AN EFFICIENT SCHEME FOR INVARIANT OPTICAL CHARACTER RECOGNITION USING TRIPLE CORRELATIONS

Yannis Avrithis, Anastassios Delopoulos and Stefanos Kollias

Computer Science Division, National Technical University of Athens,
Zografou 15773, Athens, Greece.

Abstract

The implementation of an efficient scheme for translation, rotation and scale invariant optical character recognition is presented in this paper. An image representation is used, which is based on appropriate clustering and transformation of the image triple-correlation domain. This representation is one-to-one related to the class of all shifted-rotated-scaled versions of the original image, as well as robust to a wide variety of additive noises. Special attention is given to binary images, which are used for Optical Character Recognition, and simulation results illustrate the performance of the proposed implementation.

1. Introduction

Most practical Optical Character Recognition systems involve many different tasks which require either integrated treatment of entire documents, or treatment of isolated words or characters. A complete text reading system includes the following major tasks: *analysis* of the document into its constituents, such as photographs, graphics and text; *segmentation* of the text into columns, paragraphs, lines, words and characters; *recognition* of the segmented characters; *ambiguity resolution* which might involve returning back to previous stages of the segmentation / recognition procedure. Other tasks also include *preprocessing* of the input image (gray scale normalization, noise elimination), *postprocessing* of the derived text (spelling verification or correction, sometimes incorporating customized lexicons), as well as the unavoidable interaction with human operators.

The character recognition task is divided into two phases: In the first, *feature extraction,* all unnecessary or undesired attributes are filtered out and the image in each segment is described as a vector of fixed length, containing all the "essential" characteristics of the character. The second phase is *classification.* A classifier, which learns to discriminate classes by generalizing from a training set, outputs the character label it believes is represented by the feature vector, or, if it is unsure, it outputs a set of choices and associated confidences. In this paper we deal only with the image recognition task and more specifically with feature extraction. In particular, an image transformation computed in terms of the third-order correlation of the pattern is used, and the obtained representation constitutes a feature vector which is insensitive to translation, rotation and scale transformations of the original image. We also achieve insensitivity to noise and to small shape distortions when moving from the pattern space to the feature space.

The transformation that projects the image space to our feature space is introduced in Section 2. Section 3 deals with practical considerations regarding the obtained representation, such as discretizations and computational complexity reduction. An optimized algorithm for the special case of binary images is also presented. Finally, Section 4 includes simulation results, illustrating the aforementioned properties of our representation and the performance of the algorithm with real input images.

2. The Invariant Representation

The representation of 2-D images that we used is described in this Section, having the following properties: [P1] shift-rotation-scale invariance (SRS), [P2] unique correspondence between the class of original images that are mutually related with rotation- translation-scaling transformation and the new representation domain, and [P3] noise insensitivity.

This representation is expressed in terms of the third-order correlation of the input image, which possesses some very important properties, especially regarding noise suppression [1]. The use of triple correlation for SRS invariant recognition is also proposed in [2], [3] and [4], while the use of third-order neural networks has also been examined in [3] and [4]. What we novelly propose here is a new efficient scheme for reducing the high computational complexity involved in extracting the desired features of the image. Besides efficient implementation of the recognition procedure using, for example, an artificial neural network, this method also allows real-time processing in the special case of binary (black & white) images which are most often used in OCR systems.

Definition and properties of 3^{rd} order correlations : Let $x(\mathbf{t})$ be a real 2-D signal with support $S = [0 \cdots N-1] \times [0 \cdots N-1]$. Its triple-correlation is defined as,

$$x_3(\tau_1, \tau_2) \triangleq \frac{1}{N^2} \sum_S x(\mathbf{t})\; x(\mathbf{t}+\tau_1)\; x(\mathbf{t}+\tau_2)\;, \tag{1}$$

The triple correlation of a 2-D signal $x(\mathbf{t})$ is a function of two 2-D vector indices, τ_1, τ_2, each of them spanning the set $S' = [(-N+1) \cdots (N-1)] \times [(-N+1) \cdots (N-1)]$. The triple correlation has the following *symmetries* :

$$x_3(\tau_1, \tau_2) = x_3(\tau_2, \tau_1) = x_3(\tau_1 - \tau_2, -\tau_2) = x_3(\tau_2 - \tau_1, -\tau_1) \tag{2}$$

It is also well known [1] that triple correlation is *insensitive* to additive Gaussian or any other linear and symmetrically distributed noise, and that there is *one-to-one correspondence* with the original signal. This property implies that we can safely compare two signals by only comparing their triple correlations. Let us now consider the image $y(\mathbf{t}) = x(\mathbf{T}_{\alpha,\theta}\,\mathbf{t} + \mathbf{t}_0)$, where $\mathbf{T}_{\alpha,\theta}$ is a scaling and rotation matrix and $\mathbf{t}_0$ a shifting vector; it can be easily checked out that, ignoring boundary points of S,

$$y_3(\tau_1, \tau_2) = x_3(\mathbf{T}_{\alpha,\theta}\,\tau_1, \mathbf{T}_{\alpha,\theta}\,\tau_2)\;, \tag{3}$$

i.e., when the signal plane shifts, the triple correlation is unaffected and when the signal plane rotates and/or is rescaled by $\mathbf{T}$, the same happens in the triple correlation domain for both lag indices τ_1, τ_2.

The proposed representation : By definition, $x_3(\tau_1, \tau_2)$ is the accumulation of all triple products formed by the values of $x(\mathbf{t})$ that lie on the corners of those *equal* triangles that are *shifts* of a prototype triangle defined by arbitrary vectors τ_1, τ_2. Hereafter we shall call $W(\tau_1, \tau_2)$ the set of all these triangles. Define, next, the set $K(\tau_1, \tau_2)$ of all triangles that are *similar* to the members of $W(\tau_1, \tau_2)$. For any set $K(\tau_1, \tau_2)$, we define a corresponding *class* $C(\tau_1, \tau_2)$ as the set of all triple-correlation lags whose indices form, on the $\mathbf{R}^2$ plane, triangles similar to the triangle defined by the vectors τ_1, τ_2. Note that if we let τ_1, τ_2 span the entire S', identical classes will be generated for different indices (τ_1, τ_2), (τ_1', τ_2'), if these indices form similar triangles. This redundancy can be removed [3] by fixing τ_1 to a constant vector and varying τ_2 in a subset of S'.

It can be verified [3] that any rotation θ and/or scaling α of the original 2-D plane, $x(\mathbf{t})$, results in an internal rearrangement of the elements of $C(\tau_1, \tau_2)$ without any inter-class interference, since it translates the specific $W(\tau_1, \tau_2)$ subset to another subset in $K(\tau_1, \tau_2)$.

We next define the following arrangement between the members of each class:

$$\tilde{x}_3(\rho, \phi; \tau_1, \tau_2) \triangleq x_3(\mathbf{T}_{\beta,\phi}\,\tau_1, \mathbf{T}_{\beta,\phi}\,\tau_2)\;, \tag{4}$$

where, $\mathbf{T}_{\beta,\phi}$ is defined similarly to $\mathbf{T}_{\alpha,\theta}$ and $\rho = \log\beta$. Variables ρ and ϕ are introduced to represent any scaled (in log form) and rotated triangle $W(\mathbf{T}_{\beta,\phi}\,\tau_1, \mathbf{T}_{\beta,\phi}\,\tau_2)$ when compared to a prototype triangle of class $C(\tau_1, \tau_2)$.

For the image $y(\mathbf{t}) = x(\mathbf{T}_{\alpha,\theta}\,\mathbf{t} + \mathbf{t}_0)$ it is easy to derive that

$$\tilde{y}_3(\rho, \phi; \tau_1, \tau_2) = \tilde{x}_3(\rho + \log\alpha, \phi + \theta; \tau_1, \tau_2)\,. \tag{5}$$

Conversely, if Eq. (5) holds for all classes $C(\tau_1, \tau_2)$ with the *same values* of α and θ, then, $y(\mathbf{t})$ can be generated from $x(\mathbf{t})$ by rotation (θ), rescaling (α) and any arbitrary translation. The above establishes the equivalence of the rotation and/or scaling of the original 2-D signal with a 2-D shift in the $\tilde{x}_3(\rho, \phi; \tau_1, \tau_2)$ domain with respect to ρ and ϕ.

Based on this conclusion, any transformation of the classes $C(\tau_1, \tau_2)$ that is shift invariant with respect to ρ, ϕ will provide a shift-rotation-scale invariant representation. It is well known that the 2-D Fourier transform $\tilde{X}_3(P, \Phi; \tau_1, \tau_2)$ of the field $\tilde{x}_3(\rho, \phi; \tau_1, \tau_2)$ with respect to the "space" variables ρ and ϕ is such a transform. In [3] it is shown that using the amplitude and phase information of this transform, a new representation F_x is obtained which has a unique correspondence with the class of original images that are mutually related with rotation-translation-scaling transformation. On the other hand, F_x is expressed in terms of the third-order correlation of $x(\mathbf{t})$. Thus, for a large image size N, the representation becomes insensitive to any type of additive noise having zero third order correlations. Properties [P1] - [P3] are therefore satisfied.

It should be emphasized that F_x is a *stand-alone* representation which can be used as a direct input to a neural network based or any other conventional classifier [4]; this is in contrast to other representations that require a *matching* procedure of the pattern to be classified with all available prototypes.

3. Feature Size Reduction - Discrete Implementation

A reduction in the size of the invariant representation is achieved, abolishing uniqueness in favor of computational efficiency. A reduced representation results if only the amplitude information of the Fourier transform of each triple-correlation class is kept dropping the phase information. A further reduction can be derived, by using only the zero-frequency Fourier coefficient of each class as a *sufficient* feature for classification. In our case, after careful consideration of simulation results, we concluded that the former choice provides an alternative which, without being very demanding in terms of the calculations involved and the amount of memory required, preserves sufficient information so as not to ruin the uniqueness of the representation.

A reduction of the model redundancy can also be obtained as stated above and explained in [3]. In that case, we can parametrize the initial triangles $W(\tau_1, \tau_2)$ using one of the following schemes:

(A) $(\tau_1, \tau_2) = (\tau_0, [k, l]), \qquad k \in [0, 1]\,,\ l \in [0, \infty)$

(B) $(\tau_1, \tau_2) = (\tau_0, \tau_2) \rightarrow (\theta_1, \theta_2) \qquad \theta_1, \theta_2 \in [0, \frac{\pi}{2}]$

(C) $(\tau_1, \tau_2) = (\tau_0, \tau_2) \rightarrow (\theta_1, \lambda) \qquad \theta_1 \in [0, \frac{\pi}{2}]\,,\ \lambda \in [0, \infty)$

where $\tau_0 = (1, 0)$, θ_1, θ_2 are the angles included between the plane vectors (τ_1, τ_2) and $(\tau_2, \tau_2 - \tau_1)$ respectively, and λ is the ratio of the length of vector τ_1 to the length of vector τ_2. In all of the above cases, the four dimensional space spanned by (τ_1, τ_2) is reduced, without any loss of information, to a 2-D space.

In order to reduce the computational burden in real life applications, where the original 2-D signal $x(\mathbf{t})$ is always available in discrete form, it is preferable to obtain $x_3(\tau_1, \tau_2)$ as the

inverse FFT of the bispectrum $X_3(\mathbf{u}, \mathbf{v})$, (the Fourier transform of $x_3(\tau_1, \tau_2)$). This is so, because

$$X_3(\mathbf{u}, \mathbf{v}) = X(\mathbf{u})\ X(\mathbf{v})\ X^*(\mathbf{u}+\mathbf{v}) \tag{6}$$

where $X(\mathbf{u})$ is the 2-D Fourier transform of $x(\mathbf{t})$. As a consequence, $X_3(\mathbf{u}, \mathbf{v})$ can be computed as the triple product of a 2-D FFT using fast software or hardware implementations. If the size of the initial input image is $N \times N$, this method can reduce the complexity of the algorithm from $O(N^6)$ (when $x_3(\tau_1, \tau_2)$ is computed via the definition) to $O(N^4 \log_2 N)$. Unfortunately, the price paid for this remarkable improvement is the enormous amount of memory required to store the entire 4-D representation of $x_3(\tau_1, \tau_2)$.

For this reason, an alternative scheme was used, which exploits the fact that binary images are actually matrices whose elements are either 0 or 1. In this case, successive rows of each matrix can be stored in binary integers. In addition, multiplication is then equivalent to logical AND and as a result, N multiplications can be executed in a single machine cycle, given that the wordsize of the machine used is always greater than or equal to the input image size N. Thus, the resulting complexity is now reduced to $O(N^5)$. This scheme requires no extra memory, as the elements of the triple correlation are calculated exactly when they are needed.

The next step is to to specify a discrete 2-D grid of (k, l) in (A), (θ_1, θ_2) in (B) or (θ_1, λ) in (C) that will determine the number of distinct classes $C(\tau_1, \tau_2)$. The quantization should be rather coarse defining the effectively distinct classes. Clearly, the above quantizations result in a possible loss of information. However, they provide the invariant representation with robustness to small distortions of the original 2-D signal due to the implicit averaging they introduce.

Finally, a further discretization should be applied in the interior of each class. The field $\tilde{x}_3(\rho, \phi; \tau_1, \tau_2)$ should be computed on a discrete grid of the parameters ρ and ϕ. The sampling rate in this domain is conceptually related to the number of triple-correlation lags that are assigned to each class.

At this point it should be noted that $\tilde{x}_3(\rho, \phi; \tau_1, \tau_2)$ is not calculated the way Eq. (4) implies: instead of calculating $x_3(\mathbf{T}_{\beta,\phi}\ \tau_1, \mathbf{T}_{\beta,\phi}\ \tau_2)$ for each ρ, ϕ, we first calculate $x_3(\tau_1, \tau_2)$ and then decide to which ρ, ϕ the calculated value corresponds. We also decide to which class this value belongs, depending on the class parametrization we have chosen, and we add it to the appropriate element of $\tilde{x}_3$. In addition, for each $\tau_1 \in S'$, τ_2 needs not span the entire S'; in fact it is restricted to the region

$$S''(\tau_1) = \{\ \tau_2 \in S' \colon 0 \le \tau_1 . \tau_2 \le |\tau_1|^2,\ 0 \le \tau_1 . \tau_2',\ (\tau_1 - \tau_2) \in S'\ \} \tag{7}$$

where τ_2' is derived from τ_2 by a 90 degrees clockwise rotation. The first two restrictions that are imposed on τ_2 are due to symmetries of triple correlation, while the last one simply follows from the fact that $x_3(\tau_1, \tau_2) = 0$ for $(\tau_1 - \tau_2) \notin S'$. Restricting τ_2 to a relatively small area greatly simplifies the calculation procedure.

Having available $\tilde{x}_3(\rho, \phi; \tau_1, \tau_2)$ for each class $C(\tau_1, \tau_2)$, in the form of a 2-D matrix, an FFT algorithm can be used to compute $\tilde{X}_3(P, \Phi; \tau_1, \tau_2)$. Since there is no interrelation between different classes, a parallel implementation of computations is possible.

Two more improvements can be made to speed up computations: First, all the horizontal shifts of the input image are calculated and stored for later use. Second, all possible products of the form $x(\mathbf{t})\ x(\mathbf{t} + \tau_1)$ are calculated before entering the loop that corresponds to τ_2. The optimized algorithm, in the case of the (θ_1, θ_2) class parametrization, is as follows:

Step 1: Calculate all the horizontal shifts of the input image.

Step 2: For each $\tau_1 \in S'$:

1) Find and store all possible products $x(\mathbf{t})\ x(\mathbf{t} + \tau_1)$, using the already calculated horizontal shifts and the logical operator AND.

2) Calculate and quantize ρ, ϕ by comparing τ_1 with $\tau_0 = (1, 0)$.

3) For each $\tau_2 \in S''(\tau_1)$:

3a) Find $x_3(\tau_1, \tau_2)$, using the products $x(\mathbf{t})\, x(\mathbf{t} + \tau_1)$.

3b) If $x_3(\tau_1, \tau_2) = 0$, proceed to the next value of τ_2.

3c) Calculate and quantize the included angles θ_1, θ_2.

3d) $\tilde{x}_3(\rho, \phi; \theta_1, \theta_2) := \tilde{x}_3(\rho, \phi; \theta_1, \theta_2) + x_3(\tau_1, \tau_2)$

Step 3: Find $\tilde{X}_3(P, \Phi; \theta_1, \theta_2)$ as a 2-D FFT transform of $\tilde{x}_3(\rho, \phi; \theta_1, \theta_2)$; keep the amplitude of the transform only.

4. Simulation Results

The performance of the proposed efficient scheme for invariant optical character recognition was tested on real images obtained by an optical laser scanner. Figure 1 depicts some of the images that were used, namely the uppercase and lowercase characters 'N', 'T', 'U' and 'A' in three sizes (pointsize 9, 10 and 11) and two directions. Figure 2 shows parts of the invariant representations of some of the images shown in Figure 1, namely the horizontal capital letters 'N', 'T', 'U', 'A' of pointsize 10, in the (θ_1, θ_2) domain. The average size of these letters was approximately 20x20, while (θ_1, θ_2) were quantized to a discrete grid of size (16x16) and (ρ, ϕ) to another of size (8x8), resulting in a 4-D representation of total size (128x128). However similar these representations might seem, they are quite different from each other, as the two Tables below show. On the contrary, the representations of three scaled and rotated versions of the capital letter 'N', which are depicted in Figure 3, are almost identical, demonstrating invariance as well as robustness to small shape distortions. In Table I, the difference between the representations of a set of letters is estimated by using Euclidean distances (sums of squares), while in Table II similar distances are shown between two sets of letters which are related to each other with rotation and scaling. As it can be clearly seen, our representation remains quite unchanged despite the transformations of the input images. Note, however, that the classifier would be very easily fooled in the case of the lowercase letters 'n' and 'u'. Similar difficulties were encountered with other pairs of letters such as 'b', 'q' and 'd', 'p', which are mutually related to each other with a rotation of 180 degrees, but can be easily overcome using other techniques.

5. Conclusions

A new efficient scheme for invariant optical character recognition was introduced in this paper, which allows real-time processing of binary images. Invariance of classification with respect to input image transformations and robustness to additive noise and distortions were achieved while the performance of the derived algorithm was tested on real image data.

6. References

[1] J. M. Mendel, *Tutorial in higher-order statistics (spectra) in signal processing and system theory: theoretical results and applications,* Proc. IEEE, 79, pp. 287-305, 1991.

[2] M. K. Tsatsanis, and G. B. Giannakis, *Object and texture classification using higher-order statistics,* IEEE Trans. on PAMI (to appear).

[3] A. Delopoulos, A. Tirakis and S. Kollias, *Invariant Image Classification Using Cumulant-Based Neural Networks,* IEEE Trans. on Neural Networks, to appear, 1993.

[4] A. Tirakis, A. Delopoulos and S. Kollias, *Cumulant-Based Neural Network Classifiers,* Proceedings of ICANN 1992, Brighton, 1992.

NTUA

ntua

NTUA

ntua

NTUA

ntua

Figure 1a

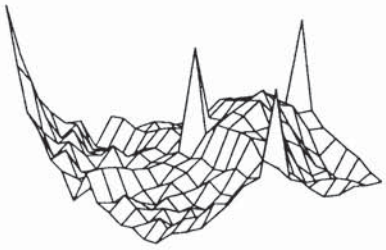

Figure 2a

Figure 2b

NTUA

ntua

NTUA

ntua

NTUA

ntua

Figure 1b

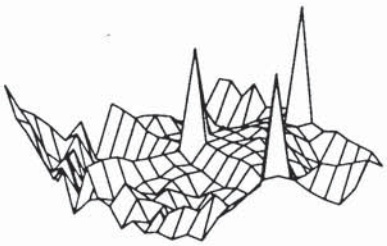

Figure 2c

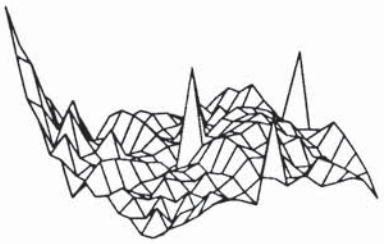

Figure 2d

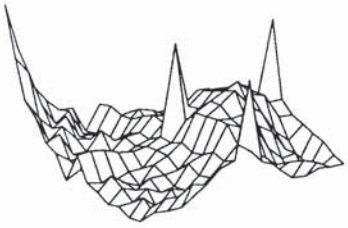

Figure 3a

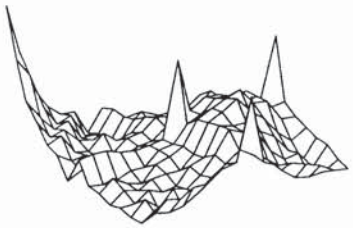

Figure 3b

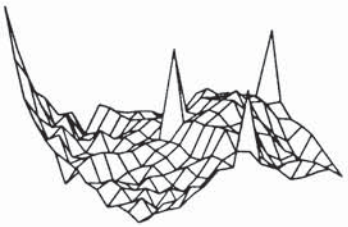

Figure 3c

Horizontal, Pointsize 10

Horizontal, Pointsize 10	'N'	'T'	'U'	'A'	'n'	't'	'u'	'a'
'N'	0.00	0.96	1.70	0.44	1.31	1.63	1.02	0.85
'T'	0.96	0.00	2.65	0.63	2.78	0.89	2.19	1.79
'U'	1.70	2.65	0.00	1.82	0.88	4.58	1.17	1.47
'A'	0.44	0.63	1.82	0.00	1.49	1.59	1.04	0.77
'n'	1.31	2.78	0.88	1.49	0.00	4.40	0.24	0.55
't'	1.63	0.89	4.58	1.59	4.40	0.00	3.55	3.09
'u'	1.02	2.19	1.17	1.04	0.24	3.55	0.00	0.34
'a'	0.85	1.79	1.47	0.77	0.55	3.09	0.34	0.00

Table I

Rotated, Pointsize 9

Horizontal, Pointsize 10	'N'	'T'	'U'	'A'	'n'	't'	'u'	'a'
'N'	0.06	0.97	2.49	0.49	1.55	1.88	1.68	0.97
'T'	1.35	0.07	3.69	0.62	2.98	1.34	3.20	1.99
'U'	1.94	2.44	0.18	1.97	1.45	4.57	1.36	1.46
'A'	0.57	0.55	2.48	0.08	1.47	1.77	1.68	0.84
'n'	1.30	2.54	1.23	1.64	0.47	4.47	0.37	0.60
't'	1.90	1.21	5.92	1.52	4.61	0.16	4.87	3.34
'u'	1.00	1.94	1.62	1.15	0.33	3.59	0.29	0.72
'a'	0.91	1.66	2.11	0.84	0.64	3.35	0.76	0.13

Table II