Optical recognition of hand-printed characters of any size, position, and orientation

by S. Di Zenzo
M. Del Buono
M. Meucci
A. Spirito

This paper deals with the optical recognition of text data in documents such as engineering drawings, land-use and land-register maps, and utility maps. The automatic computer acquisition of these documents is performed through the basic steps of vectorization of the line-structure and recognition of the text data interspersed in the document. The latter data are usually handwritten by professional draftsmen, and may have any size, position, and orientation. We review some of the features appropriate to this particular OCR problem, and suggest a special recognition strategy. Numerous examples are given. The results obtained with a prototype system on actual land-register maps are reported.

1. Introduction

Traditionally, the recognition of hand-printed characters has been considered of importance for applications in which the automatic reading of forms written by hand is needed. More recently, the recognition of hand-printed characters has gained importance as part of a larger application known as *intelligent forms processing* [1].

Another application that requires the recognition of hand-printed characters is the processing of text data in the automatic acquisition of engineering drawings and land-use maps. Much literature on this acquisition problem exists (surveys can be found in [2, 3]). However, only a few papers deal with the particular OCR problem associated with this application [4–6]. This particular OCR problem can be very difficult—indeed, the symbols can be of any size and orientation in the image frame; they are often isolated, offering no contextual information; symbols and lines may overlap; and the separation between symbols arranged into strings may be imperfect.

The recognition rate for hand-printed characters cannot be as good as, say, one for typeset data. This is particularly true when characters are rotated. The recognition rate depends on the number of writers and their training: If this number is high, recognition can be

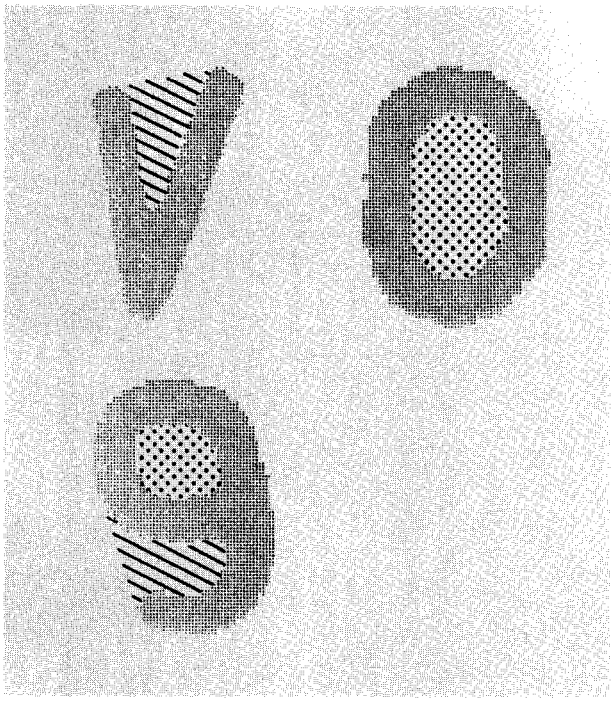

Figure 1

Bays (hatched areas) and lakes (dotted areas).

very difficult. If the original documents were produced by professional writers who were motivated to imitate some specific font or style, the recognition rate can be considerably higher [7].

In this paper, we illustrate character recognition techniques that rely upon the use of *features*. A feature is a property that can be measured on the objects to be recognized. For plane figures such as characters, typical examples of features are height, width, area, central moments, number of horizontal or vertical strokes, number of endpoints, and number of multiple points.

One of our concerns in this paper is to find features that are completely invariant under motions of the plane such as shifts, rotations, and contractions/dilations, and are resistant to reasonable distortions as well as to noise. We also suggest a strategy of recognition appropriate for these features.

Some of our features are based on lakes (holes), bays (concavities), and sides. **Figure 1** illustrates lakes and bays; sides are introduced later. The use of such features as lakes and bays for OCR was first suggested by Unger in 1956 [8]. Munson [9] and Freeman [2] suggested extensions of these ideas. A systematic account can be found in Duda and Hart [10].

We emphasize that no great claims of originality are intended for most of the concepts and methods that we discuss. However, we feel that the overall viewpoint of this paper is novel. The amount of literature on OCR with rotated characters is quite limited, and it seems desirable to present a self-contained treatment of a possible approach to this problem.

There is a considerable body of literature on hand-printed character recognition. Among the classical papers, we quote [11–13]. Recent surveys on the subject include [14, 15]. An interesting recent research paper is [16]. A recent survey of the general problem of character recognition is found in [17]. A survey paper by one of the main contributors to OCR is [7]. In the past few years, handwritten-character recognition using dynamic information has received much attention. Surveys of the intense research activity in this field can be found in [18, 19].

We conclude this introductory section with a remark on serifs. It is well known that in OCR a large number of misclassifications originate from imperfect separation between adjacent characters. With serif fonts, the fraction of touching characters, hence the number of misclassifications, tends to increase because serifs bridge adjacent characters. In this paper, we ignore serifs, and assume that writers are motivated to avoid serifs in hand-printing characters. This greatly simplifies our exposition. However, our methodology still applies if serifs are present: What changes is the number of different shapes that may correspond to any single character (see the subsection on decomposition of symbols into shapes).

2. Overview of the application

The features and the strategy of recognition discussed in this paper have been implemented within the character recognition subsystem of a larger system designed for the raster-to-vector conversion of engineering drawings and land-use maps. The latter system has been implemented in the framework of an independent technical effort, namely the automatic acquisition of the maps of the Italian Land Register Authority.

In this section we make some preliminary comments on the application, and, more specifically, on the nature of the original documents that are processed. More detailed treatment of this kind of application can be found in [20–22]. The overall map acquisition system that we developed is reported in [23].

A land-register map consists of a set of interconnected thin lines on a contrasting background. Text information is always interspersed within the line structure, and dotted/dashed lines are almost always present.

A portion of a land-register map is shown in **Figure 2**. Continuous lines define the boundaries of land properties and buildings; names identify streets; numbers identify units or parcels of land property; dashed lines and special cadastral signs (e.g., arrows) carry conventional information.

The standard image data that are input to the raster-to-vector conversion program are the two-level images produced by the raster digitization of the originals. An important processing step is the separation of the text data from lineal data (there can be spurious contacts between the two types of data). The image pieces classified as "lines" and those classified as "symbols" are then routed along different computation paths: The lines are vectorized, while the symbols are submitted to the recognition subsystem.

The symbols that can occur in a land-register map are usually listed in the standards of the pertinent administration. For the Italian Land Register Authority they are 1) ten numerical digits; 2) 52 alphabetic characters (lowercase and uppercase); and 3) 25 legal cadastral signs (though only twelve of these signs are actually encountered in maps).

A small fraction of the symbols are hand-printed by means of a lettering guide; the rest are handwritten by professional draftsmen. There may be a wide variety of fonts (each draftsman has his own calligraphic style). However, a fortunate circumstance has made recognition somewhat easier—serifs are almost always absent.

The recognition task is quite difficult in this particular instance of an OCR application. Indeed, the symbols may appear in any size and orientation and are usually scattered around in the picture in the most unpredictable way. Symbols may be completely isolated, or may be overlapped with fragments of lines.

Symbols arranged in strings often touch one another; also, they are often aligned incorrectly. Thus, because the baseline of a string can be evaluated with only limited accuracy, it is not advisable to use this information for recognition. (It can be used for only the broad distinction between "up" and "down"—for deciding between, say, 6 and 9, b and q, and d and p.)

It must be noted that the organization of nonisolated symbols into strings, though of limited use for the recognition of symbols, is, however, an important processing step. Indeed, we are eventually interested in recognizing names and numbers, and in assigning them to the appropriate geometric entities. We mention this here, since we do not cover this topic in this paper. A detailed treatment of string detection can be found in [20].

3. Shape features

In a digital image, an object, or figure, is represented by a set of black points on a white background. By computing certain features of this set, we can obtain a description of the object.

Examples of features include the area (number of black points), the diameter (greatest distance between any two black points), the ratio between height and width, and, say, the abscissa of the centroid of the object.

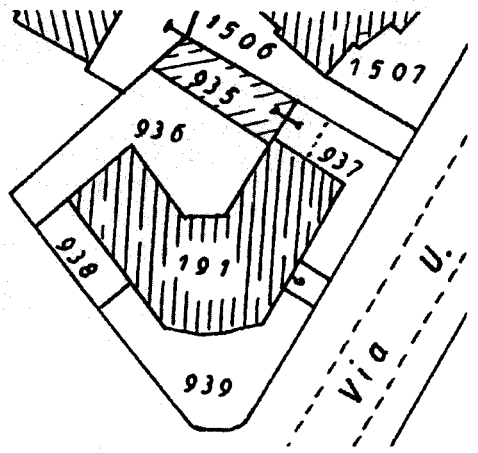

Figure 2

Portion of a land-register map.

Note that the area and diameter both depend on the object size, the ratio between height and width depends on the orientation of the object, and the abscissa of the centroid depends on the horizontal displacement of the object. Hence, these features are not appropriate for characterizing the *shape* of the object; objects of different sizes, positions, and orientations may well have the same shape.

However, if we divide the area by, say, the area of the convex hull of the object, and the diameter by, say, the perimeter of the convex hull, the two resulting features are completely independent of any possible translation, rotation, and variation in size of the object. In brief, they are invariant under the *similarity transformations* of the plane—motions of the plane within its own two dimensions that are combinations of translations, rotations, and scalings.

Since we are concerned with the recognition of hand-printed characters of any size, position, and orientation, we need features that do not change value under the similarity transformations of the plane. This is the first requirement to be set forth for the features that we intend to study.

Various features display these invariance properties; normalized moments and Hu invariants are very popular examples, as well as the Fourier descriptors of the object contour. Also, various interesting features with the required invariance properties can be computed in terms of both moments and the power spectrum of the object.

Other similarity-transformation invariants can be derived from *medial-axis* transformation (MAT), which maps the object into its skeleton. The computation of geometric properties from MAT is investigated in [24, 25].

The second requirement for the features to be used in the recognition of hand-printed characters is the following: They should be as insensitive as possible to noise and to certain transformations of the plane within its own two dimensions, such as stretching along one direction and rubber-sheet distortions. These more general motions of the plane are usually present in hand-printed characters.

Unfortunately, the higher-order terms of both the Hu invariants and the Fourier descriptors are very sensitive to noise, while the low-order ones alone provide a very incomplete and gross representation of a figure. Also, all of these features are strongly affected by stretching and rubber-sheet distortions.

Thus, we arrive at the problem of obtaining new features satisfying the two requirements stated above. We need features that are exactly invariant under similarity transformations, are rather insensitive to noise (in particular to quantization errors), and do not vary much under stretching and local distortions of the plane. Features that satisfy these requirements will be called *shape features.*

An important class of shape features comprises *topological features*, which do not change under the *topological* transformations of the plane (one-to-one continuous mappings whose inverse is also continuous). Topological features are completely invariant under similarity transformations, stretching, and any kind of rubber-sheet distortions. Unfortunately, there are only two independent topological features of a plane set: the number of connected components and the number of holes [10]. Since two features are too few, we require some additional features.

4. A set of shape features

Quite obviously, the number of lakes is an excellent shape feature. Two other obvious candidate shape features are the number of bays and the number of corners. A combined use of these three numbers might seem convenient, since these numbers are actually independent as features. Indeed, it is possible to create as many concavities as desired (in a continuous figure), while keeping both the number of holes and the number of corners fixed on certain given values. It is possible to create as many holes (corners) as desired, while keeping fixed both the number of corners (holes) and the number of concavities. Unfortunately, both the number of bays and the number of corners have serious limitations as features.

In this section we attempt to show why the number of holes is such a good feature, and why the other two numbers have limitations. Finally, we define certain new shape features which, according to our experiments, can be quite useful in the recognition of rotated characters.

- *Number of lakes*

As mentioned previously, the number of lakes is a useful recognition feature. Indeed, it provides a reasonably reliable discrimination among three clusters of symbols: no lakes, one lake, and two lakes (we are not interested in shapes with more than two lakes). The discrimination is reliable, since it seldom happens that lakes such as those in the characters A, B, ··· are created or destroyed by noise, at least if the sampling rate is adequate. Hence, one can rely upon a threshold separating noisy from actual lakes.

In discriminating between "good" and "bad" shape features, the following criteria are useful. A good shape feature

- Has discriminating capability.
- Is distributed normally within each of the subpopulations corresponding to symbols.
- Has small variance within each of these subpopulations.
- Is not strongly affected by the sampling rate.
- Is computationally cheap.

To meet the first requirement in this list, the distributions over the symbol subpopulations should be as spaced as far apart as possible, so that the global variance of the feature, as computed on the overall population, is high.

- *Bays and lids*

The number of concavities must be handled with some care in a recognition task; indeed, when using it as a feature, we need a threshold to separate noise concavities from true concavities. Strictly speaking, any threshold value would be arbitrary, and would be a source of errors.

With reference to our criteria for good features, we can be more specific. The number of bays should at least discriminate between convex and nonconvex objects. Since the discrimination of true vs. spurious bays is somewhat ambiguous, we may detect bays on a convex object while missing all the bays on a nonconvex one. Thus, the variance of this feature within the subpopulations it should discriminate from one another (shapes with no bay, one bay, . . .) is high. Besides, this feature is very sensitive to the sampling rate: If the sampling rate is lowered, the area of the spurious bays increases, and the performance of the feature decreases.

In practice, bays whose normalized area is greater than 0.03 very likely correspond to actual concavities in the figure (the normalized area of a bay is the area of the bay divided by the area of the convex hull of the overall figure). If we discard all bays with a normalized area of less than 0.03, we will very likely discard some true

concavities also. However, it is unlikely that we will discard *large* concavities. This remark perhaps suggests that large concavities are more reliable as safe recognition features than concavities. (In the prototype system described in Section 6 we use only large concavities, i.e., concavities whose normalized area is no less than 0.03.)

Though number of bays is somewhat unreliable as a feature, bays can be extremely useful in the recognition of rotated characters. There is one proviso: One should not try to use the bays themselves, but their *lids* instead. A lid of a plane figure is a maximal portion of the perimeter of the convex hull of the figure not belonging to the figure itself. The arrow in **Figure 3** represents a lid. Note that the bay associated with the lid in that figure is concave, which may produce the impression that there are two bays in the figure.

Lids are very simple entities; they are vectors, each equipped with a tail, a tip, and an orientation. We stipulate that the orientation is chosen such that each lid leaves the figure on the left (anticlockwise orientation).

Lids play an important role in this paper. We build certain new features using lids that convey information on shape. When there is more than one lid, the relative positions of the successive lids are sometimes sufficient information for recognition.

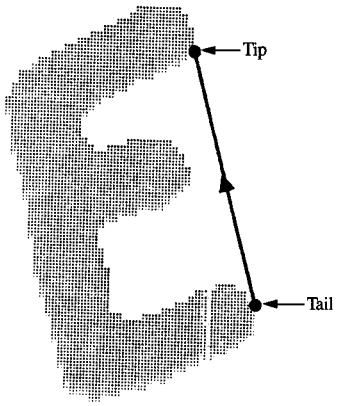

Figure 3

Illustration of a lid.

• *Corners and sides*

Obviously, corner detection has good possibilities as an aid for recognition. The difference between an acute and an obtuse angle is meaningful, and there is no reasonable deformation that will carry, say, a regular triangle into a square. It should then be reasonable to expect corner detection to be a powerful tool for the recognition of plane shapes.

However, in the presence of noise, corners actually become very subjective: When corners are smoothed or destroyed by noise, as in Figure 4(a), it may be difficult for even the human eye to evaluate their positions, or even their existence in a figure. **Figures 4(a)–(c)** suggest that certain almost imperceptible degradations of an object can transform a triangle into a blobby object without corners at all—it is impossible to establish the stage in this transformation at which the object ceases to be a triangle. This demonstrates that any threshold separating objects with and without corners would be somewhat arbitrary.

On the other hand, we maintain that almost the same information associated with corners is conveyed by the *sides* of a figure, when they exist. The absence or presence of sides in the convex hull of a figure is relatively easy to evaluate. If there are sides, their relative position conveys as much information on shape as the relative position of lids.

In this paper, we adopt the following, simplified, definition of the term *side*. First, we note that the boundary of the convex hull of a figure always consists of a sequence of segments (see **Figure 5**). We normalize the lengths of these segments by dividing them by their sum. Segments whose normalized length is greater than a fixed threshold, typically 0.15, are called *sides*. Examples of

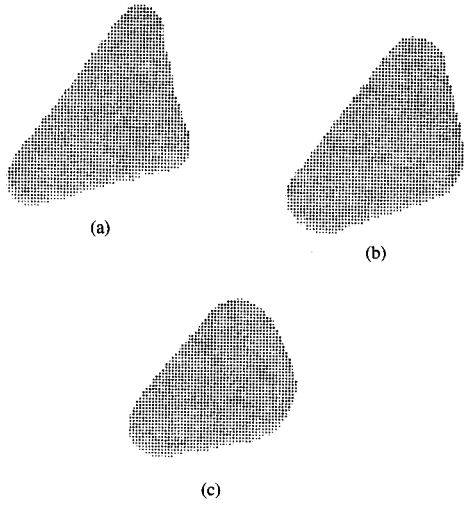

Figure 4

Triangles affected by noise.

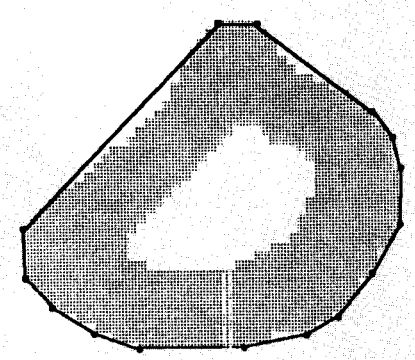

Figure 5

Perimeter of the convex hull.

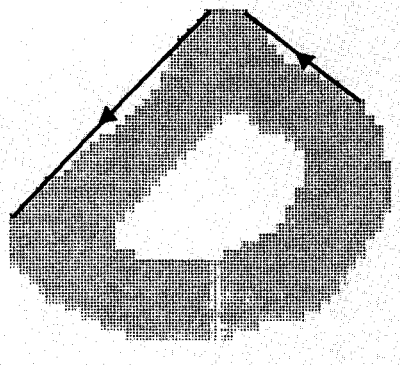

Figure 6

Illustration of sides.

sides are given in **Figure 6.** (It is not so straightforward, however, to actually implement the concept of sides in a system. Some heuristics are needed, at least for merging two consecutive sides that are almost parallel.)

Note that, since it is a vector, each side has a tail, a tip, and an orientation. For orientation, we stipulate the same convention as for lids.

There is experimental evidence that sides and lids are quite independent as features. A possible explanation of this fact is suggested by the following observations: Objects without bays have no lids but may have sides; on the other hand, in an object with bays a side may include one or more lids, and there can be short lids not included in any side.

This (relative) independence may justify the claim that lids and sides actually convey (relatively) independent information on shape. In the next subsection, we define various features in terms of lids and sides.

- *New shape features*

In the following, $d(A, B)$ denotes the normalized distance between points A and B. The normalization factor is the perimeter of the convex hull. $\overline{AB}.\overline{CD}$ denotes a scalar product. The following terminology applies to both lids and sides:

- Two lids (sides) $\overline{AB}$, $\overline{CD}$ are *near* to one another if $d(B, C) \leq t_1$ or $d(D, A) \leq t_1$, where t_1 is a suitable threshold.
- Two lids (sides) $\overline{AB}$, $\overline{CD}$ are *far* from one another if $d(B, C) \geq t_2$ and $d(D, A) \geq t_2$, where t_2 is a suitable threshold.

Of course, we must avoid the case in which a pair of lids or sides are simultaneously near to and far from one another. Thus, we must have $t_1 < t_2$. Typical values are $t_1 = 0.2$ and $t_2 = 0.25$. Let us complete our terminology:

- Two lids (sides) $\overline{AB}$, $\overline{CD}$ are *consecutive* if either there are no lids (sides) between B and C, or there are no lids (sides) between D and A.
- Two lids (sides) $\overline{AB}$, $\overline{CD}$ are *cooperating* or *competing* according to whether $\overline{AB}.\overline{CD} > 0$ or $\overline{AB}.\overline{CD} < 0$.
- Two lids (sides) that are far and competing form a *twisted pair* of lids (sides).
- A sequence of lids (sides), each (but the last) near and consecutive to the following, forms a *chain* of lids (sides).
- If the last lid (side) in a chain of lids (sides) is near and consecutive to the first, the chain forms a *cycle* of lids (sides).

We are now able to define the following six shape features:

- By the *lid-torsion* (*side-torsion*) of a shape, we mean its number of twisted pairs of lids (sides).
- The *lid-chain-length* (*side-chain-length*) of a shape is the number of lids (sides) in its chain of lids (sides) [0, if there is no chain of lids (sides)].
- The *lid-cycle-length* (*side-cycle-length*) of a shape is the number of lids (sides) in its cycle of lids (sides) [0, if there is no cycle of lids (sides)].

Figure 7 shows characters with different lid-torsion values. Lid-torsion meets all the criteria for features, with

the possible exception of the last (we discuss efficiency in the subsection on computing the shape features). Lid-torsion is a very stable property for numerous shapes. Thus, for example, we found exactly two 2s having a lid-torsion value different from 1 in a population of more than 70 000 handwritten characters. Over the same population, no X was found with a lid-torsion different from 2.

Figure 8 shows characters with different lid-chain-length and lid-cycle-length values. These features meet all of the criteria except the last, to a satisfactory extent; however, they are not as good as the lid-torsion. On a practical basis, this amounts to stating that we can achieve good results by using these features, but we cannot use them in a straightforward way: We must implement techniques that will provide reliable results from less reliable data.

The "side" versions of these features are less reliable than the corresponding "lid" versions. Their use must be supplemented with special recognition techniques that can produce results that are more reliable than the input data. We discuss this topic in Section 5.

We conclude this section by discussing four shape features that can be useful in the recognition of hand-printed characters. To our knowledge, the last three

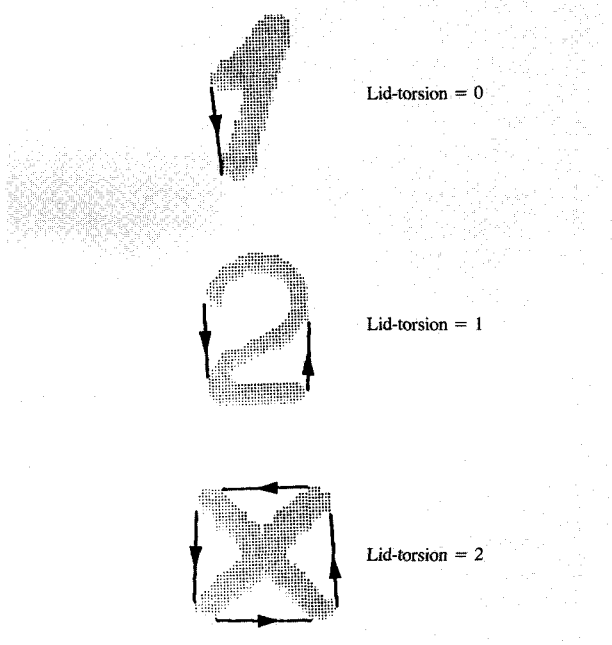

Figure 7

Lid-torsion values for different characters.

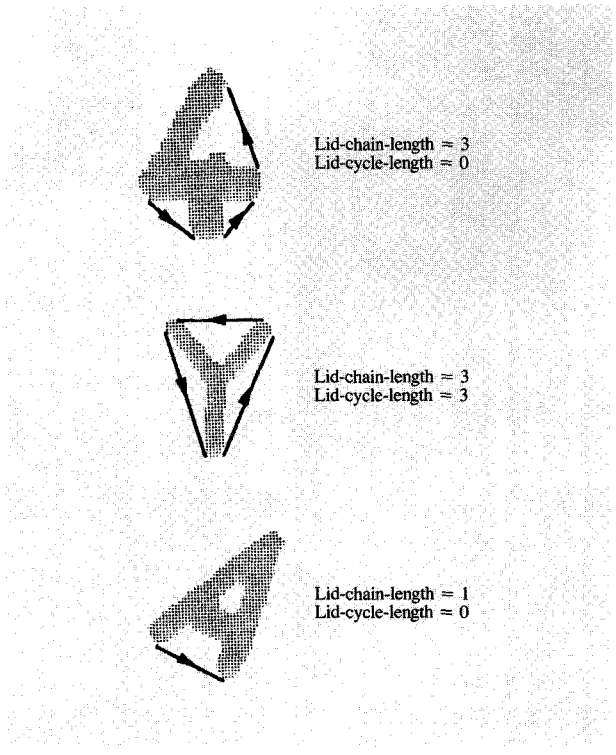

Figure 8

Lid-chain-length and lid-cycle-length values for different characters.

features in this list have not previously been discussed in the literature. The first is well known—it is mentioned here since it cooperates efficiently with the other three.

- *Complexity* is defined to be p^2/A, where p is the perimeter of the figure and A is the area. In the real plane, the "isoperimetric inequality" states that $p^2/A \geq 4\pi$ for any shape. This quantity increases when the shape becomes elongated or irregular [24].
- *Circularity* is defined to be the ratio A/C, where A is again the area of the object, and C is the area of the least circle that contains the object and is centered on its centroid (**Figure 9**).
- By the *symmetry* of a plane figure, we mean the fraction of figure points P whose symmetric P' relative to the centroid belongs to the figure.
- The *color* of the centroid is a feature which takes values 1 or 0 according to whether or not the centroid of the object belongs to the object.

Let us comment briefly upon these features with reference to our criteria for good features:

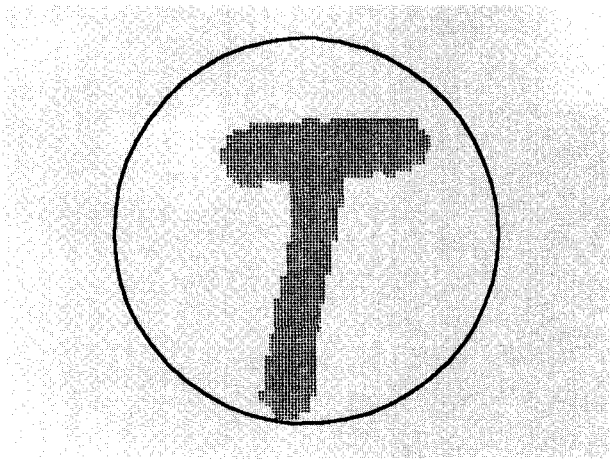

Figure 9

Circularity.

- Complexity (as well as perimeter) is considerably affected by the sampling rate. Also, since perimeter is sensitive to noise, this feature has relatively high variance over the symbol subpopulations.
- Circularity is a reasonable feature, with relatively high discriminating capability. It becomes very good when the centroid of the object is replaced by the centroid of its convex hull. This last is much more stable (it is not affected by the width of the lines); thus, the variance of the feature over symbol subpopulations is lowered (a different measure of circularity is proposed in [26]).
- Symmetry is not very good as a feature. Its variance over the symbol subpopulations is high (it is very sensitive to stretching).
- The color of the centroid is a very interesting feature. It has very small variance on certain symbols such as I and O, and high variance on certain other symbols such as F and G. It can be useful if the recognition strategy is appropriate (this is discussed further in Section 5).

We suggest a special strategy for the use of these features in the next section. We find it convenient, however, to emphasize an important issue at this point. We must be able to use a feature on only those shapes for which it shows a stable behavior. For example, it should be possible for us to use the color of the centroid on such shapes as O and I, but not on such shapes as F and G.

5. How to use the features

- *Principles of recognition theory*

The task of any recognition application is to assign certain objects to given classes $C_0, C_1, \cdots, C_m$ according to the values of certain features $X_1, \cdots, X_n$. It is understood that the objects to be recognized are value-bearing individuals for $X_1, \cdots, X_n$; i.e., these quantities can be measured, or computed, on each of these objects.

$C_1, \cdots, C_m$ are the classes of interest, or *proper* classes. In a character recognition application, $C_1, \cdots, C_m$ represent the different shapes to be discriminated (each character or symbol usually corresponds to more than one shape). C_0 is a special class, called the *reject* class; if an object is classified as belonging to C_0, it actually means that the recognizer is unable to classify it as belonging to exactly one of the proper classes $C_1, \cdots, C_m$.

$X_1, \cdots, X_n$ can be regarded as coordinates in an n-dimensional *feature space*, and the objects to be recognized can be represented by points in that space. (This notion of feature space is very general; it is just the Cartesian product of the ranges of the n features.) In this view, building up a recognizer amounts to defining m subsets of the feature space, not necessarily pairwise disjoint, to be used as *decision regions* for the proper classes. An object is classified as belonging to a proper class if its representative point P lies only in the corresponding decision region. If P lies in more than one, or none, of these regions, the recognizer realizes that it is impossible to assign the object to exactly one proper class, and rejects the object.

In the statistical approach to recognition, the feature space is usually taken to be a real n-dimensional Euclidean space, and the decision regions are usually defined by putting thresholds on certain class probability density functions. For example, if these density functions are normal, by putting thresholds on them we obtain decision regions that are hyperellipsoids. The density functions, in turn, are estimated on suitable training samples (one sample for each proper class). For example, if the density functions are normal, an n-vector of means and an $n \times n$ covariance matrix must be estimated for each of the proper classes. Incidentally, if n is large, huge amounts of sample data are needed.

- *Suggested strategy*

For various reasons, the statistical approach is not appropriate for exploiting topological and shape features. More generally, it is not suited for expressing requirements on the topology of a figure.

We set forth the following two design issues for a recognizer that uses shape features for classifying plane shapes:

1. We suggest that a character recognizer should be implemented as a question-answering system in which the requirements for a symbol to be classified as belonging to C_i, $i = 1, \cdots, m$, are stored as axioms in some logical language, and the question to be answered

(which of the C_is is correct) is stated as a theorem to be proved (or refused) by means of the inference rules available in the language. Thus, the approach that we advocate will make a recognizer very much like a knowledge-based system.

2. Each requirement set forth for a symbol to belong to class C_i, $i = 1, \cdots, m$, should be as stringent as possible, with the constraint that the shapes actually belonging to C_i must satisfy that requirement with probability 1. This device allows the addition of more and more requirements with almost no danger that the performance may begin to deteriorate at some specific point. (This is discussed further in the next subsection.)

Before we give further details about these design issues, we find it convenient to illustrate with an example the recognition strategy we suggest.

The following is a (very simplified) set of requirements that must be met by a symbol in order to be considered as a candidate T. With the notations given in **Figure 10,**

1. Number of lakes = 0.
2. Number of large bays = 2.
3. Lid-chain-length = 2.
4. Lid-torsion = 0.
5. Neither of the two lids is twice as long as the other.
6. The angle between $\overline{C_1C_2}$ and $\overline{C_3C_4}$ is greater than 90°.

A few remarks may further clarify the situation:

- As long as we can rely upon a feature extractor that can detect almost all actual lakes and actual large bays, we may expect that requirements 1 and 2 are satisfied by almost all Ts, i.e., with probability 1.
- Requirement 3 may fail to be satisfied if the normalized distance between C_2 and C_3 is greater than t_1. If we take $t_1 = 0.2$ as suggested above, that distance will be less than t_1 in a T even in extremely blurred images (in normalized units of length, $t_1 = 0.2$ amounts to one fifth of the whole perimeter of the convex hull of the figure).
- By the same token, requirement 4 is satisfied in almost all Ts. Indeed, $\overline{C_1C_2}$ and $\overline{C_3C_4}$ can form a twisted pair in a T if the normalized distance between C_2 and C_3 is greater than $t_2 > t_1$.
- If one of the two lids in a T is twice as long as the other, the T is so slanted that even the human eye is likely to be unable to distinguish it from a Y.
- With reference to Figure 10, it is quite evident that the angle between $\overline{C_1C_2}$ and $\overline{C_3C_4}$ can be acute only in a grossly deformed T. In other words, the probability that this angle will be acute in a T can be taken to be 0.

In a recognizer implementing issues 1 and 2, each shape to be recognized has a portion of the program (a *box*) fully dedicated to it. This box contains a collection of rules involving features whose variance over the subpopulation corresponding to the shape in question is low. Thus, in the box corresponding to O we include the requirement that the color of the centroid be white, in the box corresponding to I we require it to be black, and in the boxes corresponding to F and G we ignore the color of the centroid.

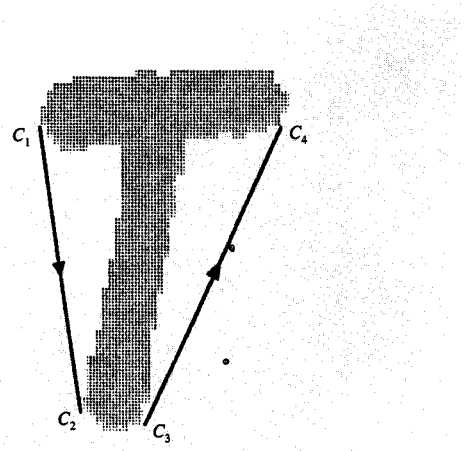

Figure 10

Shape T.

- *Formulating the rules*

In this subsection, we explain the reasons for the second of the two design issues that we set forth for a recognizer implementing shape features.

The example given in the preceding subsection shows how recognition can be done in a rule-based character recognition system. It was seen that decision making is possible provided certain numerical values are specified. Thus, in rules 1 through 4 certain numerical values are specified for certain features, while in rules 5 and 6 two thresholds occur, one associated with the ratio between the lengths of the lids and the other with the angle between the same lids.

In general, we expect to have rules of the form

$$a \leq f \leq b, \tag{1}$$

where f is any one of the features on which the recognition system is based.

Regarding the actual use of such a system, we must demonstrate, or at least explain, how these numerical values can be determined. For example, with reference to rule 6 in the above example, how do we know that the

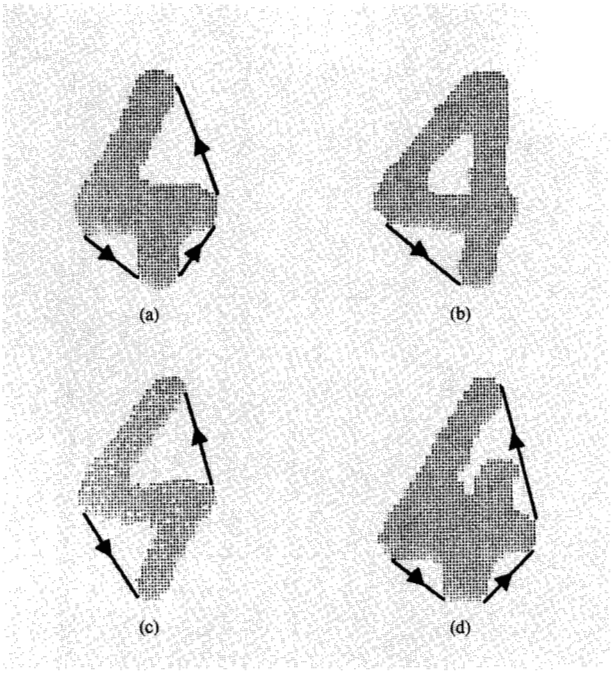

Figure 11

Shapes corresponding to **4**.

angle between $\overline{C_1C_2}$ and $\overline{C_3C_4}$ must be greater than 90° and not, say, 100°?

Traditionally, these numerical values are estimated from samples of observations. In principle, we also suggest some moderate use of training samples. However, we strongly recommend a special *modus operandi*, as explained below.

A commonly occurring situation in the design of statistical recognition systems is the following. Intuitively, one expects the classification error rate to decrease if the dimensionality of the feature vector is increased, or at worst, if the added features contain no new information, to remain unchanged. However, this is not always the case. In practice, the performance of a classifier often reaches a peak corresponding to a certain set of features, and decreases if more features are added [27].

This phenomenon is connected with the fact that the sample sets that we can use are finite, allowing for the estimation of a limited number of parameters. Adding new features requires new parameters to be estimated, and eventually the accumulated imprecision of the estimates becomes too great. This phenomenon is by no means confined to statistical recognition systems. It can occur whenever we introduce new items such as a and b in Equation (1), i.e., new parameters that are estimated from samples.

Suppose, however, that we take for a a value not greater than α, and for b a value not less than β, where α and β are the ideal values that would be estimated by means of an infinite training sample. Our values for a and b would actually be independent of the sample size. If the values for a and b in Equation (1) are estimated this way, we may expect that the addition of the requirement represented by Equation (1) does not imply that we are asking too much of a finite sample; thus, it seems reasonable that we should not see any degradation.

That is what we actually observed. If we assume ranges that are almost surely satisfied, i.e., are satisfied with probability 1, we may have no improvement in performance, but we also have no degradation.

This criterion can be implemented in the following procedure for evaluating the interval parameters in the classifier rules. We determine a, b in Equation (1) from a (very large) training sample as follows. The sample is searched for the characters that bear the smallest (largest) value of f; if the human eye can recognize one of these characters, the corresponding value of f is taken to be the value of a (of b) in Equation (1). If a character cannot be recognized by the human eye, the characters are discarded, and the procedure is iterated with the remaining characters in the sample.

The critical premise of this method of rule formulation is the following: Let us again refer to Equation (1) as the typical rule. Quite obviously, the discriminating power of a rule of this form is diminished if we take a so small and b so large that $a \leq \alpha$ and $b \geq \beta$ both hold with probability 1.

However, with the recognition strategy that we suggest, the ability to discriminate shapes is an outcome of the accumulation of numerous rules of the form (1). In turn, such an accumulation of rules is made possible precisely by choosing a, b satisfying $a \leq \alpha$ and $b \geq \beta$ with probability 1.

- *Decomposition of symbols into shapes*

In general, each character, or sign, to be recognized can be written in many shapes and forms. For this reason, splitting the character or sign into more than one shape is useful. (The versions of a character or sign which are to be regarded as different from one another depend on the feature set that we choose for recognition.)

To the recognizer, these shapes appear as different characters to be discriminated. For example, **Figure 11** shows four distinct shapes, all corresponding to **4**: The recognizer treats these shapes as different symbols.

A shape identification number (SIN) is assigned to each shape. The recognizer returns a set of SINs corresponding to each query symbol. Each SIN is then translated into the corresponding set of character identification numbers (CINs). (In the great majority of cases, this last set will contain exactly one CIN. However, it is possible that a

shape may be shared by more than one character, in which case the set will contain more than one element.)

• *Computing the shape features*

The computation of features invariant under many possible motions is intrinsically complex, and may require a significant amount of time. The OCR algorithm using decision trees by Casey and Nagy [28] is a good example to illustrate this fact. This is one of the most efficient algorithms reported in the literature. However, the speed of recognition depends on the quality of the input; if the input is distorted or noisy, the time required for recognition is noticeably increased [1].

It is advisable to choose an image representation suited to the particular computations to be performed. For example, Freeman's chain code representation of the border is well suited for features based on bays, corners, and other singularities of the object contour [29].

Another image representation suited for the features discussed in Section 3 is the *graph representation*. A preliminary report on this image representation can be found in [30], and a detailed study is in preparation.*

However, even if the image representation is carefully chosen, the computational complexity of the features suggested in Section 3, particularly of those based on the convex hull, is high; thus, an OCR system based on these features is condemned to be slow. For vectorization applications such as the one described in Section 2, the time required for character recognition is not critical: The overall time required for processing one drawing is bound to the vectorization process.

Indeed, there are about two thousand characters and special signs in one map, so, even with a slow OCR subsystem, the time required for character recognition is a small fraction of that required by the raster-to-vector conversion (besides, the two processing tasks are independent, and can be done in parallel).

• *Use of context*

Quite obviously, the methods discussed in this paper do not allow for discriminating between, say, 6 and 9, q and b, or d and p. In general, if two symbols can be mapped into one another by a rotation, they cannot be discriminated by these techniques.

In all cases in which it is impossible to use some context, e.g., when the symbols are isolated, the recognizer rejects these symbols. All rejected symbols are submitted to an operator for visual recognition.

If a symbol rejected by the recognizer belongs to a string, and the other elements of the string are unambiguously classified, the information available is in general sufficient for resolving the ambiguity (possibly with the aid of a dictionary).

*S. Di Zenzo, "A New Binary Image Representation," unpublished work.

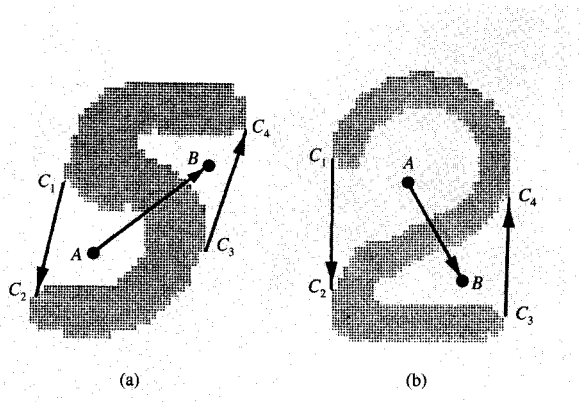

Figure 12

Discriminating between mirror images.

6. Examples

• *Discriminating between mirror images*

It is well known that shape is in general not preserved in mirror reflections. For example, an S is carried into a different shape S′ by a mirror reflection: S′ is quite different from S, since there is no similarity transformation of the plane that will carry S′ back into S. In a very noisy and blurred image, S′ might be a 2. Similarly, a b is mapped into a d by a mirror reflection.

A recognizer of rotated characters should embody the capability to discriminate between such mirror images with an expected error rate very close to 0. None of the features discussed above can distinguish between mirror images. Indeed, all the features discussed until now are not sensitive to mirror reflections.

However, it is possible by means of lids (or sides) to achieve an almost sure discrimination between mirror images.

We give two examples.

To discriminate between S and 2 we can proceed as follows (**Figure 12**): The list of requirements for a symbol to be classified as an S includes the following items:

- There should be at least two large bays.
- The vector from the centroid of the first bay to that of the second should form an obtuse angle with the lid of the first bay.

Bays are arranged in order of decreasing area. However, it is easy to verify that this criterion is independent of which

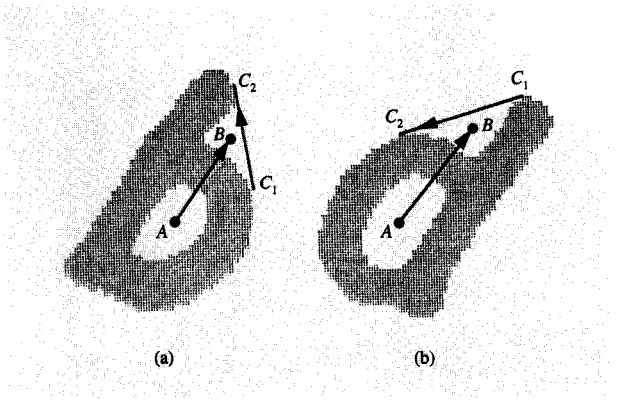

Figure 13

Discriminating between mirror images.

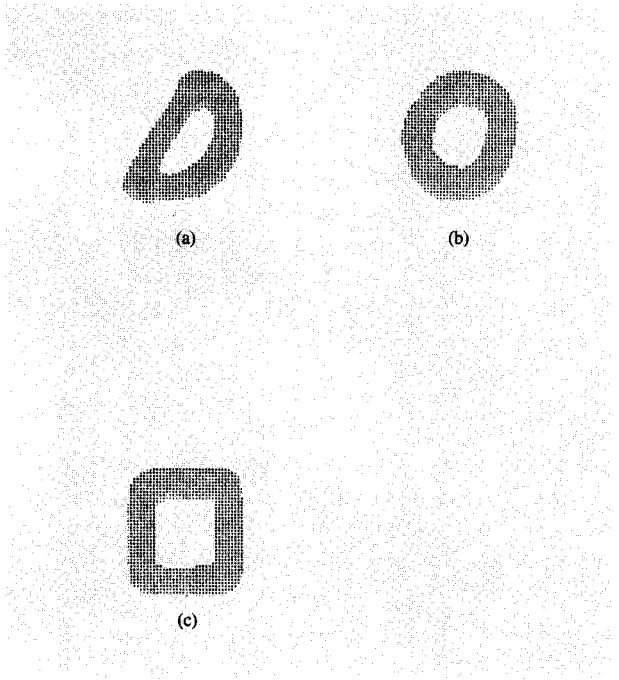

Figure 14

Examples of the characters D and O.

of the two largest bays is taken as the first. On the other side, the list of requirements for the symbol 2 includes the following:

- There should be at least two large bays.
- The vector from the centroid of the first bay to that of the second should form an acute angle with the lid of the first bay.

Our second example illustrates the discrimination between b and d. We can proceed as follows (**Figure 13**): The list of requirements for the symbol b includes the following items:

- There should be one lake and at least one bay.
- The vector from the centroid of the lake to that of the first bay should form an acute angle with the lid of the first bay.

The list of requirements for the symbol d includes the following:

- There should be one lake and at least one bay.
- The vector from the centroid of the lake to that of the first bay should form an obtuse angle with the lid of the first bay.

It is self-evident, and is confirmed by our experiments, that these requirements for b and d are satisfied even in grossly deformed bs and ds. We have experienced no error in discriminating between mirror images in all of the experiments done so far.

- *Discriminating between D and O*

In early OCR systems, discriminating between D and O was a critical subtask. This is no longer true, at least for unrotated characters. Thus, the misclassification rate of D and O in most of the commercially available OCR programs that rely upon knowledge of the baseline is about average. If the characters are rotated, the discrimination between D and O is still somewhat critical [4].

In this example, we discuss the discrimination of D and O in the case of hand-printed characters of any size, position, and orientation (**Figure 14**).

The following is the relevant subset of requirements that should be met by a symbol to qualify as a candidate O [Figure 14(b)]:

1. Number of lakes = 1.
2. Number of large bays = 0.
3. Color of centroid = 0.
4. Symmetry ≥ 0.3.
5. Circularity ≥ 0.4.
6. Side-chain-length ≤ 1.
7. Maximum side-length ≤ 0.3.

The discrimination procedure is as follows:

- As in the example given in the subsection on strategy, we assume that our system is equipped with a feature extractor able to discard almost all noisy lakes and bays. If that facility is in fact available, we may expect that requirements 1 and 2 are satisfied by almost all Os.

- The color of the centroid is useful when it is almost surely 0 or almost surely 1. In other cases this feature should not be used.
- The theoretical value of both circularity and symmetry for a perfectly round O is 1. This theoretical value is replaced by the interval (0.3, 1) to account for noise and distortion: This enlargement of a possible range of values for a feature is very well suited to illustrate what we would call the "probability = 1" strategy.
- With reference to requirement 6, we assume that an O may exhibit one or more sides, but not a pair of sides that are both consecutive and near. This is a weak assumption for Os that, at least in the intention of the writer, are round. Note that an O like that in Figure 14(c) must be regarded as a new symbol, and must be represented by a separate shape.

The first three requirements for D are exactly the same as for O, and we do not repeat them. Requirements 4–7 are replaced by the following:

4′. $0.2 \leq$ symmetry ≤ 0.95.
5′. $0.3 \leq$ circularity ≤ 0.7.
6′. Side-chain-length ≥ 1.
7′. Maximum-side-length ≥ 0.3.

Let us try to explain the thresholds in 4′–7′. We imagine that the convex hull of an ideal D is just a semicircle. Then, an obvious computation shows that the normalized length of the straight portion of the border is

$$\frac{2}{2+\pi} \simeq 0.4.$$

With reference to this ideal shape of a D, one finds a theoretical circularity of 0.423. The theoretical value of symmetry depends on the line width, with a maximum of 0.931 when the line width is so large that the lake disappears and the whole symbol becomes a semicircle. Obviously, these very theoretical values serve as reference values. In this context, here are a few comments on rules 4′–7′:

- It is highly improbable that a distorted figure will exhibit a symmetry value greater than the theoretical one. This explains why the upper bound of the symmetry in 4′ is not much greater than the theoretical value. The lower bound for symmetry is very low; experience has shown that the actual value of the symmetry of a figure whose theoretical value is less than 1 may decrease to very low values.
- The upper/lower bounds of circularity in 5′ are the maximum/minimum circularity values measured on isolated Ds recognized as such by the human eye.

Table 1 Character classification results for a total of 71 796 characters.

Result of classification	*Number of characters*	*Percentage of total characters*
Successful recognition	66387	92.47
Ambiguous classification	195	0.27
Strict rejection	4722	6.57
Substitution error	492	0.69

- It is required that a D must have a side of normalized length at least 0.3. For an O, which theoretically should not have sides, a side of length at most 0.3 is tolerated. Thus, the decision regions of O and D turn out to be disjoint. This choice excludes ambiguity between these two symbols (which would produce a reject). Hence, success or misclassification error are the only possibilities.

7. Experiments

Table 1 summarizes the classification results over a sample of 40 actual cadastral maps. The figures have been computed as follows.

The recognition subsystem outputs a set of SINs corresponding to each query symbol. From these, a set of CINs is computed that we call *SET*. If $SET = \varnothing$ or $|SET| > 1$, the system is able to detect its inability to recognize the symbol, and will *reject* it. If $|SET| = 1$, we have *success* or *error* according to whether the unique character identification number in *SET* is correct or not.

Ambiguous classifications occur when $|SET| > 1$, strict rejection when $|SET| = \varnothing$.

Each map contains about 1800 alphameric characters and special cadastral signs. The maps come from various locations, and the estimated number of writers is 20.

8. Conclusions

In this paper, we have discussed a special OCR problem—recognizing the characters and special symbols found in land-register maps.

We have suggested a feature-based approach, using features that are not sensitive to rotations, translations, and scaling, and are resistant to noise and distortions as well. Various features which, according to our experimentation, have these properties have been reviewed.

It has been suggested that a recognizer implementing such features can be structured as a rule-based system,

and an operational procedure has been specified for evaluating the parameters that occur in the rules.

Though in this paper we have focused on land-register maps, the approach presented here can be applied to other kinds of technical drawings. The authors have undertaken an effort to extend this OCR technology to utility maps.

Acknowledgments

We gratefully acknowledge the encouragement and advice of George Nagy, Stefano Levialdi, and Professor Herbert Freeman.

References

1. R. G. Casey and D. R. Ferguson, "Intelligent Forms Processing," *IBM Syst. J.* **29**, 435–450 (1990).
2. H. Freeman, "Computer Processing of Line-Drawing Images," *Computing Surv.* **6**, 57–97 (1974).
3. R. W. Smith, "Computer Processing of Line Images: A Survey," *Pattern Recogn.* **20**, 7–15 (1987).
4. R. M. Brown, T. H. Fay, and C. L. Walker, "Handprinted Symbol Recognition System," *Pattern Recogn.* **21**, 91–118 (1988).
5. D. Casasent and D. Psaltis, "Position, Rotation, and Scale Invariant Optical Correlation," *Appl. Opt.* **7**, 1795–1799 (1976).
6. S. Kahan, T. Pavlidis, and H. S. Baird, "On the Recognition of Printed Characters of Any Font and Size," *IEEE Trans. Pattern Anal. & Machine Intell.* **PAMI-9**, 274–288 (1987).
7. G. Nagy, "Optical Character Recognition. Theory and Practice," *Handbook of Statistics, Vol. 2*, P. R. Krishnaiah and L. N. Kanal, Eds., North-Holland Publishing Co., Amsterdam, 1982, pp. 621–649.
8. S. H. Unger, "A Computer Oriented Toward Spatial Problems," *Proc. IRE* **46**, 1744–1750 (1958).
9. J. H. Munson, "Experiments in the Recognition of Handprinted Text: Part 1—Character Recognition," *Proc. AFIPS 1969 Fall Joint Computer Conf.*, Washington, DC, 1968, pp. 1125–1138.
10. R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*, John Wiley & Sons, Inc., New York, 1973.
11. C. K. Chow, "An Optimum Character Recognition System Using Decision Functions," *IRE Electron. Computers* **6**, 247–257 (1957).
12. E. C. Greanias, P. F. Meagher, R. J. Norman, and P. Essinger, "The Recognition of Handwritten Numerals by Contour Analysis," *IBM J. Res. Develop.* **7**, 14–21 (1963).
13. M. K. Hu, "Visual Pattern Recognition by Moment Invariants," *IRE Trans. Info. Theory* **IT-8**, 179–187 (1962).
14. S. Mori, K. Yamamoto, and M. Yasuda, "Research on Machine Recognition of Handprinted Characters," *IEEE Trans. Pattern Anal. & Machine Intell.* **PAMI-6**, 386–405 (1984).
15. C. Y. Suen, M. Berthod, and S. Mori, "Automatic Recognition of Handprinted Characters: The State of the Art," *Proc. IEEE* **68**, 469–487 (1980).
16. T. Taxt, J. B. Olafsdottir, and M. Dahlen, "Recognition of Handwritten Symbols," *Pattern Recogn.* **23**, 1155–1166 (1990).
17. V. K. Govindan and A. P. Shivaprasad, "Character Recognition—A Review," *Pattern Recogn.* **23**, 671–683 (1990).
18. F. Nouboud and R. Plamondon, "On-Line Recognition of Handprinted Characters: Survey and Beta Tests," *Pattern Recogn.* **23**, 1031–1044 (1990).
19. C. C. Tappert, C. Y. Suen, and T. Wakahara, "On-Line Handwriting Recognition—A Survey," *Research Report RC-14045*, IBM Thomas J. Watson Research Center, Yorktown Heights, NY, 1987.
20. L. A. Fletcher and R. Kasturi, "A Robust Algorithm for Text String Separation from Mixed Text/Graphics Images," *IEEE Trans. Pattern Anal. & Machine Intell.* **10**, 910–918 (1988).
21. R. M. Haralick and D. Queeney, "Understanding Engineering Drawings," *Comput. Vision, Graph. & Image Process.* **20**, 244–258 (1982).
22. V. Nagasamy and N. A. Langrana, "Engineering Drawing Processing and Vectorization System," *Comput. Vision, Graph. & Image Process.* **49**, 379–397 (1990).
23. L. Boatto, V. Consorti, M. Del Buono, S. Di Zenzo, V. Eramo, A. Esposito, F. Melcarne, M. Meucci, A. Morelli, M. Mosciatti, S. Scarci, and M. Tucci, "An Interpretation System for Land Register Maps," *IEEE Computer*, pp. 23–33 (July 1992).
24. A. Rosenfeld and A. C. Kak, *Digital Picture Processing*, 2nd Ed., Academic Press, Inc., New York, 1982.
25. A. Y. Wu, S. K. Bhaskar, and A. Rosenfeld, "Parallel Computation of Geometric Properties from the Medial Axis Transform," *Computer Vision, Graph., & Image Process.* **41**, 323–332 (1988).
26. R. M. Haralick, "A Measure for Circularity of Digital Images," *IEEE Trans. Syst. Man Cybernet.* **SMC-4**, 394–396 (1974).
27. A. K. Jain and B. Chandrasekaran, "Dimensionality and Sample Size Considerations in Pattern Recognition," *Handbook of Statistics, Vol. 2*, P. R. Krishnaiah and L. N. Kanal, Eds., North-Holland Publishing Co., Amsterdam, 1982, pp. 835–855.
28. R. G. Casey and G. Nagy, "Decision Tree Design Using a Probabilistic Model," *IEEE Trans. Info. Theory* **IT-30**, 93–99 (1983).
29. H. Freeman, "Lines, Curves, and the Characterization of Shape," *Proceedings of IFIP Congress 80*, Tokyo, October 6–9, 1980, pp. 629–639.
30. S. Di Zenzo and A. Morelli, "A Useful Image Representation," *Progress in Image Analysis and Processing*, V. Cantoni and S. Levialdi, Eds., World Scientific Press, 1989, pp. 170–178.

Received May 8, 1991; accepted for publication March 9, 1992

Silvano Di Zenzo *IBM Southern Europe Middle East Africa Corporation, Scientific and Technical Solutions Center, viale Oceano Pacifico 173, 00144 Rome, Italy (DIZENZO at ROMESC).* Dr. Di Zenzo is director of research at the IBM SEMEA Scientific and Technical Solutions Center in Rome. He has been an assistant professor of mathematics at the University of Genoa, an IBM systems engineer, and the manager of an IBM industry project in image processing. He has worked on several research projects in image analysis and pattern recognition, both at the IBM SEMEA Scientific and Technical Solutions Center and at the IBM Palo Alto Scientific Center. Dr. Di Zenzo received the electrical engineering degree from the University of Genoa in 1962.

Monica Del Buono *IBM Southern Europe Middle East Africa Corporation, Scientific and Technical Solutions Center, viale Oceano Pacifico 173, 00144 Rome, Italy (DELBUONO at ROMESC).* Dr. Del Buono is a researcher at the IBM SEMEA Scientific and Technical Solutions Center in Rome. Her research activity is in the area of optical character recognition and intelligent document processing. Her research interests include the theory of recognition and image analysis. Dr. Del Buono received her degree in mathematics at the University of Rome in 1985.

Marco Meucci *IBM Southern Europe Middle East Africa Corporation, Scientific and Technical Solutions Center, viale Oceano Pacifico 173, 00144 Rome, Italy (MEUCCI at ROMESC).* Dr. Meucci joined the IBM Rome Scientific Center in 1988. He is currently a systems engineer working in the OCR field at the IBM SEMEA Scientific and Technical Solutions Center. Dr. Meucci's technical interests include image analysis and understanding, and user interfaces. He received his degree in mathematics at the University of Rome in 1988.

Aldo Spirito *IBM Southern Europe Middle East Africa Corporation, Scientific and Technical Solutions Center, viale Oceano Pacifico 173, 00144 Rome, Italy (SPIRITO at ROMESC).* Dr. Spirito received the electronic engineering degree from the University of Naples in 1977. From 1977 to 1979 he worked on stochastic models of air pollution at the Polytechnic of Milan and at the International Institute for Applied System Analysis in Laxenburg, Austria. In 1980 Dr. Spirito joined the IBM Rome Scientific Center, working on various research projects in the area of air pollution diffusion modeling, image processing, and character recognition. He is currently responsible for academic activities at the IBM SEMEA Scientific and Technical Solutions Center.