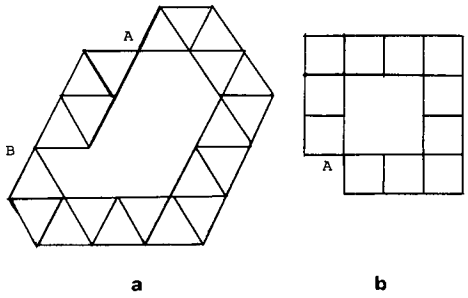
$$13 - 23 + 11 = 1.$$

Once again this corresponds to the true value of the genus; there are now only two components, pattern and background, whereas before there were three, one of the pattern and two of the background. With 3-way connectivity in Figure 6(d) there is in fact a gap at A. The same argument would apply had the pattern in Figure 6(d) contained all but the center element and any other of the 12 neighbors.

As a further example, consider the arrangement shown in Figure 7(a). If 12-neighbor connectivity is assumed, then there will be no gap at A; the connection

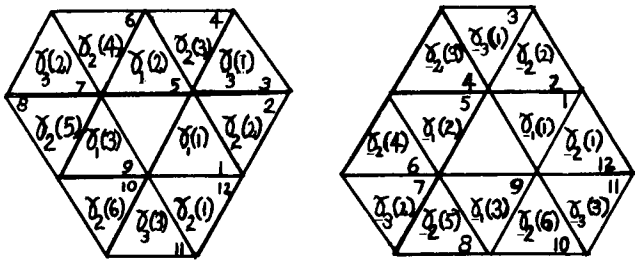
is maintained by an element belonging to the third set of neighbors. Similarly, there would not be a gap at B, where connectivity is maintained by an element belonging to the second set of neighbors. This result, in principle, is similar to the anomalies arising in rectangular arrays: in Figure 7(b) there would be a gap at A unless 8-neighbor connectivity is assumed.

Fig. 7.



a b

Fig. 8.



a b

The Thinning Algorithm

Let the pattern be 12-neighbor connected, and let each peripheral element be denoted by $\gamma_h(k)$ where k is the number of the element belonging to the set h = 1, 2 or 3. At the same time it will be convenient to refer to any of the 12 neighboring elements simply as $\gamma(k)$.[3] In the latter representation k denotes the neighbor number, 1–12, in general. Thus $\gamma_2(3) \equiv \gamma(4)$; see Figures 8(a) and 8(b). The arrangement of a computer printout of a triangular array is shown in Figure 9(a), together with the 12 neighbors of $\triangle$ and $\triangledown$, corresponding to Figure 8. The x and y "distances" of neighbor $\gamma(1)$ through $\gamma(12)$ from $\triangle$ are given by the pairs:

(1, 1), (1, 2), (0, 3), (−1, 2), (−1, 1), (−2, 0)
(−2, −1), (−1, −2), (0, −1), (1, −2), (2, −1), (2, 0),

respectively.

For $\triangledown$ the corresponding "distances" are:

(1, −1), (2, 0), (2, 1), (1, 2), (0, 1), (−1, 2)
(−2, 1), (−2, 0), (−1, −1), (−1, −2), (0, −3),
(1, −2), respectively.

It will be observed that the $\gamma_1(k)$ elements are flanked on either side by two $\gamma_2(k)$ neighbors. In turn, each such pair of $\gamma_2(k)$ elements is separated by a $\gamma_3(k)$ type element. Similarly, each $\gamma_2(k)$ neighbor has as its immediate neighbor a $\gamma_1(k)$ and a $\gamma_3(k)$ neighbor, while every $\gamma_3(k)$ neighbor has two $\gamma_2(k)$ elements as its immediate neighbors. Depending upon the connectivity of the pattern, these elements can combine in various arrays, as the following example shows.

Consider the formation of a "hole" (white) on the array, and let the hole consist of one element. Then, if the pattern were 3-way connected, for either $\triangle$ or $\triangledown$ to be completely surrounded by pattern elements, all the 12-neighbors would have to belong to the pattern. On the other hand, for a 12-way connected pattern the central element can be completed surrounded by combinations of either $\gamma_1(k)$ elements or $\gamma_2(k)$ elements or both. The $\gamma_3(k)$ type neighbors can be bypassed. This fact will be used later on.

In view of its rather large size, there arise within the window itself (Figure 8) a number of closed subpatterns, the connectivity of which must be preserved One such case is shown in Figure 10(a) in which

$$\gamma_1(3) = 0 \text{ and } \gamma_2(5) = \gamma_2(6) = \triangle = 1.$$

[3] It will be clear from the text which particular value of the element is meant.

There are a total of six such cases, in each of which the central element together with the two $\gamma_2(n)$ neighbors flanking its $\gamma_1(n)$ neighbor have the value 1, while the $\gamma_1(n)$ element itself is 0. We thus have the first connectivity rule which must be satisfied before either $\triangle$ or $\triangledown$ can be deleted from the pattern:

$$\overline{\gamma_1(k)} \wedge \gamma_2(2k-1) \wedge \gamma_2(2k) = 0 \text{ for } k = 1, 2, \text{ or } 3.$$

The 2-neighbor rule used for the other arrays cannot be applied indiscriminately here. Figure 10(b) shows two patterns, one a rotated version of the other, both of which would vanish if this rule were to be applied. There are a total of 12 such cases, six for each of the 2-neighbor arrangements. Accordingly, we have a further rule which must be satisfied prior to the erasure of either $\triangle$ or $\triangledown$:

$$\sum_{k=1}^{12} \gamma(k) > 2,$$

or if

$$\sum_{k=1}^{12} \gamma(k) = 2, \text{then } \gamma_1(k) \wedge [\gamma_2(2k-1) \vee \gamma_2(2k)] = 0$$
for $k = 1, 2, \text{ or } 3$.

Fig. 9.

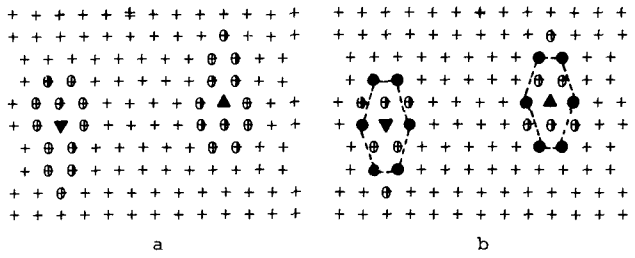


Fig. 10.

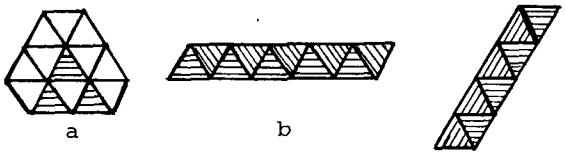


Fig. 11.

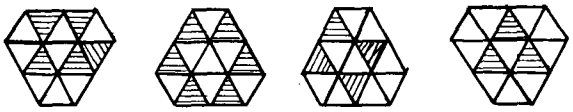
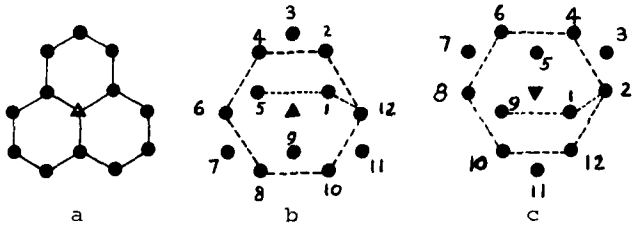


Fig. 12.



It was stated above that in tracing a 12-way connected path around $\triangle$ it is immaterial whether or not the $\gamma_3(n)$ elements are included in the path. Thus a further condition must be satisfied if the central element is to be deleted:

$$\prod_{k=1,2,3} \{\gamma_1(k) \vee [\gamma_2(2k-1) \wedge \gamma_2(2k)]\} = 0.$$

This test examines whether the central element is completely surrounded by a path in the pattern. If the above expression is nonzero, then the element cannot be erased since it combines with its nearest and next nearest set of neighbors to form one component of the pattern. Some examples are shown in Figure 11.

Two separate thinning algorithms were developed for this array, the difference between them being the method of defining the crossing number. In the first algorithm all the peripheral elements are used just as in the case of the rectangular arrays. In the second algorithm the crossing number was calculated to be the number of connected components around $\triangle$ or $\triangledown$.

Thus, for the first algorithm, we have the same rule as for the preceding cases, namely that $\chi \geq 2$, considering 3-neighbor connected elements.

From Figure 9(b) it will be seen that the $\gamma_2(n)$-type neighbors form a hexagon around $\triangle$ (or $\triangledown$). It was therefore thought convenient to apply, in principle, the rules developed for the hexagonal arrays. Accordingly two further sets of rules were added, both of which must be satisfied before the central element is removed:

1. If the central element is $\triangle$, we require
   $\gamma(2) \wedge \gamma(4) \wedge \gamma(12) = 0$
   $\gamma(2) \wedge \gamma(10) \wedge \gamma(12) = 0$
   $\chi_{\gamma_2}(2) \geq 2$
2. If the central element is $\triangledown$, we require
   $\gamma(2) \wedge \gamma(4) \wedge \gamma(6) = 0$
   $\gamma(2) \wedge \gamma(4) \wedge \gamma(12) = 0$
   $\chi_{\gamma_2}(2) \geq 2$

It was observed from the results that images are not reduced to single-line thickness. Briefly, this was due to the abnormally large window size, and to the method whereby the crossing number was defined. This algorithm was not developed any further because the one described below gave better thinning results. A final point in connection with this algorithm: an improvement may result if the rules containing the associated $\gamma_1(n)$ and $\gamma_3(n)$ elements are also included. Thus, for the deletion of $\triangle$ we would require the condition that:

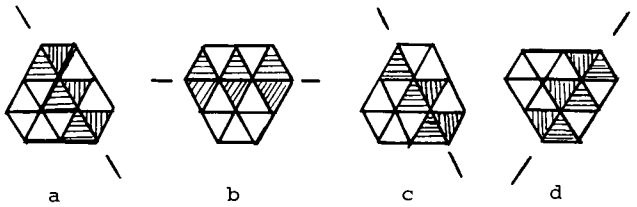
$\gamma(2) \wedge \gamma(4) \wedge \gamma(5) = 0$, or
$\gamma(3) \wedge \gamma(4) \wedge \gamma(5) = 0$.

Similarly for the deletion of $\triangledown$ we would require that:

$\gamma(2) \wedge \gamma(3) \wedge \gamma(12) = 0$, or
$\gamma(1) \wedge \gamma(2) \wedge \gamma(4) = 0$.

Let us inspect the triangular array a little more closely. The neighbors of $\triangle$ or $\triangledown$, together with $\triangle$ or $\triangledown$ respectively, can be thought of as forming three hexagonal lobes (see Figure 12(a)); in each lobe the central element provides the sixth vertex. One such lobe differs

Fig. 13.



from the ordinary hexagonal arrangement in that the former has no central element. (This is a method of generating triangular arrays; but in doing so, the density of points changes. In the experiments described here the density of points remained as high as possible for a given fixed basic rectangular array.) This being the case, and since each vertex of each lobe is connected directly to all the other vertices in that lobe (this also applies to the elements which form part of the neighboring lobes), we can define a new crossing number as the value of χ' where,

$$\chi' = \sum_{k=1,5,9} | \{\gamma(k) \vee [\gamma(k-1) \wedge \gamma(k+1)]\} - \{\gamma(k+4) \vee [\gamma(k+3) \wedge \gamma(k+5)]\} |.$$

The value of χ', just as in the modified crossing number definition for rectangular arrays, gives the number of 12-way connected components surrounding $\triangle$ or $\triangledown$. Thus in the modified algorithm we have the rule $\chi' \geq 2$ which replaces the old $\chi \geq 2$ rule: unless $\chi' \geq 2$, $\triangle$ or $\triangledown$ cannot be erased. Note that the rule concerning the number of neighbors remains as before.

It is now necessary to establish new connectivity conditions, for those used in the previous algorithm are inadequate. This will become apparent as the new rules are developed below.

The similarity of the formation of the 12 neighbors to a hexagonal arrangment is still pursued, but with the new definition of the crossing number there are two hexagonal arrangements to consider: First, that formed by the $\gamma_2(n)$ neighbors, as before; and second, that portion of a hexagonal array formed by two $\gamma_1(n)$ elements and one $\gamma_2(n)$ element (Figure 13(b) and (c)). The latter arrangement did not have to be considered in the previous algorithm because the crossing number of this combination, if it had existed separately, would not have been 2. The neighbors forming the additional *partial* hexagonal arrangement around $\triangle$ and $\triangledown$ respectively are:

$\gamma(1), \gamma(5), \gamma(12)$, and $\gamma(1), \gamma(9), \gamma(2)$.

It will have been observed that the 12-neighbor arrangement of $\triangledown$ is identical with that of $\triangle$ rotated through 180°. This being the case, we will first develop the rules for the neighbor of $\triangle$ and then apply them to those of $\triangledown$. Reference to the rules developed for the hexagonal arrays above indicates that there is one additional rule, applying to the neighbors of $\triangledown$ only, which involves the neighbors $\gamma(1)$, $\gamma(9)$, and $\gamma(2)$.

Consider once again Figure 12(b) and the first connectivity rule for the hexagonal array. The equivalent of this rule here would be the two rules:

$\gamma(2) \wedge \gamma(4) \wedge \gamma(12) = 0$, and

$\gamma(1) \wedge \gamma(5) \wedge \gamma(12) = 0$.

At the same time, however, the three element pairs $\{\gamma(4), \gamma(5)\}$; $\{\gamma(2), \gamma(1)\}$; and $\{\gamma(11), \gamma(12)\}$ are disposed around $\triangle$ so that they also form part of a hexagonal structure. Accordingly, we have the connectivity condition:

$$[\gamma(1) \vee \gamma(2)] \wedge [\gamma(4) \vee \gamma(5)] \wedge [\gamma(12) \vee \gamma(11)] = 0 \cdots T_1$$

which also incorporates the two rules stated above. The above rule incorporates the connectivity conditions of parts of the hexagonal arrays formed by the $\gamma_2(n)$ neighbors and the $\gamma_2(n)$ and $\gamma_1(n)$ neighbors.

Similarly the second connectivity rule, "borrowed" from the hexagonal array, requires that:

$\gamma(2) \wedge \gamma(10) \wedge \gamma(12) = 0$.

There is, however, a further element combination to the right of $\triangle$ which cannot be disturbed, namely that formed by $\gamma(2)$, and the pairs of $\gamma(1)$ and $\gamma(12)$ and $\gamma(10)$ and $\gamma(9)$. Accordingly we have the combined rule for the above two cases:

$$[\gamma(1) \vee \gamma(12)] \wedge [\gamma(9) \vee \gamma(10)] \wedge \gamma(2) = 0 \cdots T_2 .$$

Finally in the same vein we have the rule which will maintain connectivity between the $\gamma_1(n)$ neighbors only, $\gamma(1) \wedge \gamma(9) = 0 \cdots T_3$.

The connectivity situations discussed so far involved combinations of either $\gamma_1(n)$ or $\gamma_2(n)$ elements or both. There are, however, further combinations involving all three types of neighbors.

Figure 13 shows some examples of a pattern consisting of $\gamma_1(n)$, $\gamma_2(n)$ and $\gamma_3(n)$ type elements. It will be appriciated that such combinations can only be formed along the three principal lines of the triangular window. Each such line includes one of the sides of either $\triangle$ or $\triangledown$.

In Figure 13(a), we have the full combination of elements:

$\gamma_1(k), \gamma(k \pm 2), \gamma(k \pm 3) = 1$ for $k = 1$,

yet the partial combination of $\gamma_1(n)$, $\gamma_2(n)$ and $\gamma_3(n)$ elements shown in Figure 13(c) must also be considered.

In total there are six such principal combinations, three for $\triangle$ and three for $\triangledown$. However from Figure 13(a) it becomes clear that the combination shown is the

only one that has to be considered for $\triangle$. The remaining two combinations formed along the remaining two principal lines lie either to the left of or below $\triangle$ and need not be examined. We thus have the rule that unless $\gamma(1)$, together with any three of the elements $\gamma(k \pm 2)$ and $\gamma(k \pm 3)$, satisfies the following:

$\gamma_1(k) \wedge \gamma(k \pm 2) \wedge \gamma(k \pm 3) = 0$ for $k = 1 \cdots T_4$, $\triangle$ cannot be erased.

There will be no direct equivalent of rule T_1 for $\triangle$, in the case of ∇. However here the equivalent of the third rule for hexagonal arrays yields the rule:

$\gamma(2) \wedge \gamma(4) \wedge \gamma(6) = 0$

and by supplementing this rule so that it also includes the $\gamma_2(n)$ and $\gamma_3(n)$ neighbor we get the combined rule:

$$[\gamma(2) \vee \gamma(3)] \wedge [\gamma(4) \vee \gamma(5)] \wedge \gamma(6) = 0 \cdots T_5 .$$

The equivalent of rule T_2 is:

$$[\gamma(2) \vee \gamma(1)] \wedge [\gamma(4) \vee \gamma(5)] \wedge \gamma(12) = 0 \cdots T_6 .$$

A further rule, applicable to ∇ only, which is the equivalent of the fifth rule:

$\gamma(2) = 1$ and $\chi'_{\gamma(2)} = 2$.

The equivalent of rule T_3 will be:

$\gamma(1) \wedge \gamma(5) = 0$.

However, there is a further combination which has to be accounted for: namely,

$\gamma(2) \wedge \gamma(5) \wedge \gamma(9) = 0$.

Thus we have on combining the two:

$$[\gamma(1) \vee (\gamma(2) \wedge \gamma(9))] \wedge \gamma(5) = 0 \cdots T_7 .$$

Lastly, we have to consider the connectivity situations involving all three types of elements. Whereas there was only one such principal (along one principal line) case to be considered for $\triangle$, there are two principal cases here (Figures 13(b) and (d)). Accordingly, we have the two connectivity rules similar to rule T_4:

$\gamma_1(k) \wedge \gamma(k \pm 2) \wedge \gamma(k \pm 3) = 0$ for $k = 1$ and $2 \cdots T_8$.

To summarize, the thinning rules applicable to the triangular array are:

1. $\sum_{k=1}^{12} \gamma(k) > 2$, or if $\sum_{k=1}^{12} \gamma(k) = 2$, then $\gamma_1(k) \wedge [\gamma_2(k-1) \vee \gamma_2(k)] = 0$, $k = 1, 2,$ and 3.
2. $\tilde{\gamma}_1(k) \wedge \gamma_2(2k-1) \wedge \gamma_2(k) = 0$, $k = 1, 2,$ and 3.
3. $\chi' = 2$.

For $\triangle$

4a $\gamma(1) \wedge \gamma(9) = 0$
5a $[\gamma(4) \vee \gamma(5)] \wedge [\gamma(1) \vee \gamma(2)] \wedge [\gamma(11) \vee \gamma(12)] = 0$
6a $[\gamma(1) \vee \gamma(12] \wedge [\gamma(9) \vee \gamma(10)] \wedge \gamma(2) = 0$
7a $\gamma_1(k) \wedge \gamma(k \pm 2) \wedge \gamma(k \pm 3) = 0$, $k = 1$

For ∇

4b $[\gamma(1) \vee (\gamma(2) \wedge \gamma(9))] \wedge \gamma(5) = 0$
5b $[\gamma(4) \vee \gamma(5)] \wedge [\gamma(2) \vee \gamma(3)] \wedge \gamma(6) = 0$
6b $[\gamma(2) \vee \gamma(1)] \wedge [\gamma(4) \vee \gamma(5)] \wedge \gamma(12) = 0$
7b $\gamma_1(k) \wedge \gamma(k \pm 2) \wedge \gamma(k \pm 3) = 0$, $k = 1, 2$
8. $\gamma(2) = 1$, and $\chi'_{\gamma(2)} \neq 2$

Rules 1 through 3 remain unchanged for the isotropic algorithm; however, rules 4 through 8 are changed by symmetry as follows:

For $\triangle$

9a $[\gamma(5) \vee (\gamma(1) \wedge \gamma(6))] \wedge \gamma(9) = 0$
10a $[\gamma(6) \vee \gamma(7)] \wedge [\gamma(8) \vee \gamma(9)] \wedge \gamma(10) = 0$
11a $[\gamma(5) \vee \gamma(6)] \wedge [\gamma(8) \vee \gamma(9)] \wedge \gamma(4) = 0$
12a $\gamma_1(k) \wedge \gamma(k \pm 2) \wedge \gamma(k \pm 3) = 0$, $k = 2, 3$
13. $\gamma(6) = 1$ and $\chi'_{\gamma(6)} \neq 2$

For ∇

9b $\gamma(5) \wedge \gamma(9) = 0$
10b $[\gamma(1) \vee \gamma(12)] \wedge [\gamma(9) \vee \gamma(10)] \wedge [\gamma(8) \vee \gamma(7)] = 0$
11b $[\gamma(5) \vee \gamma(6)] \wedge [\gamma(8) \vee \gamma(9)] \wedge \gamma(10) = 0$
12b $\gamma_1(k) \wedge \gamma(k \pm 2) \wedge \gamma(k \pm 3) = 0$, $k = 2$

Results and Discussion

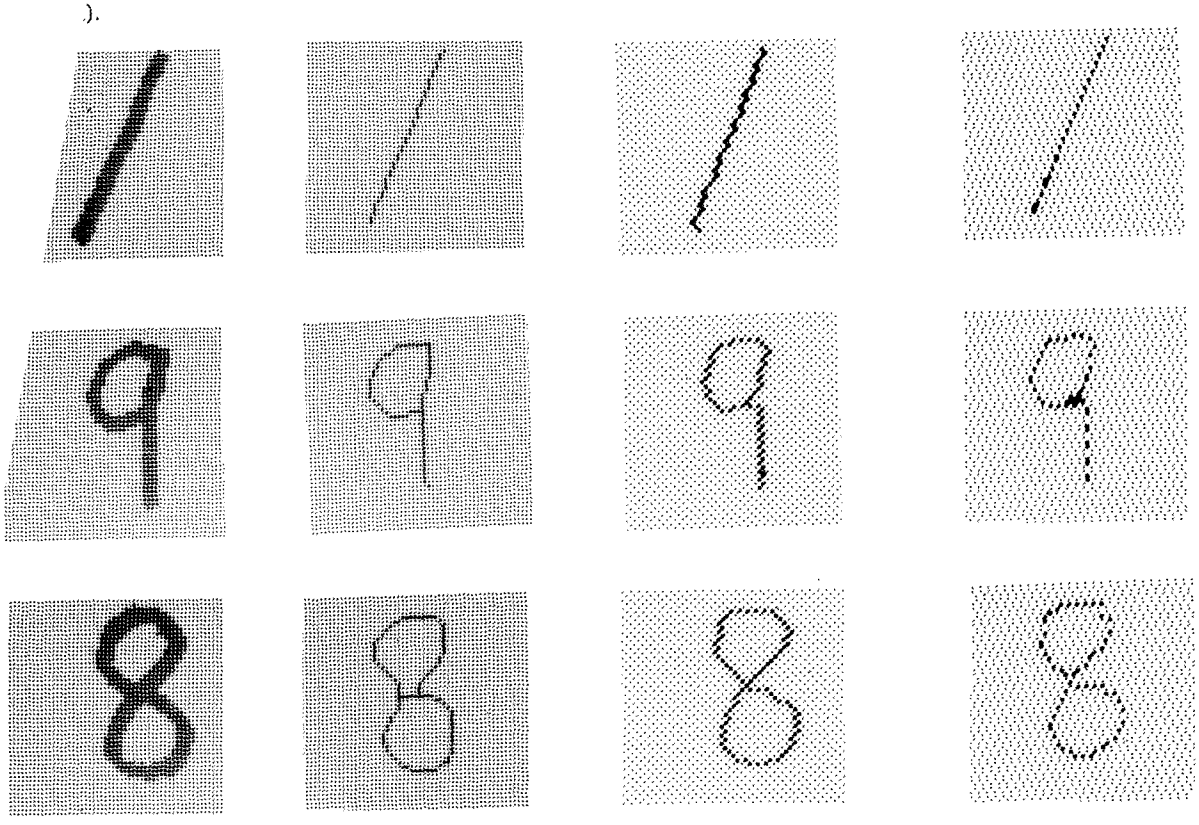
A field in which thinning algorithms have found a wide application is character recognition, in which, prior to the encoding of a character's shape, it is necessary to reduce the original image to a line drawing. A good testing for the performances of the algorithms developed above would be their application to alphanumeric characters. The following would be of interest: given a fixed rectangular array of points can these points, interconnected to form a different array structure and in conjunction with an appropriate thinning algorithm, yield a more useful processor? Clearly the ultimate usefulness of any preprocessor will depend upon its performance within an entire image-recognition system of which it forms only a part. However, given that the desired result is a simple line drawing of the original image consisting of a minimal number of points, it is possible to compare such algorithms.

The algorithms developed in this paper are all parallel in operation, and similar operations in each algorithm use similar instructions. A useful way of comparing their "cost" would be to compare the storage and processing time of each. Another factor worth comparing is the data reduction resulting from each algorithm, bearing in mind that the densities of points in the array differ.

Figure 14(a) and (b) show some images operated upon by the thinning algorithms, and in Figure 15 the various performance parameters pertaining to each algorithm are summarized.

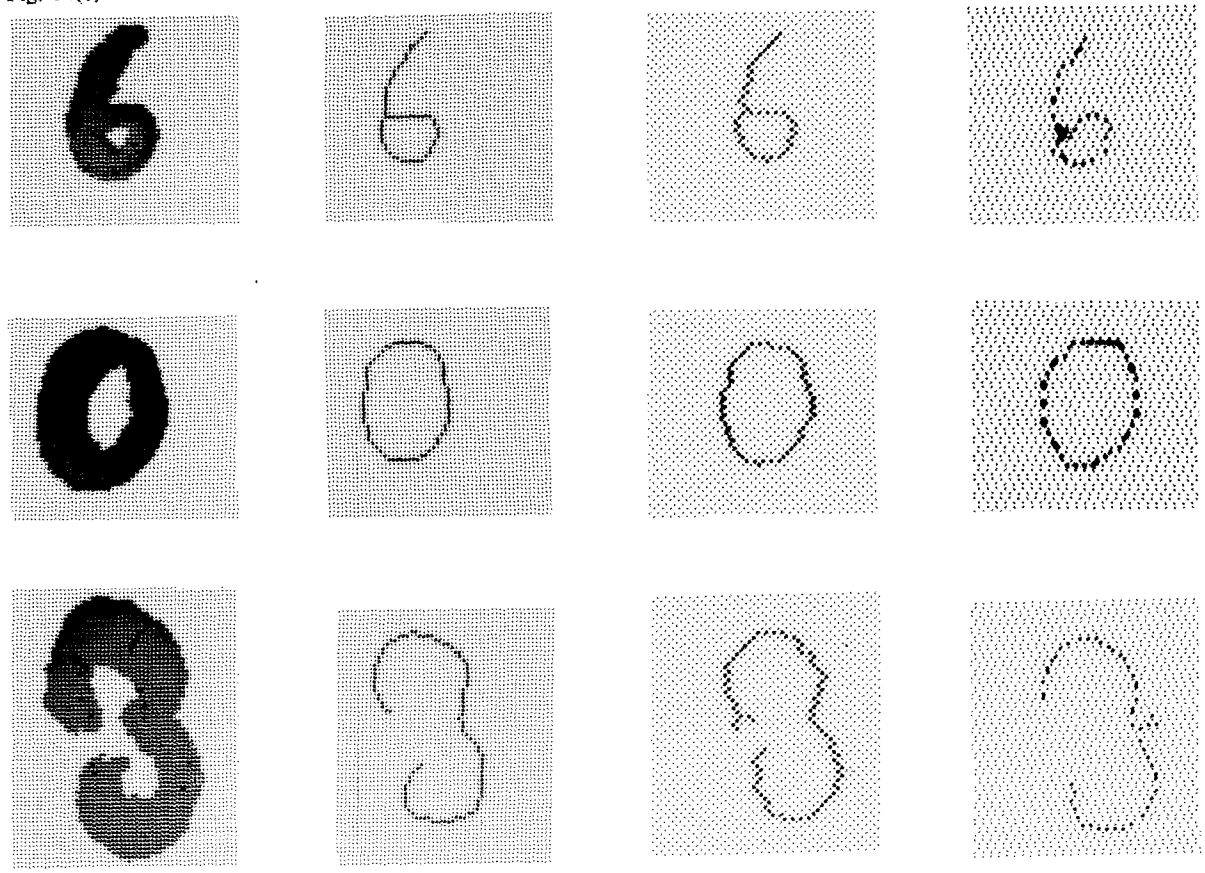
It will be seen from Figure 14 that of all the resulting thinned images those obtained using the triangular array contain the least number of points per image. This result is not unexpected, since on this array the neighbors span the largest distance. The ratio of the maximum distances of any neighbor on the rectangular, hexagonal, and triangular arrays is $1 : \sqrt{2} : 3/\sqrt{2}$. However, the increased size of the basic window renders the processing on a triangular array—and thus the re-

).



Original Rectangular array Hexagonal array Triangular array

Fig. 14(b).



Rectangular array Hexagonal array Triangular array

Fig. 15.

	Rectangular	Hexagonal	Triangular
Image Processing Time	1	0.5	1.8
Number of Core Locations	1	1.0	1.1
Average Image Reduction	1	1.7	0.8

sulting image—very sensitive to edge irregularities and, more important, to noise. The former effect is clearly demonstrated in the last image of Figure 14(b). From this point of view the hexagonal array is preferential, since all its neighbors—theoretically at least—are equidistant. The hexagonal array has the additional attraction that the processing time required is considerably less.

Both the triangular and hexagonal arrays contain the same number of points—half the number of points contained within the original rectangular array. Yet despite the fact that the image on the triangular array contains the minimal number of points, there is an additional argument against the use of triangular arrays to be considered. If the thinned image is to be chain encoded [8] the number of direction vectors, in the triangular array is 12, which means that the maximum number of bits required to represent a single direction vector is four; this compares with three bits required for the other two arrays. So for storage or transmission of a complete resulting line drawing, the triangular array will only be useful if the number of points of the image thereon is less than three-quarters the number of points in the image on the hexagonal array. The experiments show this is the case, yet all this is obtained at the expense of increased processing time.

The hexagonal grid also has advantages over the rectangular grid as far as processing time and data reduction are concerned. See [9] for a further discussion on the relative merits of these two types of arrays.

Finally, a word about the hardware implementation of the operations by means of cellular arrays. The rectangular and the hexagonal array algorithms can be easily implemented because each cell is connected to 8 or 6 neighboring cells. The triangular array algorithm presents some difficulties because each cell has to be connected to 12 neighboring cells; the connecting array is not planar and must be realized with a multilayer method.

Conclusion

Thinning algorithms for various types of arrays were developed and compared experimentally in this paper. It is concluded that triangular arrays, while being the most expensive in terms of processing time, will yield an image with a minimal number of points. The algorithm operating with this array is, however, very sensitive to noise and edge irregularities. The hexagonal array offers a balance between the rectangular and the triangular arrays in that it requires almost the same storage, yields a midway average reduction value, and has the shortest processing time.

Received December 1970; revised September 1971

References

1. Rutovitz, D. Pattern recognition. *J. Royal Statistical Society* {A} *129IV*, (1966), 504–530.
2. Deutsch, E.S. Comments on a line thinning scheme. *British Computer J. 12*, 4 (Nov. 1969), 142.
3. Hilditch, C.J. Linear skeletons from square cupboards. In *Machine Intelligence 4*, B. Meltzer and D. Michie (Eds), American Elsevier, New York, 1969, pp. 403–420.
4. Deutsch, E.S. Towards isotropic image reduction. IFIP Congress 71, North Holland Pub. Co., Amsterdam; Booklet TA-2, pp. 75–85.
5. Rosenfeld, A. Connectivity in digital pictures. *J ACM 17*, 1 (Jan. 1970), 146–160.
6. Deutsch, E.S. Preprocessing for character recognition. I.E.E. Conf. on Pattern Recognition, July 1968, pp. 179–190.
7. Golay, M.J.E. Hexagonal parallel pattern transformations. *Trans IEEE Comput. C18*, 8 (Aug. 1969).
8. Freeman, H. On the encoding of arbitrary geometric configuration. *IRE Trans E C* (June 1961), 260–268.
9. Deutsch, E.S. On parallel operations on hexagonal arrays. *Trans IEEE Comput. C19*, 10 (Oct. 1970), 982–983.