FEATURE EXTRACTION METHODS FOR CHARACTER RECOGNITION — A SURVEY

ØIVIND DUE TRIER,*† ANIL K. JAIN,§ and TORFINN TAXT†

†Department of Informatics, University of Oslo, P.O.Box 1080 Blindern, N-0316 Oslo, Norway
§Department of Computer Science, Michigan State University, A714 Wells Hall, East Lansing, MI 48824–1027, USA

Revised July 19, 1995

Abstract— This paper presents an overview of feature extraction methods for off-line recognition of segmented (isolated) characters. Selection of a feature extraction method is probably the single most important factor in achieving high recognition performance in character recognition systems. Different feature extraction methods are designed for different representations of the characters, such as solid binary characters, character contours, skeletons (thinned characters), or gray level subimages of each individual character. The feature extraction methods are discussed in terms of invariance properties, reconstructability, and expected distortions and variability of the characters. The problem of choosing the appropriate feature extraction method for a given application is also discussed. When a few promising feature extraction methods have been identified, they need to be evaluated experimentally to find the best method for the given application.

Feature extraction Optical character recognition Character representation Invariance Reconstructability

1 Introduction

Optical character recognition (OCR) is one of the most successful applications of automatic pattern recognition. Since the mid 1950's, OCR has been a very active field for research and development [1]. Today, reasonably good OCR packages can be bought for as little as $100. However, these are only able to recognize high quality printed text documents or neatly written hand-printed text. The current research in OCR is now addressing documents that are not well handled by the available systems, including severely degraded, omnifont machine printed text, and (unconstrained) handwritten text. Also, efforts are being made to achieve lower substitution error rates and reject rates even on good quality machine printed text, since an experienced human typist still has a much lower error rate, albeit at a slower speed.

Selection of a feature extraction method is probably the single most important factor in achieving high recognition performance. Our own interest in character recognition is to recognize hand-printed digits in hydrographic maps (Fig. 1), but we have tried not to emphasize this particular application in the paper. Given the large number of feature extraction methods reported in the literature, a newcomer to the field is faced with the following question: Which feature extraction method is the best for a given application? This question led us to characterize the available feature extraction methods, so that the most promising methods could be sorted out. An experimental evaluation of these few promising methods must still be performed to select the best method for a specific application. In this process, one might find that a specific feature extraction method needs to be further developed.

A full performance evaluation of each method in terms of classification accuracy and speed is not within the scope of this review paper. In order to study performance issues, we will have to implement all the feature extraction methods, which is an enormous task. In addition, the performance also depends on the type of classifier used. Different feature types may need different types of classifiers. Also, the classification results reported in the literature are not comparable because they are based on different data sets.

Given the vast number of papers published on OCR every year, it is impossible to include all the available feature extraction methods in this survey. Instead, we have tried to make a representative selection to illustrate the different principles that can be used.

Two-dimensional object classification has several applications in addition to character recognition. These include airplane recognition [2], recognition of mechanical parts and tools [3], and tissue classification in medical imaging [4]. Several of the feature extraction techniques described in this pa-

*Author to whom correspondence should be addressed. This work was done while Trier was visiting Michigan State University. The paper appeared in *Pattern Recognition*, Vol. 29, No. 4, pp. 641–662, 1996

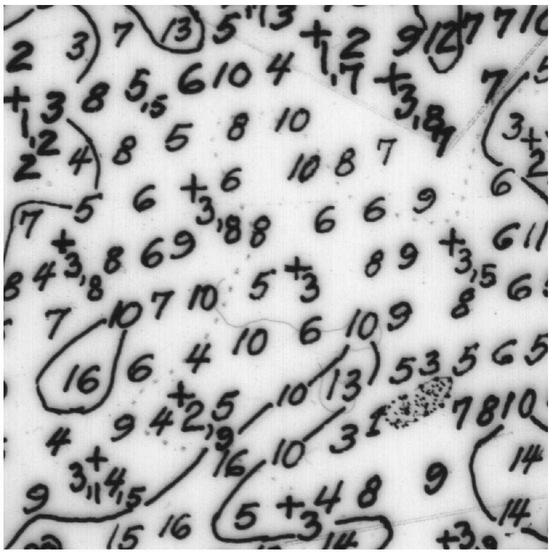


Figure 1: A gray scale image of a part of a handprinted hydrographic map.

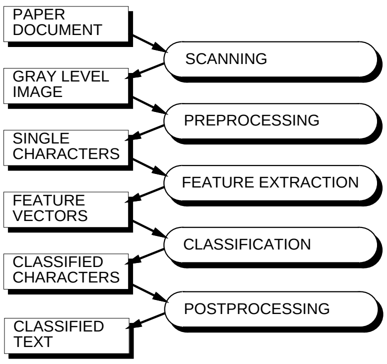


Figure 2: Steps in a character recognition system.

per for OCR have also been found to be useful in such applications.

An OCR system typically consists of the following processing steps (Fig. 2):

1. Gray level scanning at an appropriate resolution, typically 300–1000 dots per inch,
2. Preprocessing:
   (a) Binarization (two-level thresholding), using a global or a locally adaptive method,
   (b) Segmentation to isolate individual characters,
   (c) (Optional) conversion to another character representation (e.g., skeleton or contour curve),
3. Feature extraction
4. Recognition using one or more classifiers,
5. Contextual verification or postprocessing.

Survey papers [5–7], books [8–12], and evaluation studies [13–16] cover most of these subtasks, and several general surveys of OCR systems [1][17–22] also exist. However, to our knowledge, no thorough, up to date survey of feature extraction methods for OCR is available.

Devijver and Kittler define feature extraction (page 12 in [11]) as the problem of "extracting from the raw data the information which is most relevant for classification purposes, in the sense of minimizing the within-class pattern variability while enhancing the between-class pattern variability." It should be clear that different feature extraction methods fulfill this requirement to a varying degree, depending on the specific recognition problem and available data. A feature extraction method that proves to be successful in one application domain may turn out not to be very useful in another domain.

One could argue that there is only a limited number of independent features that can be extracted from a character image, so that which set of features is used is not so important. However, the extracted features must be invariant to the expected distortions and variations that the characters may have in a specific application. Also, the phenomenon called the *curse of dimensionality* [9, 23] cautions us that with a limited training set, the number of features must be kept reasonably small if a statistical classifier is to be used. A rule of thumb is to use five to ten times as many training patterns of each class as the dimensionality of the feature vector [23]. In practice, the requirements of a good feature extraction method makes selection of the best method for a given application a challenging task. One must also consider whether the characters to be recognized have known orientation and size, whether they are handwritten, machine printed or typed, and to what degree they are degraded. Also, more than one pattern class may be necessary to characterize characters that can be written in two or more distinct ways, as for example '*4*' and '4', and '*a*' and 'a'.

Feature extraction is an important step in achieving good performance of OCR systems. However, the other steps in the system (Fig. 2) also need to be optimized to obtain the best possible performance, and these steps are not independent. The choice of feature extraction method limits or dictates the nature and output of the preprocessing step (Table 1). Some feature extraction methods work on gray level subimages of single characters (Fig. 3), while others work on solid 4-connected or 8-connected symbols segmented from the binary raster image (Fig. 4), thinned symbols or skeletons (Fig. 5), or symbol contours (Fig. 6). Further, the type or format of the extracted features must match the requirements of the chosen classifier. Graph

Table 1: Overview of feature extraction methods for the various representation forms (gray level, binary, vector).

Gray scale subimage	Binary		Vector (skeleton)
	solid character	outer contour	
Template matching	Template matching		Template matching
Deformable templates			Deformable templates
Unitary Transforms	Unitary transforms		Graph description
	Projection histograms	Contour profiles	Discrete features
Zoning	Zoning	Zoning	Zoning
Geometric moments	Geometric moments	Spline curve	
Zernike moments	Zernike moments	Fourier descriptors	Fourier descriptors

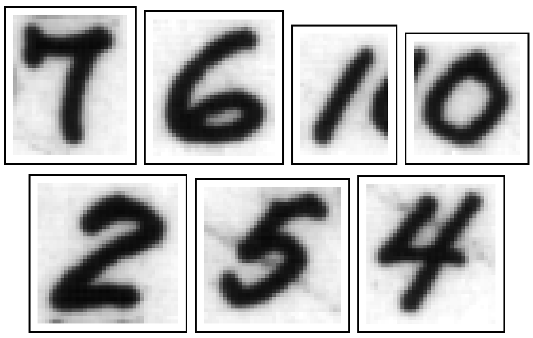


Figure 3: Gray scale subimages (≈ 30×30 pixels) of segmented characters. These digits were extracted from the top center portion of the map in Fig. 1. Note that for some of the digits, parts of other print objects are also present inside the character image.

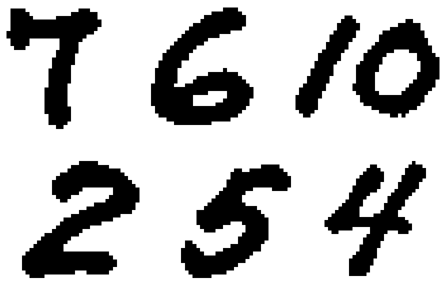


Figure 4: Digits from the hydrographic map in the binary raster representation.

descriptions or grammar-based descriptions of the characters are well suited for structural or syntactic classifiers. Discrete features that may assume only, say, two or three distinct values are ideal for decision trees. Real-valued feature vectors are ideal for statistical classifiers. However, multiple classifiers may be used, either as a multi-stage classification scheme [24, 25], or as parallel classifiers, where a combination of the individual classification results is used to decide the final classification [20, 26, 27]. In that case, features of more than one type or format may be extracted from the input characters.

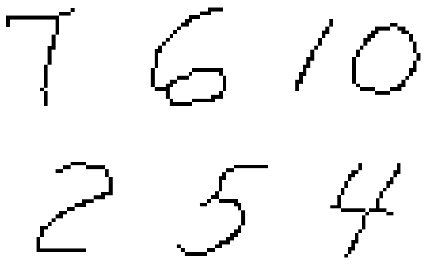


Figure 5: Skeletons of the digits in Fig. 4, thinned with the method of Zhang and Suen [28]. Note that junctions are displaced and a few short false branches occur.

1.1 Invariants

In order to recognize many variations of the same character, features that are invariant to certain transformations on the character need to be used. Invariants are features which have approximately the same values for samples of the same character that are, for example, translated, scaled, rotated, stretched, skewed, or mirrored (Fig. 7). However, not all variations among characters from the same character class (e.g., noise or degradation, and absence or presence of serifs) can be modelled by using

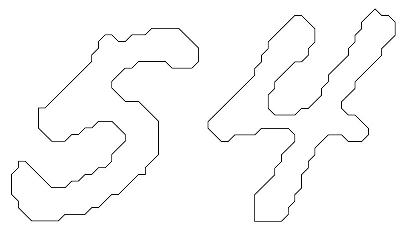


Figure 6: Contours of two of the digits in Fig. 4.

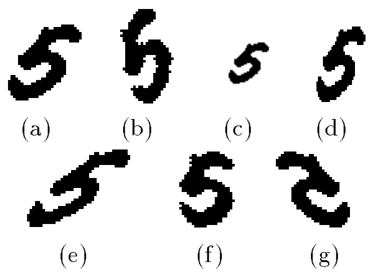


Figure 7: Transformed versions of digit '5'. **(a)** original, **(b)** rotated, **(c)** scaled, **(d)** stretched, **(e)** skewed, **(f)** de-skewed, **(g)** mirrored.

invariants.

Size- and translation invariance is easily achieved. The segmentation of individual characters can itself provide estimates of size and location, but the feature extraction method may often provide more accurate estimates.

Rotation invariance is important if the characters to be recognized can occur in any orientation. However, if all the characters are expected to have the same rotation, then rotation-variant features should be used to distinguish between such characters as '6' and '9', and 'n' and 'u'. Another alternative is to use rotation-invariant features, augmented with the detected rotation angle. If the rotation angle is restricted, say, to lie between -45° and 45°, characters that are, say 180° rotations of each other can be differentiated. The same principle may be used for size-invariant features, if one wants to recognize punctuation marks in addition to characters, and wants to distinguish between, say, '.', 'o', and 'O'; and ',' and '9'.

Skew-invariance may be useful for hand-printed text, where the characters may be more or less slanted, and multifont machine printed text, where some fonts are slanted and some are not. Invariance to mirror images is not desirable in character recognition, as the mirror image of a character may produce an illegitimate symbol or a different character.

For features extracted from gray scale subimages, invariance to contrast between print and background and to mean gray level may be needed, in addition to the other invariants mentioned above. Invariance to mean gray level is easily obtained by adding to each pixel the difference of the desired and the actual mean gray levels of the image [29].

If invariant features can not be found, an alternative is to normalize the input images to have standard size, rotation, contrast, and so on. However, one should keep in mind that this introduces new discretization errors.

1.2 Reconstructability

For some feature extraction methods, the characters can be reconstructed from the extracted features [30, 31]. This property ensures that complete information about the character shape is present in the extracted features. Although, for some methods, exact reconstruction may require an arbitrarily large number of features, reasonable approximations of the original character shape can usually be obtained by using only a small number of features with the highest information content. The hope is that these features also have high discrimination power.

By reconstructing the character images from the extracted features, one may visually check that a sufficient number of features is used to capture the essential structure of the characters. Reconstruction may also be used to informally control that the implementation is correct.

The rest of the paper is organized as follows. Sections 2–5 give a detailed review of feature extraction methods, grouped by the various representation forms of the characters. A short summary on neural network classifiers is given in Section 6. Section 7 gives guidelines for how one should choose the appropriate feature extraction method for a given application. Finally, a summary is given in Section 7.

2 Features Extracted From Gray Scale Images

A major challenge in gray scale image-based methods is to locate candidate character locations. One can use a locally adaptive binarization method to obtain a good binary raster image, and use connected components of the expected character size to locate the candidate characters. However, a gray scale-based method is typically used when recognition based on the binary raster representation fails, so the localization problem remains unsolved for difficult images. One may have to resort to the brute force approach of trying all possible locations in the image. However, one then has to assume a standard size for a character image, as the combination of all character sizes and locations is computationally prohibitive. This approach can not be used if the character size is expected to vary.

The desired result of the localization or segmentation step is a subimage containing one character, and, except for background pixels, *no other objects.* However, when print objects appear very close to each other in the input image, this goal can not always be achieved. Often, other characters or print objects may accidentally occur inside the subimage (Fig. 3), possibly distorting the extracted features. This is one of the reasons why every character recognition system has a *reject* option.

2.1 Template matching

We are not aware of OCR systems using template matching on gray scale character images. However, since template matching is a fairly standard image processing technique [32, 33], we have included this section for completeness.

In template matching the feature extraction step is left out altogether, and the character image itself is used as a "feature vector". In the recognition stage, a similarity (or dissimilarity) measure between each template T_j and the character image Z is computed. The template T_k which has the highest similarity measure is identified, and if this similarity is above a specified threshold, then the character is assigned the class label k. Else, the character remains unclassified. In the case of a dissimilarity measure, the template T_k having the *lowest* dissimilarity measure is identified, and if the dissimilarity is *below* a specified threshold, the character is given the class label k.

A common dissimilarity measure is the *mean square distance* D (Eq. 20.1-1 in Pratt [33]):

$$D_j = \sum_{i=1}^{M} (Z(x_i, y_i) - T_j(x_i, y_i))^2, \qquad (1)$$

where it is assumed that the template and the input character image are of the same size, and the sum is taken over the M pixels in the image.

Eqn. (1) can be rewritten as

$$D_j = E_Z - 2R_{ZT_j} + E_{T_j}, \qquad (2)$$

where

$$E_Z = \sum_{i=1}^{M} \left(Z^2(x_i, y_i)\right), \qquad (3)$$

$$R_{ZT_j} = \sum_{i=1}^{M} (Z(x_i, y_i)T_j(x_i, y_i)), \qquad (4)$$

$$E_{T_j} = \sum_{i=1}^{M} \left(T_j^2(x_i, y_i)\right). \qquad (5)$$

E_Z and E_{T_j} are the total character image energy and the total template energy, respectively. R_{ZT_j} is the cross-correlation between the character and the template, and could have been used as a similarity measure, but Pratt [33] points out that R_{ZT_j} may detect a false match if, say, Z contains mostly high values. In that case, E_Z also has a high value, and it could be used to normalize R_{ZT_j} by the expression $\tilde{R}_{ZT_j} = R_{ZT_j}/E_Z$. However, in Pratt's formulation of template matching, one wants to decide whether the template is present in the image (and get the locations of each occurrence). Our problem is the opposite: find the template that matches the character image best. Therefore, it is more relevant to normalize the cross-correlation by dividing it with the total template energy:

$$\hat{R}_{ZT_j} = \frac{R_{ZT_j}}{E_{T_j}}. \qquad (6)$$

Experiments are needed to decide wether D_j or $\hat{R}_{ZT_j}$ should be used for OCR.

Although simple, template matching suffers from some obvious limitations. One template is only capable of recognizing characters of the same size and rotation, is not illumination-invariant (invariant to contrast and to mean gray level), and is very vulnerable to noise and small variations that occur among characters from the same class. However, many templates may be used for each character class, but at the cost of higher computational time since every input character has to be compared with every template. The character candidates in the input image can be scaled to suit the template sizes, thus making the recognizer scale-independent.

2.2 Deformable Templates

Deformable templates have been used extensively in several object recognition applications [34, 35]. Recently, Del Bimbo et al. [36] proposed to use deformable templates for character recognition in gray scale images of credit card slips with poor print quality. The templates used were character skeletons. It is not clear how the initial positions of the templates were chosen. If all possible positions in the image were to be tried, then the computational time would be prohibitive.

2.3 Unitary Image Transforms

In template matching, all the pixels in the gray scale character image are used as features. Andrews [37] applies a unitary transform to character images, obtaining a reduction in the number of features while preserving most of the information about the character shape. In the transformed space, the pixels are ordered by their variance, and the pixels with the highest variance are used as features. The unitary transform has to be applied to a training set to obtain estimates of the variances of the pixels in the transformed space. Andrews investigated the Karhunen-Loeve (KL), Fourier, Hadamard (or Walsh), and Haar transforms in 1971 [37]. He concluded that the KL transform was too computationally demanding, so he recommended to use the Fourier or Hadamard transforms. However, the KL transform is the only (mean-squared error) optimal unitary transform in terms of information compression [38]. When the KL transform is used, the same amount of information about the input character image is contained in fewer features compared to any other unitary transform.

Other unitary transforms include the Cosine, Sine, and Slant transforms [38]. It has been shown that the Cosine transform is better in terms of information compression (e.g., see pp. 375–379 in [38]) than the other non-optimal unitary transforms. Its computational cost is comparable to that of the fast Fourier transform, so the Cosine transform has been coined "the method of choice for

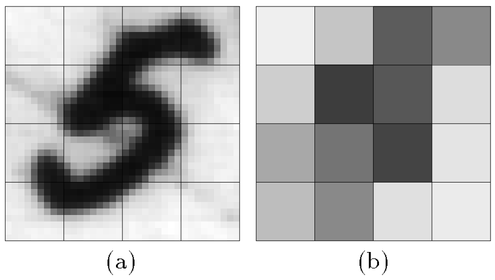


Figure 8: Zoning of gray level character images. **(a)** A 4×4 grid superimposed on a character image. **(b)** The average gray levels in each zone, which are used as features.

image data compression" [38].

The KL transform has been used for object recognition in several application domains, for example face recognition [39]. It is also a realistic alternative for OCR on gray level images with today's fast computers.

The features extracted from unitary transforms are not rotation-invariant, so the input character images have to be rotated to a standard orientation if rotated characters may occur. Further, the input images have to be of exactly the same size, so a scaling or re-sampling is necessary if the size can vary. The unitary transforms are not illumination invariant, but for the Fourier transformed image the value at the origin is proportional to the average pixel value of the input image, so this feature can be deleted to obtain brightness invariance. For all unitary transforms, an inverse transform exists, so the original character image can be reconstructed.

2.4 Zoning

The commercial OCR system by Calera described in Bokser [40] uses zoning on solid binary characters. A straightforward generalization of this method to gray level character images is given here. An $n \times m$ grid is superimposed on the character image (Fig. 8(a)), and for each of the $n \times m$ zones, the average gray level is computed (Fig. 8(b)), giving a feature vector of length $n \times m$. However, these features are not illumination invariant.

2.5 Geometric Moment Invariants

Hu [41] introduced the use of moment invariants as features for pattern recognition. Hu's *absolute orthogonal moment invariants* (invariant to translation, scale and rotation) have been extensively used (see, e.g., [29, 42, 43, 44, 45]). Li [45] listed 52 Hu invariants, of orders 2 to 9, that are translation-, scale- and rotation-invariant. Belkasim et al. [43] listed 32 Hu invariants of orders 2 to 7. However, Belkasim et al. identified fewer invariants of orders 2 to 7 than Li.

Hu also developed moment invariants that were supposed to be invariant under general linear transformations:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}, \quad (7)$$

where

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, \ \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}. \quad (8)$$

Reiss [29] has recently shown that these Hu invariants are in fact incorrect, and provided corrected expressions for them.

Given a gray scale subimage Z containing a character candidate, the regular moments [29] of order $(p+q)$ are defined as

$$m_{pq} = \sum_{i=1}^{M} Z(x_i, y_i)(x_i)^p (y_i)^q, \quad (9)$$

where the sum is taken over all the M pixels in the subimage. The translation-invariant *central* moments [29] of order $(p+q)$ are obtained by placing the origin at the center of gravity:

$$\mu_{pq} = \sum_{i=1}^{M} Z(x_i, y_i)(x_i - \overline{x})^p (y_i - \overline{y})^q, \quad (10)$$

where

$$\overline{x} = \frac{m_{10}}{m_{00}} \quad , \quad \overline{y} = \frac{m_{01}}{m_{00}}. \quad (11)$$

Hu [41] showed that*

$$\nu_{pq} = \frac{\mu_{pq}}{\mu^{(1+\frac{p+q}{2})}}, \quad p+q \geq 2 \quad (12)$$

are scale-invariant, where $\mu = \mu_{00} = m_{00}$. From the ν_{pq}'s, rotation-invariant features can be constructed. For example, the second-order invariants are

$$\phi_1 = \nu_{20} + \nu_{02}, \quad (13)$$
$$\phi_2 = (\nu_{20} - \nu_{02})^2 + \nu_{11}^2. \quad (14)$$

Invariants for general linear transformations are computed via *relative invariants* [41, 29]. Relative invariants satisfy

$$I'_j = |A^T|^{w_j} |J|^{k_j} I_j, \quad (15)$$

where I_j is a function of the moments in the original (x, y) space, I'_j is the same function computed from the moments in the transformed (x', y') space, w_j is called the weight of the relative invariant, $|J|$ is the absolute value of the Jacobian of the transposed transformation matrix, A^T, and k_j is the order of I_j. Note that the translation vector $\mathbf{b}$ does not appear in Eq. (15) as the central moments are

*Note that Eq. (12) is written with a typographical error in Hu's paper [41].

independent of translation. To generate absolute invariants, that is, invariants ψ_j satisfying

$$\psi_j' = \psi_j, \tag{16}$$

Reiss [29] used, for linear transformations,

$$|A^T| = J \text{ and } \mu' = |J|\mu, \tag{17}$$

where $\mu = \mu_{00}$:

$$\begin{aligned} I_j' &= |J|^{w_j+k_j} I_j \text{ for } w_j \text{ even}, & (18)\\ I_j' &= J|J|^{w_j+k_j-1} I_j \text{ for } w_j \text{ odd}. & (19)\end{aligned}$$

Then, it can be shown that

$$\psi_j = \frac{I_j}{\mu^{w_j+k_j}} \tag{20}$$

is an invariant if w_j is even, and $|\psi_j|$ is an invariant if w_j is odd.

For general linear transformations, Hu [41] and Reiss [29, 42] gave the following relative invariants that are functions of the second- and third-order central moments:

$$\begin{aligned}
I_1 &= \mu_{20}\mu_{02} - \mu_{11}^2 & (21)\\
I_2 &= (\mu_{30}\mu_{03} - \mu_{21}\mu_{12})^2 \\
&\quad -4(\mu_{30}\mu_{12} - \mu_{21}^2)(\mu_{21}\mu_{03} - \mu_{12}^2) & (22)\\
I_3 &= \mu_{20}(\mu_{21}\mu_{03} - \mu_{12}^2) \\
&\quad -\mu_{11}(\mu_{30}\mu_{03} - \mu_{21}\mu_{12}) \\
&\quad +\mu_{02}(\mu_{30}\mu_{12} - \mu_{21}^2) & (23)\\
I_4 &= \mu_{30}^2\mu_{02}^3 - 6\mu_{30}\mu_{21}\mu_{11}\mu_{02}^2 \\
&\quad +6\mu_{30}\mu_{12}\mu_{02}(2\mu_{11}^2 - \mu_{20}\mu_{02}) \\
&\quad +\mu_{30}\mu_{03}(6\mu_{20}\mu_{11}\mu_{02} - 8\mu_{11}^3) \\
&\quad +9\mu_{21}^2\mu_{20}\mu_{02}^2 - 18\mu_{21}\mu_{12}\mu_{20}\mu_{11}\mu_{02} \\
&\quad +6\mu_{21}\mu_{03}\mu_{20}(2\mu_{11}^2 - \mu_{20}\mu_{02}) \\
&\quad +9\mu_{12}^2\mu_{20}^2\mu_{02} \\
&\quad -6\mu_{12}\mu_{03}\mu_{11}\mu_{20}^2 + \mu_{03}^2\mu_{20}^3 & (24)
\end{aligned}$$

Reiss found the weights w_j and orders k_j to be

$$\begin{aligned} w_1 = 2, \quad w_2 = 6, \quad w_3 = 4, \quad w_4 = 6; & \quad (25)\\ k_1 = 2, \quad k_2 = 4, \quad k_3 = 3, \quad k_4 = 5. & \quad (26)\end{aligned}$$

Then the following features are invariant under translation and general linear transformations (given by Reiss [29] in 1991 and rediscovered by Flusser and Suk [46, 47] in 1993):

$$\psi_1 = \frac{I_1}{\mu^4} \quad , \quad \psi_2 = \frac{I_2}{\mu^{10}}, \tag{27}$$

$$\psi_3 = \frac{I_3}{\mu^7} \quad , \quad \psi_4 = \frac{I_4}{\mu^{11}}. \tag{28}$$

Hu [41] implicitly used $k \equiv 1$, obtaining incorrect invariants.

Bamieh and de Figueiredo [48] have suggested the following two relative invariants in addition to I_1 and I_2:[†]

$$\begin{aligned}
J_3 &= \mu_{40}\mu_{04} - 4\mu_{31}\mu_{13} + 3\mu_{22}^2 & (29)\\
J_4 &= \mu_{40}\mu_{22}\mu_{04} - 2\mu_{31}\mu_{22}\mu_{13} \\
&\quad -\mu_{40}\mu_{13} - \mu_{04}\mu_{31}^2 - \mu_{22}^3. & (30)
\end{aligned}$$

As above, these relative invariants must be divided by $\mu^\alpha = \mu^{w+k}$ to obtain absolute invariants. Regretfully, Bamieh and de Figueiredo divided J_i by μ^w (implicitly using $k \equiv 0$), so their invariants are incorrect too.

Reiss [29, 42] also gave features that are invariant under changes in contrast, in addition to being invariant under translation and general linear transformations (including scale change, rotation and skew). The three first features are

$$\theta_1 = \frac{I_4}{\mu I_2} \ , \quad \theta_2 = \frac{I_1^2}{\mu I_3} \ , \quad \theta_3 = \frac{I_1 I_3}{I_4} \tag{31}$$

Experiments with other feature extraction methods indicate that at least 10-15 features are needed for a successful OCR system. More moment invariants (ψ_j's and θ_j's) based on higher order moments are given by Reiss [42].

2.6 Zernike Moments

Zernike moments have been used by several authors for character recognition of binary solid symbols [49, 31, 43]. However, initial experiments suggest that they are well suited for gray-scale character subimages as well. Both rotation-variant and rotation-invariant features can be extracted. Features invariant to illumination need to be developed for these features to be really useful for gray level character images.

Khotanzad and Hong [49, 31] use the amplitudes of the Zernike moments as features. A set of complex orthogonal polynomials $V_{nm}(x,y)$ is used (Eqs. 32–33)[‡]. The Zernike moments are projections of the input image onto the space spanned by the orthogonal V-functions.

$$V_{nm}(x,y) = R_{nm}(x,y)e^{jm\tan^{-1}\frac{y}{x}}, \tag{32}$$

where $j = \sqrt{-1}$, $n \geq 0$, $|m| \leq n$, $n - |m|$ is even, and

$$R_{nm}(x,y) = \sum_{s=0}^{\frac{n-|m|}{2}} \frac{(-1)^s (x^2+y^2)^{\frac{n}{2}-s}(n-s)!}{s!\left(\frac{n+|m|}{2}-s\right)!\left(\frac{n-|m|}{2}-s\right)!}. \tag{33}$$

For a digital image, the Zernike moment of order n and repetition m is given by

$$A_{nm} = \frac{n+1}{\pi}\sum_x\sum_y f(x,y)[V_{nm}(x,y)]^*, \tag{34}$$

[†]An incorrect version of I_2 is given in Bamieh and de Figueiredo's paper [48].

[‡]There is an error in [49] in equation (33): In [49], the summation is taken from $s = 0$ to $n - \frac{|m|}{2}$, however, it must be taken from $s = 0$ to $\frac{n-|m|}{2}$ to avoid $\left(\frac{n-|m|}{2} - s\right)$ becoming negative.

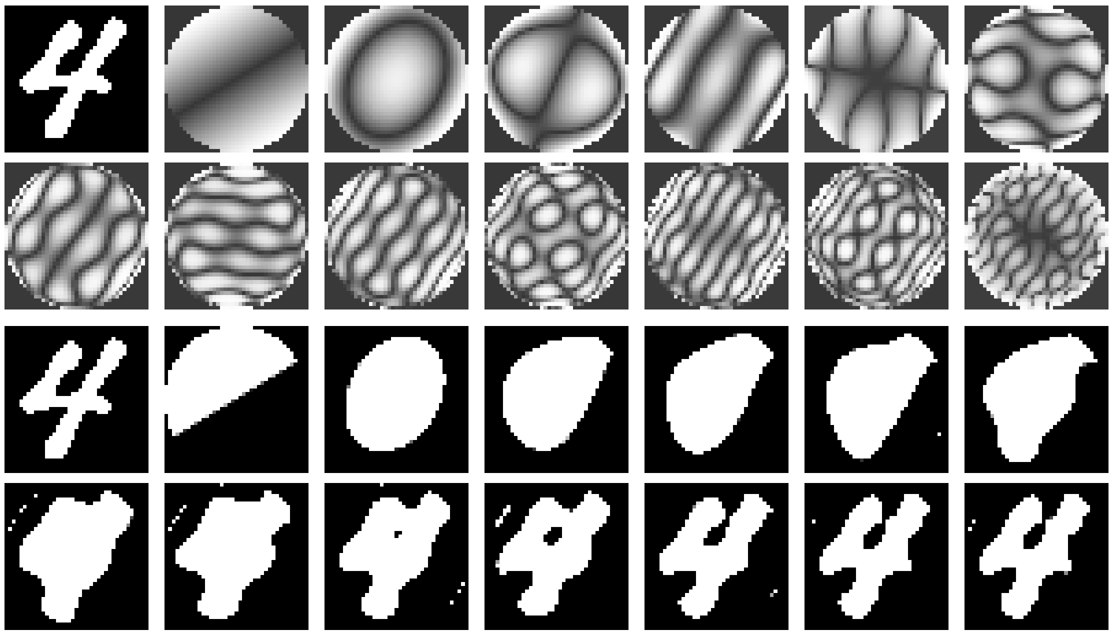


Figure 9: Images derived from Zernike moments. **Rows 1–2:** Input image of digit '4', and contributions from the Zernike moments of order 1–13. The images are histogram equalized to highlight the details. **Rows 3–4:** Input image of digit '4', and images reconstructed from the Zernike moments of order up to 1–13, respectively.

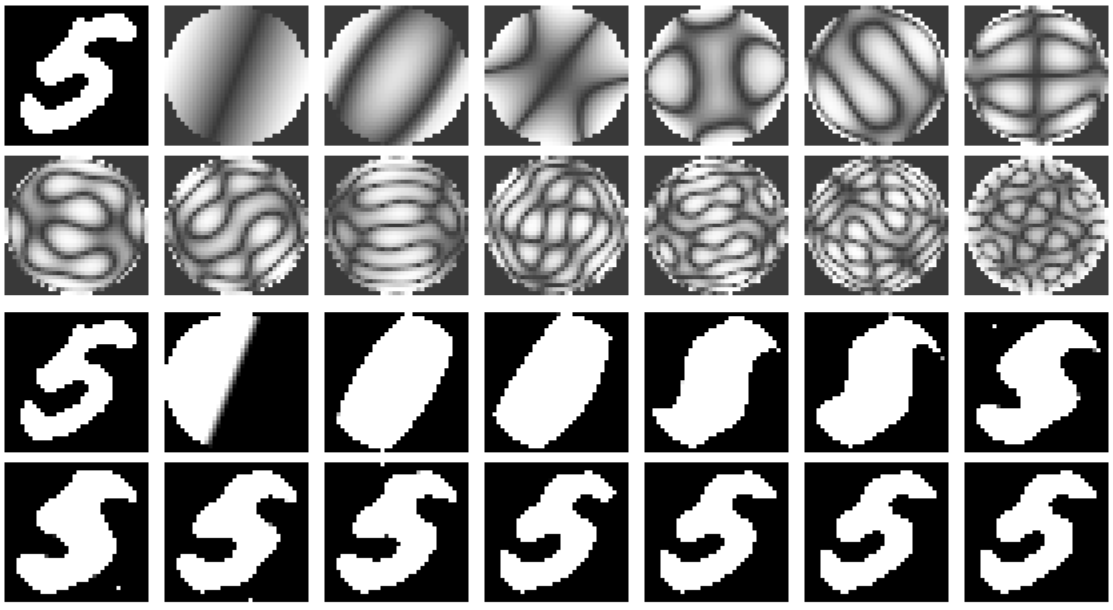


Figure 10: Images derived from Zernike moments. **Rows 1–2:** Input image of digit '5', and contributions from the Zernike moments of order 1–13. The images are histogram equalized to highlight the details. **Rows 3–4:** Input image of digit '5', and images reconstructed from the Zernike moments of order up to 1–13, respectively.

where $x^2 + y^2 \leq 1$, and the symbol $*$ denotes the complex conjugate operator. Note that the image coordinates must be mapped to the range of the unit circle, $x^2 + y^2 \leq 1$. The part of the original image inside the unit circle can be reconstructed with an arbitrary precision using

$$f(x,y) = \lim_{N\to\infty} \sum_{n=0}^{N} \sum_{m} A_{nm} V_{nm}(x,y), \qquad (35)$$

where the second sum is taken over all $|m| \leq n$, such that $n - |m|$ is even.

The magnitudes $|A_{nm}|$ are rotation invariant. To show the contribution of the Zernike moment of order n, we have computed

$$|I_n(x,y)| = \left| \sum_{m} A_{nm} V_{nm}(x,y) \right|, \qquad (36)$$

where $x^2 + y^2 \leq 1$, $|m| \leq n$, and $n - |m|$ is even.

The images $|I_n(x,y)|$, $n = 1, \ldots, 13$, for the characters '4' and '5' (Figs. 9 and 10) indicate that the extracted features are very different for the third and higher order moments. Orders one and two seem to represent orientation, width, and height. However, reconstructions of the same digits (Figs. 9 and 10) using Eq. (35), $N = 1, \ldots, 13$, indicate that moments of orders up to 8–11 are needed to achieve a reasonable appearance.

Translation- and scale-invariance can be obtained by shifting and scaling the image prior to the computation of the Zernike moments [31]. The first-order regular moments can be used to find the image center and the zeroth order central moment gives a size estimate.

Belkasim et al. [43, 44] use the following additional features

$$\begin{aligned} B_{n,n+1} &= |A_{n-2,1}||A_{n1}| \cos(\phi_{n-2,1} - \phi_{n1}), & (37) \\ B_{n,n+L} &= |A_{n1}||A_{nL}|^p \cos(p\phi_{nL} - \phi_{n1}) & (38) \end{aligned}$$

where $L = 3, 5, \ldots, n$, $p = 1/L$, and ϕ_{nm} is the phase angle component of A_{mn} so that

$$A_{mn} = |A_{mn}| \cos\phi_{mn} + j|A_{mn}| \sin\phi_{mn}. \qquad (39)$$

3 Features Extracted From Binary Images

A binary raster image is obtained by a global or locally adaptive binarization [13] of the gray scale input image. In many cases, the segmentation of characters is done simply by isolating connected components. However, for difficult images, some characters may touch or overlap each other or other print objects. Another problem occurs when characters are fragmented into two or more connected components. This problem may be alleviated somewhat by choosing a better locally adaptive binarization method, but Trier and Taxt [13] have shown that even the best locally adaptive binarization method may still not result in perfectly isolated characters.

Methods for segmenting touching characters are given by Westall and Narasimha [50], Fujisawa et al. [51], and in surveys [5, 6]. However, these methods assume that the characters appear in the same text string and have known orientation. In hydrographic maps (Fig. 1), for example, some characters touch or overlap lines, or touch characters from another text line. Trier et al. [52] have developed a method based on gray scale topographic analysis [53, 54], which integrates binarization and segmentation. This method gives a better performance, since information gained in the topographic analysis step is used in segmenting the binary image. The segmentation step also handles rotated characters and touching characters from different text strings.

The binary raster representation of a character is a simplification of the gray scale representation. The image function $Z(x,y)$ now takes on two values (say, 0 and 1) instead of the, say, 256 gray level values. This means that all the methods developed for the gray scale representation are applicable to the solid binary raster representation as well. Therefore, we will not repeat the full description of each method, but only point out the simplification in the computations involved for each feature extraction method. Generally, invariance to illumination is no longer relevant, but the other invariances are.

A solid binary character may be converted to other representations, such as the outer contour of the character, the contour profiles, or the character skeleton, and features may be extracted from each of these representations as well. For the purpose of designing OCR systems, the goal of these conversions is to preserve the relevant information about the character shape, and discard some of the unnecessary information.

Here, we only present the modifications of the methods previously described for the gray scale representation. No changes are needed for unitary image transforms and Zernike moments, except that gray level invariance is irrelevant.

3.1 Template Matching

In binary template matching, several similarity measures other than mean square distance and correlation have been suggested [55]. To detect matches, let n_{ij} be the number of pixel positions where the template pixel x is i and the image pixel y is j, with $i, j \in \{0, 1\}$:

$$n_{ij} = \sum_{m=1}^{n} \delta_m(i,j) \qquad (40)$$

where

$$\delta_m(i,j) = \begin{cases} 1 & \text{if } (x_m = i) \wedge (y_m = j) \\ 0 & \text{otherwise,} \end{cases} \qquad (41)$$

$i, j \in \{0, 1\}$, and y_m and x_m are the m-th pixels of the binary images Y and X which are being compared. Tubbs evaluated eight distances, and found the Jaccard distance d_J and the Yule distance d_Y to be the best.

$$d_J = \frac{n_{11}}{n_{11} + n_{10} + n_{01}} \quad (42)$$

$$d_Y = \frac{n_{11}n_{00} - n_{10}n_{01}}{n_{11}n_{00} + n_{10}n_{01}} \quad (43)$$

However, the lack of robustness regarding shape variations mentioned in Section 2 for the gray scale case still applies. Tubbs [55] tries to overcome some of these shortcomings by introducing weights for the different pixel positions m. Eq. (40) is replaced by

$$n_{ij} = \sum_{m=1}^{n} p_m(k|i)\delta_m(i,j), \quad (44)$$

where $p_m(k|i)$ is the probability that the input image Y matches template X_k, given that pixel number m in the template X_k is i. $p_m(k|i)$ is approximated as the number of templates (including template X_k) having the same pixel value at location m as template X_k, divided by the total number of templates.

However, we suspect that the extra flexibility obtained by introducing $p(k|i)$ is not enough to cope with variabilities in character shapes that may occur in hand-printed characters and multi-font machine printed characters. A more promising approach is taken by Gader et al. [24] who use a set of templates for each character class, and a procedure for selecting templates based on a training set.

3.2 Unitary Image Transforms

The NIST form-based hand-print recognition system [56] uses the Karhunen-Loeve transform to extract features from the binary raster representation. Its performance is claimed to be good, and this OCR system is available in the public domain.

3.3 Projection histograms

Projection histograms were introduced in 1956 in a hardware OCR system by Glauberman [57]. Today, this technique is mostly used for segmenting characters, words, and text lines, or to detect if an input image of a scanned text page is rotated [58]. For a horizontal projection, $y(x_i)$ is the number of pixels with $x = x_i$ (Fig. 11). The features can be made scale independent by using a fixed number of bins on each axis (by merging neighboring bins) and dividing by the total number of print pixels in the character image. However, the projection histograms are very sensitive to rotation, and to some degree, variability in writing style. Also, important information about the character shape seems to be lost.

The vertical projection $x(y)$ is slant invariant, but the horizontal projection is not. When measuring the dissimilarity between two histograms, it is tempting to use

$$d = \sum_{i=1}^{n} |y_1(x_i) - y_2(x_i)|, \quad (45)$$

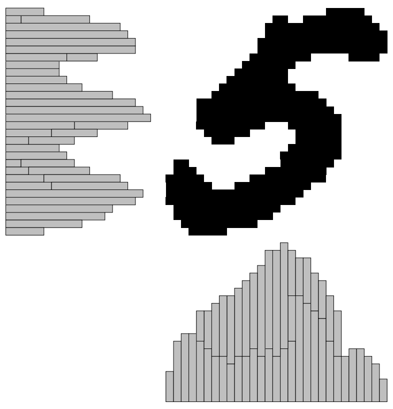


Figure 11: Horizontal and vertical projection histograms.

where n is the number of bins, and y_1 and y_2 are the two histograms to be compared. However, it is more meaningful to compare the *cumulative histograms* $Y(x_k)$, the sum of the k first bins,

$$Y(x_k) = \sum_{i=1}^{k} y(x_i), \quad (46)$$

using the dissimilarity measure

$$D = \sum_{i=1}^{n} |Y_1(x_i) - Y_2(x_i)|, \quad (47)$$

where Y_1 and Y_2 denotes cumulative histograms. The new dissimilarity measure D is not as sensitive as d to a slight misalignment of dominant peaks in the original histograms.

3.4 Zoning

Bokser [40] describes the commercial OCR system CALERA that uses zoning on binary characters. The system was designed to recognize machine printed characters of almost any non-decorative font, possibly severely degraded, by, for example several generations of photocopying. Both contour extraction and thinning proved to be unreliable for self-touching characters (Fig. 12). The zoning method was used to compute the percentage of black pixels in each zone. Additional features were needed to improve the performance, but the details were not presented by Bokser [40]. Unfortunately, not much explicit information is available about the commercial systems.

3.5 Geometric Moment Invariants

A binary image can be considered a special case of a gray level image with $Z(x,y) = 1$ for print pixels, and $Z(x_i,y_i) = 0$ for background pixels. By

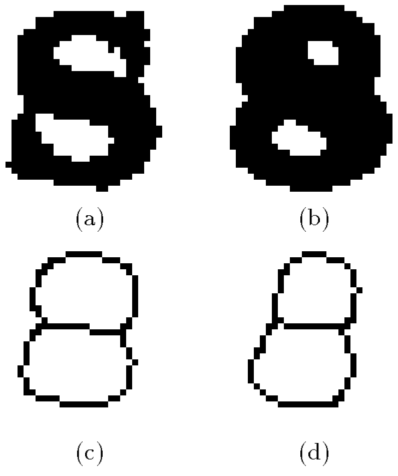


Figure 12: Two of the characters in Bokser's study [40] that are easily confused when thinned (e.g., with Zhang and Suen's method [28]). **(a)** 'S'. **(b)** '8'. **(c)** Thinned 'S'. **(d)** Thinned '8'.

summing over the N *print* pixels only, Eqs. (9)–(10) can be rewritten as

$$m_{pq} = \sum_{i=1}^{N} (x_i)^p (y_i)^q \tag{48}$$

$$\mu_{pq} = \sum_{i=1}^{N} (x_i - \overline{x})^p (y_i - \overline{y})^q, \tag{49}$$

where

$$\overline{x} = \frac{m_{1,0}}{m_{0,0}} \quad , \quad \overline{y} = \frac{m_{0,1}}{m_{0,0}}. \tag{50}$$

Then, Eqs. (12)–(24) can be used as before. However, the contrast invariants (Eq. 31) are of no interest in the binary case.

For characters that are not too elongated, a fast algorithm for computing the moments based on the character contour exists [59], giving the same values as Eq. (49).

3.6 Evaluation Studies

Belkasim et al. [43, 44] compared several moment invariants applied to solid binary characters, including regular, Hu, Bamieh, Zernike, Teague-Zernike, and pseudo-Zernike moment invariants, using a k nearest neighbor (kNN) classifier. They concluded that normalized Zernike moment invariants [43, 44] gave the best performance for character recognition in terms of recognition accuracy. The normalization compensates for the variances of the features, and since the kNN classifier uses the Euclidean distance to measure the dissimilarity of the input feature vectors and the training samples, this will improve the performance. However, by using a statistical classifier which explicitly accounts for the variances, for example, a quadratic Bayesian classifier using the Mahalanobis distance, no such normalization is needed.

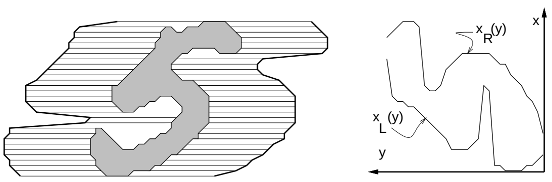


Figure 13: Digit '5' with left profile $x_L(y)$ and right profile $x_R(y)$. For each y value, the left (right) profile value is the leftmost (rightmost) x value on the character contour.

4 Features Extracted From the Binary Contour

The closed outer contour curve of a character is a closed piecewise linear curve that passes through the centers of all the pixels which are 4-connected to the outside background, and no other pixels. Following the curve, the pixels are visited in, say, counter-clockwise order, and the curve may visit an edge pixel twice at locations where the object is one-pixel wide. Each line segment is a straight line between the pixel centers of two 8-connected neighbors.

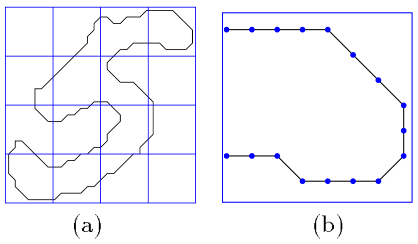
By approximating the contour curve by a parametric expression, the coefficients of the approximation can be used as features. By following the closed contour successively, a periodic function results. Periodic functions are well-suited for Fourier series expansion, and this is the foundation for the Fourier-based methods discussed below.

4.1 Contour Profiles

The motivation for using contour profiles is that each half of the contour (Fig. 13) can be approximated by a discrete function of one of the spatial variables, x or y. Then, features can be extracted from discrete functions. We may use vertical or horizontal profiles, and they can be either outer profiles or inner profiles.

To construct vertical profiles, first locate the uppermost and lowermost pixels on the contour. The contour is split at these two points. To get the outer profiles, for each y value, select the outermost x value on each contour half (Fig. 13). To get the inner profiles, for each y value, the innermost x values are selected. Horizontal profiles can be extracted in a similar fashion, starting by dividing the contour in upper and lower halves.

The profiles are themselves dependent on rotation (e.g., try to rotate the '5' in Fig. 13, say, 45° before computing the profiles). Therefore, all features derived from the profiles will also be dependent on rotation.



orientation	count
0°	9
45°	1
90°	2
135°	4

(c)

Figure 14: Zoning of contour curve. **(a)** 4×4 grid superimposed on character; **(b)** Close-up of the upper right corner zone; **(c)** Histogram of orientations for this zone.

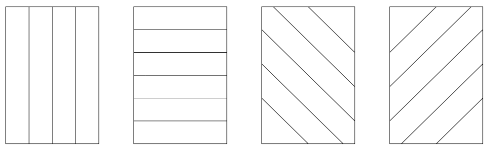


Figure 15: Slice zones used by Takahashi [60].

Kimura and Shridhar [27] extracted features from the outer vertical profiles only (Fig. 13). The profiles themselves can be used as features, as well as the first differences of the profiles (e.g, $x'_L(y) = x_L(y+1) - x_L(y)$); the width $w(y) = x_R(y) - x_L(y)$; the ratio of the vertical height of the character, n, by the maximum of the width function, $\max_y w(y)$; location of maxima and minima in the profiles; and locations of peaks in the first differences (which indicate discontinuities).

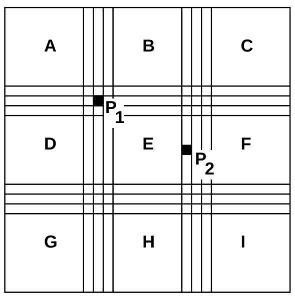


Figure 16: Zoning with fuzzy borders. Pixel P_1 has a membership value of 0.25 in each of the four zones A, B, D, and E. P_2 has a 0.75 membership of E and a 0.25 membership of F.

4.2 Zoning

Kimura and Shridhar [27] used zoning on contour curves. In each zone, the contour line segments between neighboring pixels were grouped by orientation: horizontal (0°), vertical (90°), and the two diagonal orientations (45°, 135°). The number of line segments of each orientation was counted (Fig. 14).

Takahashi [60] also used orientation histograms from zones, but used vertical, horizontal, and diagonal slices as zones (Fig. 15). The orientations were extracted from contours (if any) in addition to the outer contour when making the histograms.

Further, Takahashi identified high curvature points along both outer and inner contours. For each of these points, the curvature value, the contour tangent, and the point's zonal position were extracted. This time a regular grid was used as zones.

Cao et al. [25] observed that when the contour curve was close to zone borders, small variations in the contour curve could lead to large variations in the extracted features. They tried to compensate for this by using *fuzzy borders*. Points near the zone borders are given fuzzy membership values to two or four zones, and the fuzzy membership values sum to one (Fig. 16).

4.3 Spline curve approximation

Sekita et al. [61] identify high-curvature points, called breakpoints, on the outer character contour, and approximate the curve between two breakpoints with a spline function. Then, both the breakpoints and the spline curve parameters are used as features.

Taxt et al. [62] approximated the outer contour curve with a spline curve, which was then smoothed. The smoothed spline curve was divided into M parts of equal curve length. For each part, the average curvature was computed. In addition, the distances from the arithmetic mean of the contour curve points to N equally spaced points on the contour were measured. By scaling the character's spline curve approximation to a standard size before the features were measured, the features will become size invariant. The features are already translation invariant by nature. However, the features are dependent on rotation.

4.4 Elliptic Fourier descriptors

In Kuhl and Giardina's approach [30], the closed contour, $(x(t), y(t)), t = 1, \ldots, m$, is approximated as

$$\hat{x}(t) = A_0 + \sum_{n=1}^{N} [a_n \cos \frac{2n\pi t}{T} + b_n \sin \frac{2n\pi t}{T}] \quad (51)$$

$$\hat{y}(t) = C_0 + \sum_{n=1}^{N} [c_n \cos \frac{2n\pi t}{T} + d_n \sin \frac{2n\pi t}{T}] \quad (52)$$

where T is total contour length, and with $\hat{x}(t) \equiv x(t)$ and $\hat{y}(t) \equiv y(t)$ in the limit when $N \to \infty$. The coefficients are

$$A_0 = \frac{1}{T}\int_0^T x(t)dt \quad (53)$$

$$C_0 = \frac{1}{T}\int_0^T y(t)dt \quad (54)$$

$$a_n = \frac{2}{T}\int_0^T x(t)\cos\frac{2n\pi t}{T}dt \quad (55)$$

$$b_n = \frac{2}{T}\int_0^T x(t)\sin\frac{2n\pi t}{T}dt \quad (56)$$

$$c_n = \frac{2}{T}\int_0^T y(t)\cos\frac{2n\pi t}{T}dt \quad (57)$$

$$d_n = \frac{2}{T}\int_0^T y(t)\sin\frac{2n\pi t}{T}dt. \quad (58)$$

The functions $x(t)$ and $y(t)$ are piecewise linear, and the coefficients can, therefore, be obtained by summation instead of integration. It can be shown [30] that the coefficients a_n, b_n, c_n and d_n, which are the extracted features, can be expressed as

$$a_n = \frac{T}{2n^2\pi^2}\sum_{i=1}^{m}\frac{\Delta x_i}{\Delta t_i}[cos\phi_i - cos\phi_{i-1}] \quad (59)$$

$$b_n = \frac{T}{2n^2\pi^2}\sum_{i=1}^{m}\frac{\Delta x_i}{\Delta t_i}[sin\phi_i - sin\phi_{i-1}] \quad (60)$$

$$c_n = \frac{T}{2n^2\pi^2}\sum_{i=1}^{m}\frac{\Delta y_i}{\Delta t_i}[cos\phi_i - cos\phi_{i-1}] \quad (61)$$

$$d_n = \frac{T}{2n^2\pi^2}\sum_{i=1}^{m}\frac{\Delta y_i}{\Delta t_i}[sin\phi_i - sin\phi_{i-1}], \quad (62)$$

where $\phi_i = \frac{2n\pi t_i}{T}$,

$$\Delta x_i = x_i - x_{i-1}, \quad \Delta y_i = y_i - y_{i-1}, \quad (63)$$

$$\Delta t_i = \sqrt{\Delta x^2 + \Delta y^2}, \quad t_i = \sum_{j=1}^{i}\Delta t_j, \quad (64)$$

$$T = t_m = \sum_{j=1}^{m}\Delta t_j, \quad (65)$$

and m is the number of pixels along the boundary. The starting point (x_1, y_1) can be arbitrarily chosen, and it is clear from Eqs. (55–56) that the coefficients are dependent on this choice. To obtain features that are independent of the particular starting point, we calculate the *phase shift from the first major axis* as

$$\theta_1 = \frac{1}{2}\tan^{-1}\frac{2(a_1b_1 + c_1d_1)}{\sqrt{a_1^2 - b_1^2 + c_1^2 - d_1^2}}. \quad (66)$$

Then, the coefficients can be rotated to achieve a zero phase shift:

$$\begin{bmatrix} a_n^* & b_n^* \\ c_n^* & d_n^* \end{bmatrix} = \begin{bmatrix} a_n & b_n \\ c_n & d_n \end{bmatrix}\begin{bmatrix} \cos n\theta_1 & -\sin n\theta_1 \\ \sin n\theta_1 & \cos n\theta_1 \end{bmatrix}. \quad (67)$$

To obtain rotation invariant descriptors, the rotation, ψ_1, of the semi-major axis (Fig. 17(a)) can be found by

$$\psi_1 = \tan^{-1}\frac{c_1^*}{a_1^*} \quad (68)$$

and the descriptors can then be rotated by $-\psi_1$ (Fig. 17(b)), so that the semi-major axis is parallel with the x-axis:

$$\begin{bmatrix} a_n^{**} & b_n^{**} \\ c_n^{**} & d_n^{**} \end{bmatrix} = \begin{bmatrix} \cos\psi_1 & \sin\psi_1 \\ -\sin\psi_1 & \cos\psi_1 \end{bmatrix}\begin{bmatrix} a_n^* & b_n^* \\ c_n^* & d_n^* \end{bmatrix}. \quad (69)$$

This rotation gives $b_1^{**} = c_1^{**} = 0.0$ (Fig. 17(b)), so these coefficients should not be used as features. Further, both these rotations are ambiguous, as θ and $\theta + \pi$ give the same axes, and so do ψ and $\psi + \pi$.

To obtain size-invariant features, the coefficients can be divided by the magnitude, E, of the semi-major axis, given by

$$E = \sqrt{a_1^{*2} + c_1^{*2}} = a_1^{**}. \quad (70)$$

Then a_1^{**} should not be used as a feature as well. In any case, the low-order coefficients that are available contain the most information (about the character shape), and should always be used.

In Figs. 18–19, the characters '4' and '5' of Fig. 6 has been reconstructed using the coefficients of order up to n for different values of n. These figures suggest that using only the descriptors of the first three orders (12 features in total) might not be enough to obtain a classifier with sufficient discrimination power.

Lin and Hwang [63] derived rotation-invariant features based on Kuhl and Giardina's [30] features:

$$I_k = a_k^2 + b_k^2 + c_k^2 + d_k^2 \quad (71)$$

$$J_k = a_kd_k - b_kc_k \quad (72)$$

$$K_{1,j} = (a_1^2 + b_1^2)(a_j^2 + b_j^2) + (c_1^2 + d_1^2)(c_j^2 + d_j^2) + 2(a_1c_1 + b_1d_1)(a_jc_j + b_jd_j) \quad (73)$$

As above, a scaling factor may be used to obtain size-invariant features.

4.5 Other Fourier Descriptors

Prior to Kuhl and Giardina [30], and Lin and Hwang [63], other Fourier descriptors were developed by Zahn and Roskies [64] and Granlund [65].

In Zahn and Roskies' method [64], the angular difference $\Delta\varphi$ between two successive line segments on the contour is measured at every pixel center along the contour. The contour is followed clockwise. Then the following descriptors can be extracted:

$$a_n = -\frac{1}{n\pi}\sum_{k=1}^{m}\Delta\varphi_k \sin\frac{2\pi n t_k}{T} \quad (74)$$

$$b_n = \frac{1}{n\pi}\sum_{k=1}^{m}\Delta\varphi_k \cos\frac{2\pi n t_k}{T}, \quad (75)$$

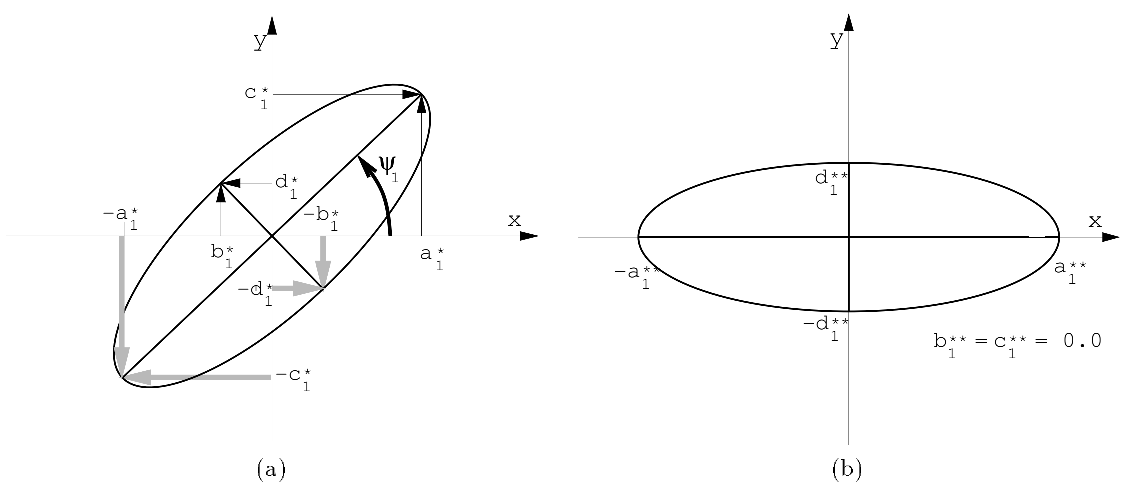


Figure 17: The rotation of the first-order ellipse used in elliptic Fourier descriptors in order to obtain rotation-independent descriptors a_1^{**} and d_1^{**}. **(a)** Before rotation, **(b)** After rotation.

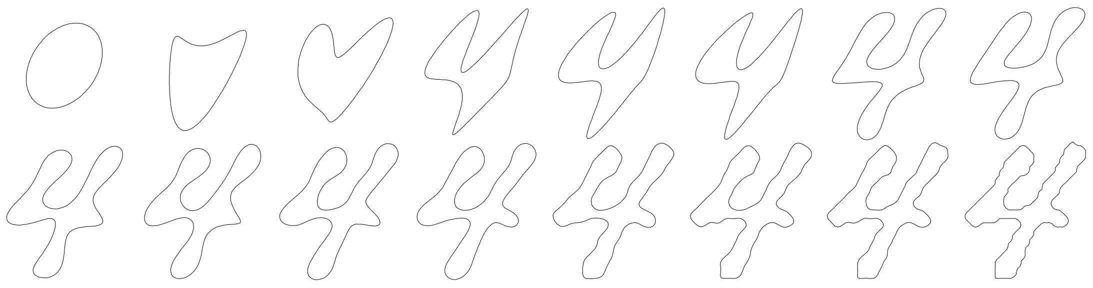


Figure 18: Character '4' reconstructed by elliptic Fourier descriptors of orders up to 1, 2, . . . 10; 15, 20, 30, 40, 50, and 100, respectively.

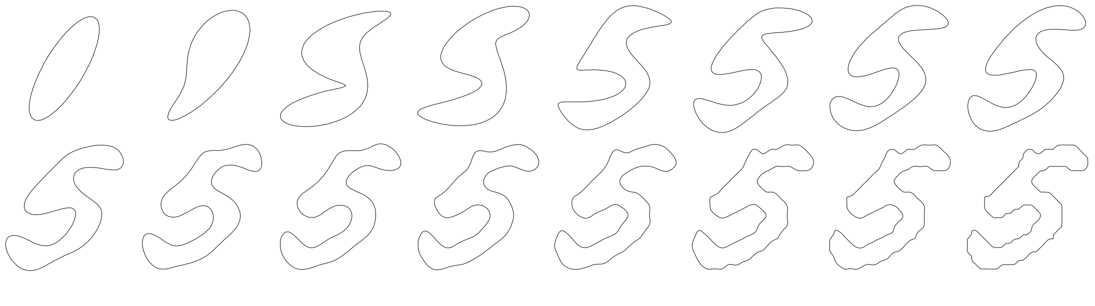


Figure 19: Character '5' reconstructed by elliptic Fourier descriptors of orders up to 1, 2, . . . 10; 15, 20, 30, 40, 50, and 100, respectively.

where T is the length of the boundary curve, consisting of m line segments, t_k is the accumulated length of the boundary from the starting point p_1 to the k-th point p_k, and $\Delta\varphi_k$ is the angle between the vectors $[p_{k-1}, p_k]$ and $[p_k, p_{k+1}]$. a_n and b_n are size- and translation-invariant. Rotation invariance can be obtained by transforming to polar coordinates. Then the amplitudes

$$A_n = \sqrt{a_n^2 + b_n^2} \tag{76}$$

are independent of rotation and mirroring, while the phase angles $\alpha_n = tan(a_n/b_n)$ are not. However, mirroring can be detected via the α_j's. It can be shown that

$$F_{kj} = j^*\alpha_k - k^*\alpha_j, \tag{77}$$

is independent of rotation, but dependent on mirroring. Here, $j^* = j/\text{gcd}(j,k)$, $k^* = k/\text{gcd}(j,k)$, and $\text{gcd}(j,k)$ is the greatest common divisor of j and k.

Zahn and Roskies warn that α_k becomes unreliable as $A_k \to 0$, and is totally undefined when $A_k = 0$. Therefore, the F_{kj} terms may be unreliable.

Granlund [65] uses a complex number $z(t) = x(t) + y(t)$ to denote the points on the contour. Then the contour can be expressed as a Fourier series:

$$z(t) = \sum_{n=-\infty}^{\infty} a_n e^{\frac{j2\pi nt}{T}}, \tag{78}$$

where

$$a_n = \frac{1}{T}\int_0^T z(t)e^{-\frac{j2\pi nt}{T}}dt \tag{79}$$

are the *complex* coefficients. a_0 is the center of gravity, and the other coefficients $a_n, n \neq 0$ are independent of translation. Again, T is the total contour length. The derived features

$$b_n = \frac{a_{1+n}a_{1-n}}{a_1^2}, \tag{80}$$

$$D_{mn} = \frac{a_{1+m}^{n/k}a_{1-n}^{m/k}}{a_1^{(m+n)/k}} \tag{81}$$

are independent of scale and rotation. Here, $n \neq 1$ and $k = \text{gcd}(m,n)$ is the greatest common divisor of m and n. Furthermore,

$$b_1^* = \frac{a_2|a_1|}{a_1^2}, \tag{82}$$

$$d_{m1}^* = \frac{a_{1+m}|a_1|^m}{a_1^{m+1}} \tag{83}$$

are scale-independent, but depend on rotation, so they can be useful when the orientation of the characters is known.

Persoon and Fu [3] pointed out that

$$a_n = a_{-n}e^{-\frac{2\pi n\alpha}{T}} \tag{84}$$

for some α. Therefore, the set of a_n's is redundant.

4.6 Evaluation Studies

Taxt et al. [62] evaluated Zahn and Roskies' Fourier descriptors [64], Kuhl and Giardina's elliptic Fourier descriptors [30], Lin and Hwang's elliptic Fourier descriptors [63], and their own cubic spline approximation [62]. For characters with known rotation, the best performance was reported using Kuhl and Giardina's method.

Persoon and Fu [3] observed that Zahn and Roskies' descriptors (a_n, b_n) converge slowly to zero as $n \to 0$ relative to Granlund's [65] descriptors (a_n) in the case of piecewise linear contour curves. This suggests that Zahn and Roskies' descriptors are not so well suited for the character contours obtained from binary raster objects nor character skeletons.

5 Features Extracted From the Vector Representation

Character skeletons (Fig. 5) are obtained by thinning the binary raster representation of the characters. An overwhelming number of thinning algorithms exist, and some recent evaluation studies give clues to their merits and disadvantages [15, 16, 66]. The task of choosing the right one often involves a compromise; one wants one-pixel wide 8-connected skeletons without spurious branches or displaced junctions, some kind of robustness to rotation and noise, and at the same time a fast and easy-to-implement algorithm. Kwok's thinning method [67] appears to be a good candidate, although its implementation is complicated.

A character graph can be derived from the skeleton by approximating it with a number of straight line segments and junction points. Arcs may be used for curved parts of the skeleton.

Wang and Pavlidis have recently proposed a method for obtaining character graphs directly from the gray level image [53, 68]. They view the gray level image as a 3D surface, with the gray levels mapped along the z-coordinate, using $z = 0$ for white (background) and, for example, $z = 255$ for black. By using topographic analysis, ridge lines and saddle points are identified, which are then used to obtain character graphs consisting of straight line segments, arcs, and junction points. The saddle points are analyzed to determine if they are points of unintentionally touching characters, or unintentionally broken characters. This method is useful when even the best available binarization methods are unable to preserve the character shape in the binarized image.

5.1 Template matching

Template matching in its pure form is not well suited for character skeletons, since the chances are small that the pixels of the branches in the input skeleton will exactly coincide with the pixels of the correct template skeleton. Lee and Park

[69] reviewed several non-linear shape normalization methods used to obtain uniform line or stroke spacing both vertically and horizontally. The idea is that such methods will compensate for shape distortions. Such normalizations are claimed to improve the performance for template matching [70], but may also be used as a preprocessing step for zoning.

5.2 Deformable Templates

Deformable templates have been used by Burr [71] and Wakahara [72, 73] for recognition of character skeletons. In Wakahara's approach, each template is deformed in a number of small steps, called *local affine transforms* (LAT) to match the candidate input pattern (Fig. 20). The number and types of transformations before a match is obtained can be used as a dissimilarity measure between each template and the input pattern.

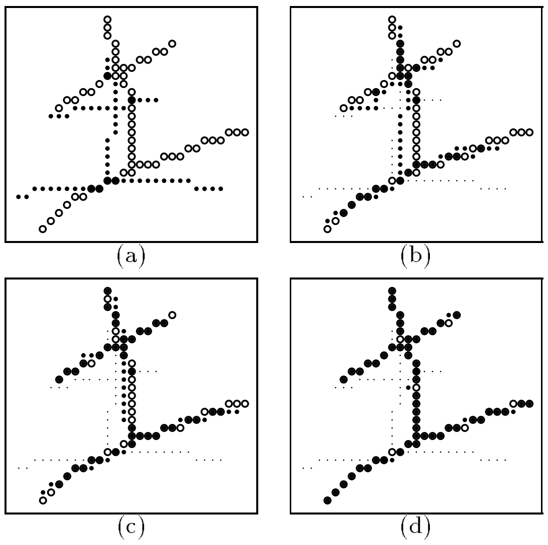


Figure 20: The deformable template matching approach of Wakahara [72]. Legend: '·'=original template pixels not in transformed template; '•'=transformed template; 'o'=input pattern; '●'=common pixels of transformed template and input pattern. **(a)** Template and input pattern of a Chinese character. **(b)**–**(d)** after 1, 5, and 10 iterations, respectively, of local affine transforms on a copy of the template.

5.3 Graph Description

Pavlidis [74] extracts approximate strokes from skeletons. Kahan et al. [75] augmented them with additional features to obtain reasonable recognition performance.

For Chinese character recognition, several authors extract graph descriptions or representations from skeletons as features [76, 77, 78]. Lu et al. [76] derive *hierarchical attributed graphs* to deal with variations in stroke lengths and connectedness due to variable writing style of different writers. Cheng et al. [79] use the Hough transform [80] on single-character images to extract stroke lines from skeletons of Chinese characters.

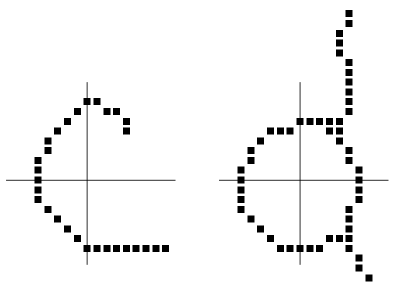


Figure 21: Thinned letters 'c' and 'd'. Vertical and horizontal axes are placed at the center of gravity. 'c' and 'd' both have one semicircle in the West direction, but none in the other directions. 'c' has one horizontal crossing and two vertical crossings. (Adopted from Kundu et al. [82].)

5.4 Discrete features

From thinned characters, the following features may be extracted [81, 82]: the number of loops; the number of T-joints; the number of X-joints; the number of bend points; width-to-height ratio of enclosing rectangle; presence of an isolated dot; total number of endpoints, and number of endpoints in each of the four directions N, S, W, and E; number of semi-circles in each of these four directions; and number of crossings with vertical and horizontal axes, respectively, the axes placed on the center of gravity. The last two features are explained in Fig. 21.

One might use crossings with many superimposed lines as features, and in fact, this was done in earlier OCR systems [1]. However, these features alone do not lead to robust recognition systems; as the number of superimposed lines is increased, the resulting features are less robust to variations in fonts (for machine printed characters) and variability in character shapes and writing styles (for handwritten characters).

5.5 Zoning

Holbæk-Hanssen et al. [83] measured the length of the character graph in each zone (Fig. 22). These features can be made size independent by dividing the graph length in each zone by the total length of the line segments in the graph. However, the features can not be made rotation independent. The presence or absence of junctions or endpoints in each zone can be used as additional features.

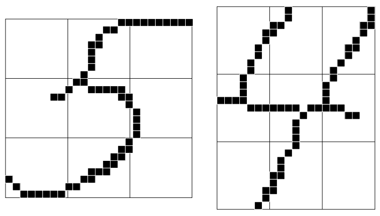


Figure 22: Zoning of character skeletons.

5.6 Fourier Descriptors

The Fourier descriptor methods described in Sections 4.4–4.5 for character contours may also be used for character skeletons or character graphs, since the skeleton or graph can be traversed to form a (degenerated) closed curve. Taxt and Bjerde [84] studied Kuhl and Giardina's elliptic Fourier descriptors [30], and stressed that for character graphs with two line endings, no junctions, and no loops, some of the descriptors will be zero, while for graphs with junctions or loops, all descriptors will be non-zero. For the size- and rotation-variant descriptors, Taxt and Bjerde stated that:

- For straight lines,
  $a_n^* = c_n^* = 0, n = 2, 4, 6, \ldots$
  $b_n^* = d_n^* = 0, n = 1, 2, 3, \ldots$
- For non-straight graphs with two line endings, no junctions, and no loops,
  $b_n^* = d_n^* = 0, n = 1, 2, 3, \ldots$
- For graphs with junctions or loops,
  $a_n^* \neq 0,\ b_n^* \neq 0,\ c_n^* \neq 0,\ d_n^* \neq 0,\ n = 1, 2, 3, \ldots$

The characteristics for rotation- and size-invariant features were also found [84]. Taxt and Bjerde observed that instances of the same character that happen to be different with respect to the above types will get very different feature vectors. The solution was to pre-classify the character graphs in to one of the three types, and then use a separate classifier for each type.

5.7 Evaluation Studies

Holbæk-Hanssen et al. [83] compared the zoning method described in Sec. 5.5 with Zahn and Roskies' Fourier descriptor method on character graphs. For characters with known orientation, the zoning method was better, while Zahn and Roskies' Fourier descriptors were better on characters with unknown rotation.

6 Neural Network Classifiers

Multilayer feedforward neural networks [85] have been used extensively in OCR, for example, by Le Cun et al. [86], Takahashi [60], and Cao et al. [25]. These networks may be viewed as a combined feature extractor and classifier. Le Cun et al. scale each input character to a 16×16 grid, which are then fed into 256 input nodes of the neural network (Fig. 23). The network has ten output nodes, one for each of the ten digit classes '0'–'9' that the network tries to recognize. Three intermediate layers were used. Each node in a layer has connections from a number of nodes in the previous layer, and during the training phase, connection weights are learned. The output at a node is a function (e.g., sigmoid) of the weighted sum of the connected nodes at the previous layer. One can think of a feedforward neural network as constructing decision boundaries in a feature space, and as the number of layers and nodes increases, the flexibility of the classifier increases by allowing more and more complex decision boundaries. However, it has been shown [85, 86] that this flexibility must be restricted to obtain good recognition performance. This is parallel to the *curse of dimensionality* effect observed in statistical classifiers, mentioned earlier.

Another viewpoint can also be taken. Le Cun et al.'s neural network can be regarded as performing hierarchical feature extraction. Each node "sees" a window in the previous layer and combines the low-level features in this window into a higher level feature. So, the higher the network layer, the more abstract and more global features are extracted, the final abstraction level being the features *digit '0'*, *digit '1'*, ..., *digit '9'*. Note that the feature extractors are not hand-crafted or ad-hoc selected rules, but are trained on a large set of training samples.

Some neural networks are given extracted features as input instead of a scaled or subsampled input image (e.g., [60, 25]). Then the network can be viewed as a pure classifier, constructing some complicated decision boundaries, or it can be viewed as extracting "superfeatures" in the combined process of feature extraction and classification.

One problem with using neural networks in OCR is that it is difficult to analyze and fully understand the decision making process [87]. What are the implicit features, and what are the decision boundaries? Also, an unbiased comparison between neural networks and statistical classifiers is difficult. If the pixels themselves are used as inputs to a neural network, and the neural network is compared with, say, a k-nearest neighbor (kNN) classifier using the same pixels as "features", then the comparison is not fair since a neural network has the opportunity to derive more meaningful features in the hidden layers. Rather, the best-performing statistical or structural classifiers have to be compared with the best neural network classifiers [88].

7 Discussion

Before selecting a specific feature extraction method, one needs to consider the total character recognition system in which it will operate. What kind of input characters is the system designed for? Is the input single-font typed or machine printed characters, multifont machine

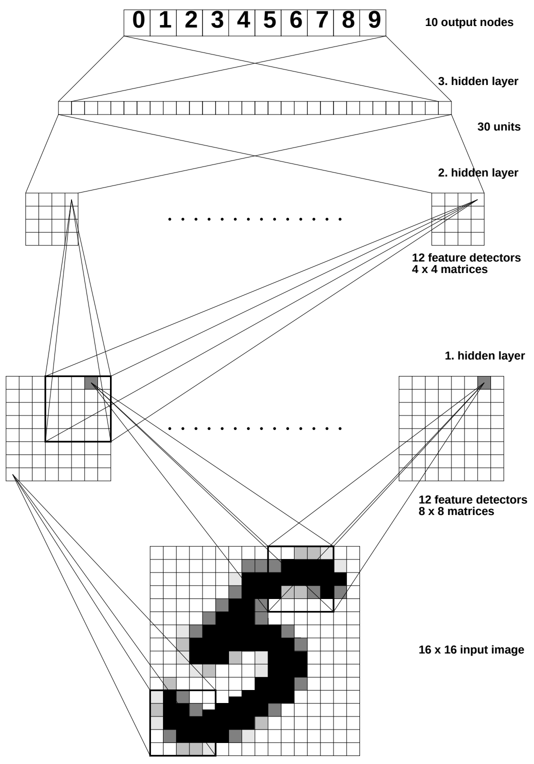


Figure 23: The neural network classifier used by Le Cun et al. [86].

printed, neatly hand-printed, or unconstrained handwritten? What is the variability of the characters belonging to the same class? Are gray-level images or binary images available? What is the scanner resolution? Is a statistical or structural classifier to be used? What are the throughput requirements (characters per second) as opposed to recognition requirements (reject vs. error rate)? What hardware is available? Can special-purpose hardware be used, or must the system run on standard hardware? What is the expected price of the system? Such questions need to be answered in order to make a qualified selection of the appropriate feature extraction method.

Often, a single feature extraction method alone is not sufficient to obtain good discrimination power. An obvious solution is to combine features from different feature extraction methods. If a statistical classifier is to be used, and a large training set is available, discriminant analysis can be used to select the features with highest discriminative power. The statistical properties of such combined feature vectors need to be explored. Another approach is to use multiple classifiers [20, 25, 26, 27, 89, 90]. In that case, one can even combine statistical, structural, and neural network classifiers to utilize their inherent differences.

The main disadvantage of using gray scale image based approaches is the memory requirements. Although Pavlidis [91] has shown that similar recognition rates may be achieved at a lower resolution for gray scale methods than for binary image based methods, gray scale images can not be compressed significantly without a loss of information. Binary images are easily compressed using, for example, run-length coding, and algorithms can be written to work on this format. However, as the performance and memory capacity of computers continue to double every 18 months or so, gray-scale methods will eventually become feasible in more and more applications.

To illustrate the process of identifying the best feature extraction methods, let us consider the digits in the hydrographic map (Fig. 1). The digits are hand-printed by one writer, and have roughly the same orientation, size, and slant (skew), although some variations exist, and they vary over the different portions of the whole map. These variations are probably large enough to affect the features considerably, if rotation-variant, size-variant, or skew-variant features are used. However, by using features invariant to scale, rotation, and skew, a larger variability is allowed, and confusion among characters such as '6' and '9' may be expected. By using a statistical classifier which assumes statistically dependent features (e.g., using the multivariate Gaussian distribution), we can hope that these variations will be properly accounted for. Ideally, it should then be possible to find the size, orientation, and perhaps slant directions in the feature space by principal component analysis (PCA), although the actual PCA does not have to be implemented. However, characters with unusual size, rotation, or skew will probably not be correctly classified. An appropriate solution may therefore be to use a mix of variant and invariant features.

For many applications, robustness to variability in character shape, to degradation, and to noise is important. Characters may be fragmented or merged. Other characters might be self-touching or have a broken loop. For features extracted from character contours or skeletons, we will expect very different features depending on whether fragmented, self-touching, or broken loop characters occur or not. Separate classes will normally have to be used for these variants, but the training set may contain too few of each variant to make reliable class descriptions.

Fourier descriptors cannot be applied to fragmented characters in a meaningful way since this method extracts features from one single closed contour or skeleton. Further, outer contour curve based methods do not use information about the interior of the characters, like holes in '8', '0', etc., so one then has to consider if some classes will be easily confused. A solution may be to use multistage classifiers [25].

Zoning, moment invariants, Zernike moments, and the Karhunen-Loeve transform may be good alternatives, since they are not affected by the above degradations to the same extent. Zoning is probably not a good choice, since the variations present in each digit class may cause a specific part of a character to fall into different zones for different instances. Cao et al. [25] tried to compensate for this by using *fuzzy borders*, but this method is only capable of compensating for small variations of the character shape. Moment invariants are invariant to size and rotation, and some moment invariants are also invariant to skew and mirror images [42]. Mirror image invariance is not desirable, so moment invariants that are invariant to skew but not mirror images would be useful, and a few such invariants do exist [42]. Moment invariants lack the reconstructability property, which probably means that a few more features are needed than for features for which reconstruction is possible.

Zernike moments are complex numbers which themselves are not rotation invariant, but their amplitudes are. Also, size invariance is obtained by prescaling the image. In other words, we can obtain size- and rotation-dependent features. Since Zernike moments have the reconstructability property, they appear to be very promising for our application.

Of the unitary image transforms, the Karhunen-Loeve transform has the best information compactness in terms of mean square error. However, since the features are only linear combinations of the pixels in the input character image, we can not expect them to be able to extract high-level features the way other methods do, so many more features are needed, and thus a much larger training set than for other methods. Also, since the features are tied to pixel locations, we can not expect to get class descriptions suitable for parametric statistical classifiers. Still, a non-parametric classifier like the k-nearest neighbor classifier [9] may perform well on the Karhunen-Loeve transform features.

Discretization errors and other high frequency noise are removed when using Fourier descriptors (Figs. 18–19), moment invariants, or Zernike moments, since we never use very high order terms. Zoning methods are also robust against high frequency noise because of the implicit low pass filtering in the method.

From the above analysis, it seems like Zernike moments would be good features in our hydrographic map application. However, one really needs to perform an experimental evaluation of a few of the most promising methods to decide which feature extraction method is the best in practice for each application. The evaluation should be performed on large data sets that are representative for the particular application. Large, standard data sets are now available from NIST [56] (Gaithersburg, MD 20899, USA) and SUNY at Buffalo [92] (CEDAR, SUNY, Buffalo, NY 14260, USA). If these or other available data sets are not representative, then one might have to collect a large data set. However, performance on the standard data sets does give an indication of the usefulness of the features, and provides performance figures that can be compared with other research groups' results.

8 Summary

Optical character recognition (OCR) is one of the most successful applications of automatic pattern recognition. Since the mid 1950's, OCR has been a very active field for research and development [1]. Today, reasonably good OCR packages can be bought for as little as \$100. However, these are only able to recognize high quality printed text documents or neatly written hand-printed text. The current research in OCR is now addressing documents that are not well handled by the available systems, including severely degraded, omnifont machine printed text, and (unconstrained) handwritten text. Also, efforts are being made to achieve lower substitution error rates and reject rates even on good quality machine printed text, since an experienced human typist still has a much lower error rate, albeit at a slower speed.

Selection of feature extraction method is probably the single most important factor in achieving high recognition performance. Given the large number of feature extraction methods reported in the literature, a newcomer to the field is faced with the following question: Which feature extraction method is the best for a given application? This question led us to characterize the available feature extraction methods, so that the most promising methods could be sorted out. An experimental evaluation of these few promising methods must still be performed to select the best method for a specific application.

Devijver and Kittler define feature extraction (page 12 in [11]) as the problem of "extracting from the raw data the information which is most relevant for classification purposes, in the sense of minimizing the within-class pattern variability while enhancing the between-class pattern variability." In this paper, we reviewed feature extraction methods including:

1. Template matching,
2. Deformable templates,
3. Unitary image transforms,
4. Graph description,
5. Projection histograms,
6. Contour profiles,
7. Zoning,
8. Geometric moment invariants,
9. Zernike moments,
10. Spline curve approximation,
11. Fourier descriptors.

Each of these methods may be applied to one or more of the following representation forms

1. Gray level character image.
2. Binary character image,
3. Character contour,
4. Character skeleton or character graph.

For each feature extraction method and each character representation form, we discussed the properties of the extracted features.

Before selecting a specific feature extraction method, one needs to consider the total character recognition system in which it will operate. The process of identifying the best feature extraction method was illustrated by considering the digits in the hydrographic map (Fig. 1) as an example. It appears that Zernike moments would be good features in this application. However, one really needs to perform an experimental evaluation of a few of the most promising methods to decide which feature extraction method is the best in practice for each application. The evaluation should be performed on large data sets that are representative for the particular application.

Acknowledgements

We thank the anonymous referee for useful comments. This work was supported by a NATO collaborative research grant (CRG 930289), and The Research Council of Norway.

References

[1] S. Mori, C. Y. Suen, and K. Yamamoto, "Historical review of OCR research and developement," *Proceedings of the IEEE*, vol. 80, pp. 1029–1058, July 1992.

[2] J. W. Gorman, O. R. Mitchell, and F. P. Kuhl, "Partial shape recognition using dynamic programming," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 10, pp. 257–266, Mar. 1988.

[3] E. Persoon and K.-S. Fu, "Shape discrimination using Fourier descriptors," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 7, pp. 170–179, Mar. 1977.

[4] L. Shen, R. M. Rangayyan, and J. E. L. Desaultes, "Application of shape analysis to mammographic calcifications," *IEEE Trans. Medical Imaging*, vol. 13, pp. 263–274, June 1994.

[5] D. G. Elliman and I. T. Lancaster, "A review of segmentation and contextual analysis for text recognition," *Pattern Recognition*, vol. 23, no. 3/4, pp. 337–346, 1990.

[6] C. E. Dunn and P. S. P. Wang, "Character segmentation techniques for handwritten text — a survey," in *Proceedings of 11th IAPR Int. Conf. Pattern Recognition*, vol. II, (The Hague, The Netherlands), pp. 577–580, IEEE Computer Society Press, 1992.

[7] L. Lam, S.-W. Lee, and C. Y. Suen, "Thinning methodologies — a comprehensive survey," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 14, pp. 869–885, Sept. 1992.

[8] K.-S. Fu, *Syntactic Pattern Recognition and Application.* Englewood Cliffs, New Jersey: Prentice-Hall, 1982.

[9] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis.* New York: John Wiley & Sons, 1973.

[10] K. Fukunaga, *Introduction to Statistical Pattern Recognition.* Boston: Academic Press, 1990.

[11] P. A. Devijver and J. Kittler, *Pattern Recognition: A Statistical Approach.* London: Prentice-Hall, 1982.

[12] J. R. Ullmann, *Pattern Recognition Techniques.* London, England: Butterworth, 1973.

[13] Ø. D. Trier and T. Taxt, "Evaluation of binarization methods for document images," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 17, pp. 312–315, Mar. 1995.

[14] Ø. D. Trier and A. K. Jain, "Goal-directed evaluation of binarization methods," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 17, pp. 1191–1201, Dec. 1995.

[15] S.-W. Lee, L. Lam, and C. Y. Suen, "Performance evaluation of skeletonizing algorithms for document image processing," in *Proceedings of the First International Conference on Document Analysis and Recognition*, (Saint-Malo, France), pp. 260–271, IEEE Computer Society Press, 1991.

[16] M. Y. Jaisimha, R. M. Haralick, and D. Dori, "Quantitative performance evaluation of thinning algorithms under noisy conditions," in *Proceedings of IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, (Seattle, WA), pp. 678–683, June 1994.

[17] V. K. Govindan and A. P. Shivaprasad, "Character recognition — a review," *Pattern Recognition*, vol. 23, no. 7, pp. 671–683, 1990.

[18] G. Nagy, "At the frontiers of OCR," *Proceedings of the IEEE*, vol. 80, pp. 1093–1100, July 1992.

[19] C. Y. Suen, R. Legault, C. Nadal, M. Cheriet, and L. Lam, "Building a new generation of handwriting recognition systems," *Pattern Recognition Letters*, vol. 14, pp. 303–315, Apr. 1993.

[20] C. Y. Suen, C. Nadal, R. Legault, T. A. Mai, and L. Lam, "Computer recognition of unconstrained handwritten numerals," *Proceedings of the IEEE*, vol. 80, pp. 1162–1180, July 1992.

[21] C. Y. Suen, M. Berthod, and S. Mori, "Automatic recognition of handprinted characters — the state of the art," *Proceedings of the IEEE*, vol. 68, pp. 469–487, Apr. 1980.

[22] J. Mantas, "An overview of character recognition methodologies," *Pattern Recognition*, vol. 19, no. 6, pp. 425–430, 1986.

[23] A. K. Jain and B. Chandrasekaran, "Dimensionality and sample size considerations in pattern recognition practice," in *Classification, Pattern Recognition, and Reduction of*

Dimensionality (P. R. Krishnaiah and L. N. Kanal, eds.), vol. 2 of *Handbook of Statistics*, pp. 835–855, Amsterdam: North-Holland, 1982.

[24] P. Gader, B. Forester, M. Ganzberger, A. Gillies, B. Mitchell, M. Whalen, and T. Yocum, "Recognition of handwritten digits using template and model matching," *Pattern Recognition*, vol. 24, no. 5, pp. 421–431, 1991.

[25] J. Cao, M. Ahmadi, and M. Shridhar, "Handwritten numeral recognition with multiple features and multistage classifiers," in *IEEE International Symposium on Circuits and Systems*, vol. 6, (London), pp. 323–326, May 30–June 2 1994.

[26] L. Xu, A. Krzyżak, and C. Y. Suen, "Methods of combining multiple classifiers and their applications to handwriting recognition," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 22, no. 3, pp. 418–435, 1992.

[27] F. Kimura and M. Shridhar, "Handwritten numerical recognition based on multiple algorithms," *Pattern Recognition*, vol. 24, no. 10, pp. 969–983, 1991.

[28] T. Y. Zhang and C. Y. Suen, "A fast parallel algorithm for thinning digital patterns," *Communications of the ACM*, vol. 27, pp. 236–239, Mar. 1984.

[29] T. H. Reiss, "The revised fundamental theorem of moment invariants," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 13, pp. 830–834, Aug. 1991.

[30] F. P. Kuhl and C. R. Giardina, "Elliptic Fourier features of a closed contour," *Computer Vision, Graphics and Image Processing*, vol. 18, pp. 236–258, 1982.

[31] A. Khotanzad and Y. H. Hong, "Invariant image recognition by zernike moments," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 12, no. 5, pp. 489–497, 1990.

[32] D. H. Ballard and C. M. Brown, *Computer Vision*, pp. 65–70. Englewood Cliffs, New Jersey: Prentice-Hall, 1982.

[33] W. K. Pratt, *Digital Image Processing.* New York: John Wiley & Sons, second ed., 1991.

[34] G. Storvik, "A bayesian approach to dynamic contours through stochastic sampling and simulated annealing," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 16, pp. 976–986, Oct. 1994.

[35] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *International Journal of Computer Vision*, vol. 5, pp. 321–331, 1988.

[36] A. D. Bimbo, S. Santini, and J. Sanz, "OCR from poor quality images by deformation of elastic templates," in *Proceedings of 12th IAPR Int. Conf. Pattern Recognition*, vol. 2, (Jerusalem, Israel), pp. 433–435, 1994.

[37] H. C. Andrews, "Multidimensional rotations in feature selection," *IEEE Trans. Computers*, vol. 20, pp. 1045–1051, Sept. 1971.

[38] R. C. Gonzalez and R. E. Woods, *Digital Image Processing.* Addison-Wesley, 1992.

[39] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of Cognitive Neuroscience*, vol. 3, no. 1, pp. 71–86, 1991.

[40] M. Bokser, "Omnidocument technologies," *Proceedings of the IEEE*, vol. 80, pp. 1066–1078, July 1992.

[41] M.-K. Hu, "Visual pattern recognition by moment invariants," *IRE Trans. Information Theory*, vol. IT-8, pp. 179–187, Feb. 1962.

[42] T. H. Reiss, *Recognizing Planar Objects Using Invariant Image Features*, vol. 676 of *Lecture Notes in Computer Science.* Springer-Verlag, 1993.

[43] S. O. Belkasim, M. Shridhar, and A. Ahmadi, "Pattern recognition with moment invariants: A comparative study and new results," *Pattern Recognition*, vol. 24, pp. 1117–1138, Dec. 1991.

[44] S. O. Belkasim, M. Shridhar, and A. Ahmadi, "Corrigendum," *Pattern Recognition*, vol. 26, p. 377, Jan. 1993.

[45] Y. Li, "Reforming the theory of invariant moments for pattern recognition," *Pattern Recognition Letters*, vol. 25, pp. 723–730, July 1992.

[46] J. Flusser and T. Suk, "Pattern recognition by affine moment invariants," *Pattern Recognition*, vol. 26, pp. 167–174, Jan. 1993.

[47] J. Flusser and T. Suk, "Affine moment invariants: A new tool for character recognition," *Pattern Recognition Letters*, vol. 15, pp. 433–436, Apr. 1994.

[48] B. Bamieh and R. J. P. de Figueiredo, "A general moment-invariants/attributed-graph method for three-dimensional object recognition from a single image," *IEEE J. Robotics and Automation*, vol. 2, pp. 31–41, Mar. 1986.

[49] A. Khotanzad and Y. H. Hong, "Rotation invariant image recognition using features selected via a systematic method," *Pattern Recognition*, vol. 23, no. 10, pp. 1089–1101, 1990.

[50] J. M. Westall and M. S. Narasimha, "Vertex directed segmentation of handwritten numerals," *Pattern Recognition*, vol. 26, pp. 1473–1486, Oct. 1993.

[51] H. Fujisawa, Y. Nakano, and K. Kurino, "Segmentation methods for character recognition: from segmentation to document structure analysis," *Proceedings of the IEEE*, vol. 80, pp. 1079–1092, July 1992.

[52] Ø. D. Trier, T. Taxt, and A. K. Jain, "Data capture from maps based on gray scale topographic analysis," in *Proceedings of the Third*

International Conference on Document Analysis and Recognition, (Montreal, Canada), pp. 923–926, Aug. 1995.

[53] L. Wang and T. Pavlidis, "Direct gray-scale extraction of features for character recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 15, pp. 1053–1067, Oct. 1993.

[54] R. M. Haralick, L. T. Watson, and T. J. Laffey, "The topographic primal sketch," *J. Robotics Res.*, vol. 2, pp. 50–72, Spring 1983.

[55] J. D. Tubbs, "A note on binary template matching," *Pattern Recognition*, vol. 22, no. 4, pp. 359–365, 1989.

[56] M. D. Garris, J. L. Blue, G. T. Candela, D. L. Dimmick, J. Gelst, P. J. Grother, S. A. Janet, and C. L. Wilson, "NIST form-based handprint recognition system," Tech. Rep. NISTIR 5469, National Institute of Standards and Technology, Gaithersburg, MD 20899, USA, July 1994.

[57] M. H. Glauberman, "Character recognition for business machines," *Electronics*, vol. 29, pp. 132–136, Feb. 1956.

[58] R. Kasturi and M. M. Trivedi, eds., *Image Analysis Applications*. New York: Marcel Dekker, 1990.

[59] L. Yang and F. Albregtsen, "Fast computation of invariant geometric moments: A new method giving correct results," in *Proceedings of 12th IAPR Int. Conf. Pattern Recognition*, vol. 1, (Jerusalem, Israel), pp. 201–204, Oct. 1994.

[60] H. Takahashi, "A neural net OCR using geometrical and zonal pattern features," in *Proceedings of the First International Conference on Document Analysis and Recognition*, (Saint-Malo, France), pp. 821–828, 1991.

[61] I. Sekita, K. Toraichi, R. Mori, K. Yamamoto, and H. Yamada, "Feature extraction of handwritten Japanese characters by spline functions for relaxation matching," *Pattern Recognition*, no. 1, pp. 9–17, 1988.

[62] T. Taxt, J. B. Ólafsdóttir, and M. Dæhlen, "Recognition of handwritten symbols," *Pattern Recognition*, vol. 23, no. 11, pp. 1155–1166, 1990.

[63] C.-S. Lin and C.-L. Hwang, "New forms of shape invariants from elliptic Fourier descriptors," *Pattern Recognition*, vol. 20, no. 5, pp. 535–545, 1987.

[64] C. T. Zahn and R. C. Roskies, "Fourier descriptors for plane closed curves," *IEEE Trans. Computers*, vol. 21, pp. 269–281, Mar. 1972.

[65] G. H. Granlund, "Fourier preprocessing for hand print character recognition," *IEEE Trans. Computers*, vol. 21, pp. 195–201, Feb. 1972.

[66] B. K. Jang and R. T. Chin, "One-pass parallel thinning: Analysis, properties and quantitative evaluation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 14, pp. 1129–1140, Nov. 1992.

[67] P. C. K. Kwok, "A thinning algorithm by contour generation," *Communications of the ACM*, vol. 31, no. 11, pp. 1314–1324, 1988.

[68] L. Wang and T. Pavlidis, "Detection of curved and straight segments from gray scale topography," *CVGIP: Image Understanding*, vol. 58, pp. 352–365, Nov. 1993.

[69] S.-W. Lee and J.-S. Park, "Nonlinear shape normalization methods for the recognition of large-set handwritten characters," *Pattern Recognition*, vol. 27, no. 7, pp. 895–902, 1994.

[70] H. Yamada, K. Yamamoto, and T. Saito, "A nonlinear normalization method for handprinted kanji character recognition — line density equalization," *Pattern Recognition*, vol. 23, no. 9, pp. 1023–1029, 1990.

[71] D. J. Burr, "Elastic matching of line drawings," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 3, pp. 708–713, Nov. 1981.

[72] T. Wakahara, "Toward robust handwritten character recognition," *Pattern Recognition Letters*, vol. 14, pp. 345–354, Apr. 1993.

[73] T. Wakahara, "Shape matching using LAT and its application to handwritten numeral recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 16, pp. 618–629, June 1994.

[74] T. Pavlidis, "A vectorizer and feature extractor for document recognition," *Computer Vision, Graphics and Image Processing*, vol. 35, pp. 111–127, 1986.

[75] S. Kahan, T. Pavlidis, and H. S. Baird, "On the recoginition of printed characters of any font and size," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 9, pp. 274–288, Mar. 1987.

[76] S. W. Lu, Y. Ren, and C. Y. Suen, "Hierarchical attributed graph representation and recognition of handwritten chinese characters," *Pattern Recognition*, vol. 24, no. 7, pp. 617–632, 1991.

[77] H.-J. Lee and B. Chen, "Recognition of handwritten Chinese characters via short line segments," *Pattern Recognition*, vol. 25, no. 5, pp. 543–552, 1992.

[78] F.-H. Cheng, W.-H. Hsu, and M.-C. Kuo, "Recognition of handprinted Chinese characters via stroke relaxation," *Pattern Recognition*, vol. 26, no. 4, pp. 579–593, 1993.

[79] F.-H. Cheng, W.-H. Hsu, and M.-Y. Chen, "Recognition of handwritten Chinese characters by modified Hough transform techniques," *IEEE Trans. Pattern Analysis and*

Machine Intelligence, vol. 11, pp. 429–439, Apr. 1989.

[80] R. O. Duda and P. E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," *Communications of the ACM*, vol. 15, pp. 11–15, Jan. 1972.

[81] S. R. Ramesh, "A generalized character recognition algorithm: A graphical approach," *Pattern Recognition*, vol. 22, no. 4, pp. 347–350, 1989.

[82] A. Kundu, Y. He, and P. Bahl, "Recognition of handwritten word: First and second order hidden Markov model based approach," *Pattern Recognition*, vol. 22, no. 3, pp. 283–297, 1989.

[83] E. Holbæk-Hanssen, K. Bråthen, and T. Taxt, "A general software system for supervised statistical classification of symbols," in *Proceedings of the 8th International Conference on Pattern Recognition*, (Paris, France), pp. 144–149, Oct. 1986.

[84] T. Taxt and K. W. Bjerde, "Classification of handwritten vector symbols using elliptic Fourier descriptors," in *Proceedings of 12th IAPR Int. Conf. Pattern Recognition*, vol. 2, (Jerusalem, Israel), pp. 123–128, Oct. 1994.

[85] J. Hertz, A. Krogh, and R. G. Palmer, *Introduction to the Theory of Neural Computation*. Redwood City, CA: Addison-Wesley, 1991.

[86] Y. L. Cun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubband, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, pp. 541–551, 1989.

[87] R. P. W. Duin, "Superlearning and neural network magic," *Pattern Recognition Letters*, vol. 15, pp. 215–217, Mar. 1994.

[88] A. K. Jain and J. Mao, "Neural networks and pattern recognition," in *Proceedings of the IEEE World Congress on Computational Intelligence*, (Orlando, FL), June 1994.

[89] T. K. Ho, J. J. Hull, and S. N. Srihari, "Decision combination in multiple classifier system," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 16, pp. 66–75, Jan. 1994.

[90] M. Sabourin, A. Mitiche, D. Thomas, and G. Nagy, "Classifier combination for handprinted digit recognition," in *Proceedings of the Second International Conference on Document Analysis and Recognition*, (Tsukuba Science City, Japan), pp. 163–166, IEEE Computer Society Press, Oct. 1993.

[91] T. Pavlidis, "Recognition of printed text under realistic conditions," *Pattern Recognition Letters*, vol. 14, pp. 317–326, Apr. 1993.

[92] J. J. Hull, "A database for handwritten text research," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 16, pp. 550–554, May 1994.

About the author— ØIVIND DUE TRIER was born in Oslo, Norway, in 1966. He received his M.Sc. degree in Computer Science from The Norwegian Institute of Technology in 1991, and was with the Norwegian Defense Research Establishment from 1991 to 1992. Since 1992, he has been a Ph.D. student at the Department of Informatics, University of Oslo. His current research interests include pattern recognition, image analysis, document processing, and geographic information systems.

About the author— ANIL K. JAIN received a B.Tech. degree in 1969 from the Indian Institute of Technology, Kanpur, and the M.S. and Ph.D. degrees in Electrical Engineering from the Ohio State University, in 1970 and 1973, respectively. He joined the faculty of Michigan State University in 1974, where he currently holds the rank of University Distinguished Professor in the Department of Computer Science. Dr. Jain served as Program Director of the Intelligent Systems Program at the National Science Foundation (1980-81), and has held visiting appointments at Delft Technical University, Holland, Norwegian Computing Center, Oslo, and Tata Research Development and Design Center, Pune, India. He has also been a consultant to several industrial, government and international organizations. His current research interests are computer vision, image processing, and pattern recognition.

Dr. Jain has made significant contributions and published a large number of papers on the following topics: statistical pattern recognition, exploratory pattern analysis, Markov random fields, texture analysis, interpretation of range images, and 3D object recognition. Several of his papers have been reprinted in edited volumes on image processing and pattern recognition. He received the best paper awards in 1987 and 1991, and received certificates for outstanding contributions in 1976, 1979 and 1992 from the Pattern Recognition Society. Dr. Jain was the Editor-in-Chief of the *IEEE Transactions on Pattern Analysis and Machine Intelligence* (1991-1994), and is on the editorial boards of *Pattern Recognition*, *Pattern Recognition Letters*, *Journal of Mathematical Imaging*, *Journal of Applied Intelligence*, and *IEEE Transactions on Neural Networks*. He is the co-author of Algorithms for Clustering Data, Prentice-Hall, 1988, has edited the book Real-Time Object Measurement and Classification, Springer-Verlag, 1988, and has co-edited the books, Analysis and Interpretation of Range Images, Springer-Verlag, 1989, Neural Networks and Statistical Pattern Recognition, North-Holland, 1991, Markov Random Fields: Theory and Applications, Academic Press, 1993, and 3D Object Recognition, Elsevier, 1993.

Dr. Jain is a Fellow of the IEEE. He was the Co-General Chairman of the 11th International Conference on Pattern Recognition, Hague (1992), General Chairman of the IEEE Workshop on Interpretation of 3D Scenes, Austin (1989), Director of the NATO Advanced Research Workshop on Real-time Object Measurement and Classification, Maratea (1987), and co-directed NSF supported Workshops on "Future Reserach Directions in Computer Vision", Maui (1991), "Theory and Applications of Markov Random Fields", San Diego (1989) and "Range Image Understanding", East Lansing (1988). Dr. Jain was a member of the IEEE Publications Board (1988-90) and served as the Distinguished Visitor of the IEEE Computer Society (1988-90).

About the author— TORFINN TAXT was born in 1950 and is Professor for the Medical Image Analysis Section at the University of Bergen and Professor in image processing at the University of Oslo. He is an associate editor of the *IEEE Transactions on Medical Imaging* and of *Pattern Recognition*, and has a Ph.D. in developmental neuroscience (1983), Medical Degree (1976), and M.S. in computer science (1978) from the University of Oslo. His current research interests are in the areas of image restoration, image segmentation, multispectral analysis, medical image analysis, and document processing. He has published papers on the following topics: restoration of medical ultrasound and magnetic resonance images, magnetic resonance and remote sensing multispectral analysis, quantum field models for image analysis, and document processing.