Character templates learning for textual images recognition as an example of learning in structural recognition

Bogdan Savchynskyy, Olexander Kamotskyy
International Research and Training Center
of Information Technologies and Systems,
Acad. Glushkova Ave., 40, Kiev, Ukraine
bogdan@image.kiev.ua

Abstract

[1] *Document recognition for digital libraries is characterized by high requirements to a recognition quality and processing of significant amount of single-type documents. So this is a perfect area for single-font approaches because they provide a smaller error rate comparing to multifont approaches and a learning of the font is carried out relatively rarely, because of significant amount of single-type documents.*

Traditionally character templates learning is performed for separated characters on a basis of the set of character examples. It leads to recognition errors like in situations when closely placed parts of neighbouring characters are recognized as a single, separate character.

We propose another approach to character templates learning. Namely such templates must be constructed that the result of recognition of a text line image as a whole must match to a text string specified by a teacher. The approach guarantees that not only images of separate characters will be recognized correctly, but also the segmentation of the whole text image into characters will be performed without errors. So in our approach a learning sample consists not from labelled images of separated characters, but from text line images with corresponding text strings.

1. Introduction

Document recognition for digital libraries is characterized by high requirements to a recognition quality and processing of significant amount of single-type documents. So this is a perfect area for single-font approaches because:

- the single-font approaches provide smaller error rate comparing to multifont ones (see [2, 11]);
- a learning of the font is carried out relatively rarely, because of significant amount of single-type documents.

A striking example of a single-font approach is a Document Image Decoding approach described in papers [1, 2, 3].

Traditionally character templates learning is performed for separated characters on a basis of the character examples set. It leads to recognition errors like in situations when closely placed parts of neighbouring characters are recognized as a single character.

We propose a novel approach to character templates learning. Namely such templates must be constructed that the result of recognition of a text line image as a whole must match to a text string specified by a teacher. The approach guarantees that not only images of separate characters will be recognized correctly, but also the segmentation of the whole text image into characters will be performed without errors. So in our approach a learning sample consists not from labelled images of separated characters, but from text line images with corresponding text strings.

This learning approach can be used only along with a proper recognition algorithm. Namely the algorithm must perform a segmentation and a recognition of text line image simultaneously via dynamic programming like it was discovered in [5] for the first time and like it was rediscovered in modern papers(see for ex. [2, 7, 8].

Our approach of templates learning is significantly based on the approach of tuning autonomous stochastic automata parameters, described in [10]. Namely the idea of learning problem formulation and solution is the development of the ideas presented in [10].

This paper consists of six sections. We formulate main definitions and a statement of the templates learning problem in the second section. The solution of the problem is presented in the third and the forth sections. The fifth section contains results of experimental testing and the last one is devoted to conclusions.

[1]This paper was supported by the EU INTAS project PRINCESS 04-77-7347

2. Learning problem formulation

First we will formulate the problem of a single text line image recognition, because its exact formulation is absolutely necessary for a formulation of the learning problem.

We call a rectangular subset of two-dimensional integer lattice *a field of vision*. Its height and width will be denoted H and W correspondingly:

$$T = \{(i,j) \mid i \in \overline{1,H}, j \in \overline{1,W}\} .$$

Elements of the field of vision T will be called *pixels*.

A denotation V will be used for the set of pixel brightnesses. This set will be later called *the set of signals*. *An image* is a function of the form $x : T \to V$. Brightness of the pixel t of the image x is denoted with $x(t)$ or x_t. Denotation V^T is used for the set of all possible images.

Let A be a finite set consisting of names of characters and $\alpha \in A$ be the name of a character. *A text line* is considered as a sequence of character names.

We consider the case when images of all the characters have the same height but their width depends just on the character's name. Thus the function $d : A \to \mathbb{N}$ will be regarded as a *function of a character width*. We consider this function to be known, so its values will not change during learning. We assume that the alphabet A contains also a special character *space* having width equal to one, so any spaces between characters and words can be modelled with a sequence of *spaces*.

A pair (α, j), where α is a name from the alphabet A and $j \in (1, 2, \ldots, W)$ is a column number of the field of vision, is called a *segment*. Each segment $s = (\alpha, j)$ determines a fragment $T(s)$ of the field of vision, which starts with a column number $j(s)$, has the name $\alpha(s)$, the height H and the width $d(\alpha(s))$:

$$T(s) = \left\{(i,j) \mid i \in \overline{1,H}, j \in \overline{j(s), j(s) + d(\alpha(s)) - 1}\right\} .$$

Thus segments are fragments of the field of vision labelled by names. The set of all possible segments is denoted with S.

Each text line $(\alpha_1, \ldots, \alpha_L)$ of the length L, depicted in the image x, uniquely corresponds to the sequence of closely fitted segments $(s_1, \ldots, s_L)$ which cover the whole field of vision T:

$$\begin{cases} \alpha(s_l) = \alpha_l \quad \forall l \in \overline{1,L} \\ j(s_1) = 1 \\ j(s_{l+1}) = j(s_l) + d(\alpha(s_l)) \quad \forall l \in \overline{1,L-1} \\ j(s_L) + d(\alpha(s_L)) = W + 1 \end{cases} . \quad (1)$$

We call the sequence $\overline{s} = (s_1, \ldots, s_L)$ satisfying conditions (1) *a segmentation*. Its length will be denoted as $L(\overline{s})$. The set of all possible segmentations of the field of vision will be denoted with $\overline{S}$.

Let us introduce the set E of parameter vectors. Each parameter vector $\overline{e}$ consists of templates of all characters: $\overline{e} = \{e(\alpha),\ \alpha \in A\}$. A template $e(\alpha)$ determines an ideal view of the character α. A template of a character can be, for example, a noise-free image of the character. Each template $e(\alpha)$ can be regarded as a vector itself. Thus we can interpret the parameter vector $\overline{e}$ as a vector in a multidimensional vector space, meaning that coordinates of templates are coordinates of the vector.

We will assume that *a local dissimilarity function* of the form $f : E \times S \times V^T \to \mathbb{R}$ is defined. Its value $f_{\overline{e}}(s, x)$ determines a similarity between a segment s of an image x and the character $\alpha(s)$. The greater the similarity, the less value the function takes. A value of the function can be for example equal to a sum of squared differences of image pixels and pixels of character's $\alpha(s)$ template.

The similarity of the whole image x and a text line $(\alpha_1, \ldots, \alpha_L)$, having a corresponding segmentation $\overline{s} = (s_1, \ldots, s_L)$, is defined as a total similarity of all the segments, i.e. it is equal to $\sum_{l=1}^{L(\overline{s})} f_{\overline{e}}(s_l, x)$.

Recognition of an image x representing a text line consist in a search for the segmentation $\overline{s}^* = (s_1^*, s_2^*, \ldots, s_L^*)$, which is the most similar to the image:

$$\overline{s}^* = \arg\min_{\overline{s} \in \overline{S}} \sum_{l=1}^{L(\overline{s})} f_{\overline{e}}(s_l, x) . \quad (2)$$

The problem (2) can be solved via dynamic programming, which was done for the first time by Kovalevsky [5].

Generally speaking, the function $\sum_{l=1}^{L(\overline{s})} f_{\overline{e}}(s_l, x)$ takes its minimal value on some set of optimal segmentation $\overline{S}^*$. In this case we will consider that $\overline{s}^*$ denotes any element of this set. A situation when the solution (2) must be unique will be distinguished with a sign $\stackrel{!}{=}$.

Further we will assume that a procedure for the search of an optimal segmentation $\overline{s}^*$ is available. This procedure solves the problem (2) for any triple $P = \langle A, d, f_{\overline{e}} \rangle$, which is the set of characters A, the width function d and a local disparity function $f_{\overline{e}}$, defined up to the vector of parameters $\overline{e}$. We will call this procedure *a recognition algorithm* $\overline{s}_P^*(x, \overline{e})$. Its implementation is not important for us, but we will assume that it can track the cases when the solution $\overline{s}^*$ is not unique.

Templates learning algorithm, corresponding to the recognition problem (2), consists in choosing a parameter vector $\overline{e}$ (i.e. choosing character templates) for a given triple $P = \langle A, d, f_{\overline{e}} \rangle$.

Let $X^o = \{x_r^o \mid r \in \overline{1,R}\}$ be a learning set of images and $\overline{S}^o = \{\overline{s}_r^o \mid r \in \overline{1,R}\}$ be the set of corresponding segmentations.

Problem 1 *(character templates learning) consists in search for such a parameter vector* $\overline{e}^*$ *that the result of*

recognition of each learning image x_r^o must be the only corresponding segmentation $\overline{s}_r^o$:

$$\overline{s}_r^o \stackrel{!}{=} \arg\min_{\overline{s}\in\overline{S}} \sum_{l=1}^{L(\overline{s})} f_{\overline{e}^*}(s_l, x_r^o) \qquad \forall r \in \overline{1,R}\ .$$

Complexity of the problem is determined by the fact that the minimum must be found on the set of segmentations $\overline{S}$, and the cardinal number of this set is proportional to the number of *all possible text lines*, which can be depicted in the field of vision.

3. Solution of the learning problem

We propose the solution of the learning problem 1 in the case, when a dissimilarity function is linear with respect to parameters $\overline{e} = \{e_1, \ldots, e_K\}$, i.e. it can be presented in the form:

$$f_{\overline{e}}(s,x) = \sum_{k=1}^{K} e_k \cdot \varphi_k(s,x) = \langle \overline{e}, \overline{\varphi}(s,x) \rangle\ . \qquad (3)$$

Here $\overline{\varphi}(s,x) = \{\varphi_1(s,x), \ldots, \varphi_K(s,x)\}$ is a vector, which directly depends on an image x and segment s.

Let us denote a learning sample consisting of images and corresponding segmentations as Π:

$$\Pi = \left\{ (x_r^o, \overline{s}_r^o) \mid r \in \overline{1,R} \right\}\ .$$

Now the learning problem 1 takes the form: find such $\overline{e}^*$, that

$$\overline{s}^o \stackrel{!}{=} \arg\min_{\overline{s}\in\overline{S}} \sum_{l=1}^{L(\overline{s})} \sum_{k=1}^{K} e_k^* \varphi_k(s_l, x^o) \quad \forall\, (x^o, \overline{s}^o) \in \Pi\ . \quad (4)$$

An equality (4) means, that the system of inequalities:

$$\begin{cases} \sum\limits_{l=1}^{L(\overline{s}^o)} \sum\limits_{k=1}^{K} e_k^* \varphi_k(s_l^o, x^o) < \sum\limits_{l=1}^{L(\overline{s})} \sum\limits_{k=1}^{K} e_k^* \varphi_k(s_l, x^o) \\ \qquad\qquad\qquad\qquad \forall \overline{s} \neq \overline{s}^o\ , \end{cases}$$

must hold for each pair $(x^o, \overline{s}^o) \in \Pi$.

These inequalities are equivalent to

$$\begin{cases} \sum\limits_{l=1}^{L(\overline{s})} \sum\limits_{k=1}^{K} e_k^* \varphi_k(s_l, x^o) - \sum\limits_{l=1}^{L(\overline{s}^o)} \sum\limits_{k=1}^{K} e_k^* \varphi_k(s_l^o, x^o) > 0 \\ \qquad\qquad\qquad\qquad \forall \overline{s} \neq \overline{s}^o\ . \end{cases}$$

After grouping components of parameter vector $\overline{e}^*$ and denoting all coeficients of e_k^* as $\varphi'_k(\overline{s}, \overline{s}^o, x^o)$ we will get:

$$\begin{cases} \sum\limits_{k=1}^{K} e_k^* \varphi'_k(\overline{s}, \overline{s}^o, x^o) > 0 \qquad \forall \overline{s} \neq \overline{s}^o. \end{cases} \qquad (5)$$

Thus the learning problem 1 consists in solving (5) with respect to the vector $\overline{e}^* = (e_1^*, \ldots, e_K^*)$.

Since the linear system (5) contains $\left|\overline{S}\right| - 1$ inequalities, the classical linear optimization methods cannot be used for its solution, because their complexity directly depends on the number of inequalities. But the solution of (5) can be found by means of linear discriminant analysis, containing algorithms which complexity does not depend on the cardinal number of the system, but on other its properties.

We used algorithms of Kozinec and perceptron (see, for ex. [10]) for the solution of the system (5). These algorithms are iterative and at each iteration parameter vector is corrected on the basis of a single unfulfilled inequality of the system (5). Thus they need only a method of effective search for such an inequality. The proof of a finite step convergence of these algorithms can be found in [10].

In our case, if any inequality of the system (5) does not hold for the current value of parameters $\overline{e}$, then it means that such a learning pair $(x^o, \overline{s}^o)$ exists that equality (4) does not hold and in its turn it means that the learning image x^o is recognized incorrectly. And vice versa, in order to check whether (5) is fulfilled we are to check whether (4) is fulfilled, in other words we must recognize all the learning images $x^o \in X^o$.

Thus Kozinec and perceptron algorithms iteratively refine parameter vector $\overline{e}$ by means of recognition procedure $\overline{s}_P^*(x, \overline{e})$. Under given $P = \langle A, d, f_{\overline{e}} \rangle$ the procedure finds the solution of the recognition problem (2). This solution determines such an inequality of the system (5) which is not fulfilled for a current value of parameters $\overline{e}$.

Algorithm 1 Kozinec algorithm for solving (5)

1: Choose any vector $\overline{e} \neq 0$ from the convex hull of the vectors set $\{\overline{\varphi}'\}$. For example, we can take it equal to $\overline{\varphi}'(\overline{s}, \overline{s}_r^o, x_r^o) \neq 0$, $\overline{s} = \overline{s}_P^*(x_r^o, 0)$ for any $r \in \overline{1,R}$.

2: Find an inequality which does not hold for the current value of $\overline{e}$:

$$\text{find } r \in \overline{1,R}, \text{ such that } \overline{s}_r^o \stackrel{!}{\neq} \overline{s}_P^*(x_r^o, \overline{e})\ .$$

3: If there is no such r then **end**. Parameter vector $\overline{e}^*$ is found.

4: Else calculate new parameter vector as a perpendicular dropped to a segment connecting vectors $\overline{e}$ and $\overline{\varphi}' = \overline{\varphi}'(\overline{s}_P^*(x_r^o, \overline{e}), \overline{s}_r^o, x_r^o)$:

$$\overline{e} := k \cdot \overline{e} + (1-k) \cdot \overline{\varphi}'\ , \qquad (6)$$

$$\text{where } k = \frac{(\overline{\varphi}', \overline{\varphi}')}{(\overline{e}, \overline{e}) + 2(\overline{e}, \overline{\varphi}') + (\overline{\varphi}', \overline{\varphi}')}\ . \qquad (7)$$

Proceed to step 2.

The perceptron algorithm looks quite similar to Kozinec one, but a bit simpler. Applied to the system (5) it takes the form:

Algorithm 2 Perceptron algorithm for solving (5)

1: Assign $\overline{e} = 0$.

2: Find an inequality which does not hold for the current value of $\overline{e}$:

$$\text{find } r \in \overline{1,R}, \text{ such that } \overline{s}_r^o \overset{!}{\neq} \overline{s}_P^* \left(x_r^o, \overline{e}\right).$$

3: If there is no such r then **end**. Parameter vector $\overline{e}^*$ is found.

4: Else calculate new parameter vector $\overline{e}$ by the formula:

$$\overline{e} := \overline{e} + \overline{\varphi}'(\overline{s}_P^* \left(x_r^o, \overline{e}\right), \overline{s}_r^o, x_r^o) \ .$$

Proceed to step 2.

To finish the consideration of the learning problem we must only concretize a local dissimilarity function $f_{\overline{e}}$. Its form determines a way of calculating vector $\overline{\varphi}'\left(\overline{s}, \overline{s}^o, x\right)$, which is used in algorithms 1 and 2.

4. Selection of the local dissimilarity function

In practice an input image of the character α is usually regarded as an ideal one $\chi\left(\alpha\right)$ distorted by some noise r. Most typically, the noise is considered as an additive one with spherically symmetric probability distribution $p_{\text{n}}\left(r\right)$:

$$x = \chi\left(\alpha\right) + r. \tag{8}$$

Here x is the distorted character image. In this case correlation approach [6, 4] to recognition can be used and a likelihood function can be used as a local dissimilarity function $f_{\overline{e}}$. Under assumption that a probability density $p_{\text{n}}\left(r\right)$ of the noise is a monotone decreasing function $f(\cdot)$ of the sum of its squared components:

$$p_{\text{n}}\left(r\right) = f\left(r^2\right) \ , \tag{9}$$

function $f_{\overline{e}}$ takes the form of the Euclidean distance between vectors $x = \left\{x_{i,j} \mid (i,j) \in T\left(s\right)\right\}$ and $\overline{e} = \chi\left(\alpha\right) = \left\{\chi_{i,j}^{\alpha} \mid (i,j) \in T\left(s\right)\right\}$:

$$f_{\overline{e}}\left(s,x\right) = \sum_{\substack{i\in\overline{1,H} \\ j=\overline{1,d(\alpha(s))}}} \left(\chi_{i,j}^{\alpha(s)} - x_{i,j(s)+j}\right)^2 \ . \tag{10}$$

Formula (10) can be presented in another form:

$$\begin{aligned} f_{\overline{e}}\left(s,x\right) = & \sum_{\substack{i\in\overline{1,H} \\ j=\overline{1,d(\alpha(s))}}} \left(\chi_{i,j}^{\alpha(s)} - x_{i,j(s)+j}\right)^2 = \\ = & \sum_{\substack{i\in\overline{1,H} \\ j=\overline{1,d(\alpha(s))}}} (x_{i,j(s)+j})^2 + \\ + & \sum_{\substack{i\in\overline{1,H} \\ j=\overline{1,d(\alpha(s))}}} \left((\chi_{i,j}^{\alpha(s)})^2 - 2\chi_{i,j}^{\alpha(s)} \cdot x_{i,j(s)+j}\right) = \\ = & \tilde{f}_{\overline{e}}\left(s,x\right) + \sum_{\substack{i\in\overline{1,H} \\ j=\overline{1,d(\alpha(s))}}} (x_{i,j(s)+j})^2 \ , \end{aligned} \tag{11}$$

where $\tilde{f}_{\overline{e}}\left(s,x\right)$ is used to denote a quantity $\sum_{\substack{i\in\overline{1,H} \\ j=\overline{1,d(\alpha(s))}}} \left((\chi_{i,j}^{\alpha(s)})^2 - 2\chi_{i,j}^{\alpha(s)} \cdot x_{i,j(s)+j}\right)$.

To avoid piling indexes we will use a nonstandard designation starting from this section. Namely, we will denote with $s \in \overline{s}$ an enumeration of all segments of segmentation $\overline{s}$. For example the designation $\sum_{l=1}^{L(\overline{s})} s_l$ is exactly the same as the designation $\sum_{s\in\overline{s}} s$.

Basing on the specified transformation (11) we rewrite a learning problem 1:

$$\begin{aligned} \overline{s}_r^o \overset{!}{=} & \arg\min_{\overline{s}\in\overline{S}} \sum_{s\in\overline{s}} f_{\overline{e}}\left(s,x\right) = \\ = & \arg\min_{\overline{s}\in\overline{S}} \left(\sum_{s\in\overline{s}} \tilde{f}_{\overline{e}}\left(s,x\right) + \sum_{s\in\overline{s}} \sum_{\substack{i\in\overline{1,H} \\ j=\overline{1,d(\alpha(s))}}} (x_{i,j(s)+j})^2 \right) = \\ = & \arg\min_{\overline{s}\in\overline{S}} \left(\sum_{s\in\overline{s}} \tilde{f}_{\overline{e}}\left(s,x\right) + \sum_{\substack{i\in\overline{1,H} \\ j\in\overline{1,W}}} (x_{i,j})^2 \right) \ . \end{aligned} \tag{12}$$

The second item in the formula (12) depends only on the input image x, not on a segmentation $\overline{s}$. Thus the item does not affect the argument of minimum search. So the formula (12) can be rewritten in the form:

$$\begin{aligned} \overline{s}_r^o \overset{!}{=} & \arg\min_{\overline{s}\in\overline{S}} \sum_{s\in\overline{s}} \tilde{f}_{\overline{e}}\left(s,x\right) = \\ = & \arg\min_{\overline{s}\in\overline{S}} \sum_{s\in\overline{s}} \sum_{\substack{i\in\overline{1,H} \\ j=\overline{1,d(\alpha(s))}}} \left((\chi_{i,j}^{\alpha(s)})^2 - 2\chi_{i,j}^{\alpha(s)} \cdot x_{i,j(s)+j}\right) \ . \end{aligned} \tag{13}$$

The target function in the formula (13) is nonlinear with respect to parameters $\chi^{\alpha}_{i,j}$, thus the problem can not be solved directly in the same manner like it was done in previous section.

For solving the problem we will substitute each function of the form $(\chi^{\alpha}_{i,j})^2 - 2\chi^{\alpha}_{i,j} \cdot x_{i,j(s)+j}$ such, that the only parameter $\chi^{\alpha}_{i,j}$ must be learned, with a function $\dot{e}^{\alpha}_{i,j} \cdot x_{i,j(s)+j} + e^{\alpha}_{i,j}$, which is linear in two parameters $\dot{e}^{\alpha}_{i,j}$ and $e^{\alpha}_{i,j}$. Obviously the set of such functions includes the set of the first-type functions and coincides with it in the case $e^{\alpha}_{i,j} \geq 0$ and $\dot{e}^{\alpha}_{i,j} = \pm 2 \cdot \sqrt{e^{\alpha}_{i,j}}$. But note that the introduced parameters $e^{\alpha}_{i,j}$ and $\dot{e}^{\alpha}_{i,j}$ do not have such a clear physical meaning like $\chi^{\alpha}_{i,j}$, which defines a colour of a given image pixel.

In this case the problem (13) takes the form:

$$\overline{s}^o_r \stackrel{!}{=} \arg\min_{\overline{s} \in \overline{S}} \sum_{s \in \overline{s}} \sum_{\substack{i \in \overline{1,H} \\ j = \overline{1,d(\alpha(s))}}} \dot{e}^{\alpha}_{i,j} \cdot x_{i,j(s)+j} + e^{\alpha}_{i,j} \ , \quad (14)$$

a parameter vector $\overline{e}$ is composed from all the coefficients:

$$\overline{e} = \left\{ \dot{e}^{\alpha}_{i,j}, e^{\alpha}_{i,j} \mid \alpha \in A, i \in \overline{1,H}, j = \overline{1,d\left(\alpha\right)} \right\},$$

and the vector $\overline{\varphi}'\left(\overline{s}, \overline{s}^o, x\right)$ from the problem (14) consists of components $\dot{\varphi}'^{\alpha}_{i,j}$ and $\varphi'^{\alpha}_{i,j}$, which depend on the colours of an input image x:

$$\overline{\varphi}'\left(\overline{s}, \overline{s}^o, x\right) = \left\{ \dot{\varphi}'^{\alpha}_{i,j}, \varphi'^{\alpha}_{i,j} \mid \alpha \in A, i \in \overline{1,H}, j = \overline{1,d\left(\alpha\right)} \right\} \ , \quad (15)$$

where

$$\begin{aligned} \dot{\varphi}'^{\alpha}_{i,j} &= \sum_{\substack{s \in \overline{s} \\ \alpha(s) = \alpha}} x^o_{i,j(s)+j} - \sum_{\substack{s \in \overline{s}^o \\ \alpha(s) = \alpha}} x^o_{i,j(s)+j} \ , \\ \varphi'^{\alpha}_{i,j} &= \sum_{\substack{s \in \overline{s} \\ \alpha(s) = \alpha}} 1 - \sum_{\substack{s \in \overline{s}^o \\ \alpha(s) = \alpha}} 1 \ . \end{aligned}$$

Experiments with Kozinec and perceptron algorithms for learning vector containing parameters $\dot{e}^{\alpha}_{i,j}$, $e^{\alpha}_{i,j}$ in the problem (14) showed, that on real images learning takes such an amount of time which is too much for practical use of these algorithms. Significant amount of the algorithm iterations can be explained with such empirical considerations. Different components of the vector $\overline{\varphi}'\left(\overline{s}, \overline{s}^o, x\right)$ affect the values of parameter vector in nonuniform way. For example, each increase of coefficient $\dot{e}^{\alpha}_{i,j}$ by value $x_{i,j(s)+j}$, which is the colour of an image at some point and takes value, for example, in the interval from 0 to 255, requires adequate learning of parameter $e^{\alpha}_{i,j}$. But this operation takes up to 255 cycles of perceptron algorithm, in each cycle $e^{\alpha}_{i,j}$ is increased by 1.

A uniform influence on different components of parameter vector can be provided with an orthonormalized basis for functions $\dot{e}^{\alpha}_{i,j} \cdot x_{i,j(s)+j} + e^{\alpha}_{i,j}$. These functions can be represented in such a form:

$$\overline{s}^o_r \stackrel{!}{=} \arg\min_{\overline{s} \in \overline{S}} \sum_{s \in \overline{s}} \sum_{\substack{i \in \overline{1,H} \\ j = \overline{1,d(\alpha(s))}}} \left(\dot{e}^{\alpha}_{i,j} \cdot \psi_1\left(x_{i,j(s)+j}\right) + e^{\alpha}_{i,j} \cdot \psi_0\left(x_{i,j(s)+j}\right) \right) , \quad (16)$$

where functions $\psi_i(x)$ are ortonormalized Chebyshev polynomials. Each $\psi_i(x)$ is an i-th order polynomial of the signal value x. Orthonormalization means that such restrictions hold:

$$\sum_{x \in V} \psi_i(x) \cdot \psi_j(x) = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases} . \quad (17)$$

In this basis the components (15) of the vector $\overline{\varphi}'\left(\overline{s}, \overline{s}^o, x\right)$ take a form:

$$\begin{aligned} \dot{\varphi}'^{\alpha}_{i,j} &= \sum_{\substack{s \in \overline{s} \\ \alpha(s) = \alpha}} \psi_1\left(x_{i,j(s)+j}\right) - \sum_{\substack{s \in \overline{s}^o \\ \alpha(s) = \alpha}} \psi_1\left(x^o_{i,j(s)+j}\right) , \\ \varphi'^{\alpha}_{i,j} &= \sum_{\substack{s \in \overline{s} \\ \alpha(s) = \alpha}} \psi_0\left(x_{i,j(s)+j}\right) - \sum_{\substack{s \in \overline{s}^o \\ \alpha(s) = \alpha}} \psi_0\left(x^o_{i,j(s)+j}\right) . \end{aligned}$$

Due to the use of such a basis for a local dissimilarity function processing time of algorithms 1 and 2 decreased by two-five times on real images.

5. Experimental testing

The proposed approach was tested on real and artificial images. Two other approaches have been used to compare with our one.

The first approach follows from the maximum likelihood estimation of templates under condition that an image noise is described with formulae (8) and (9). The solution of the estimation problem comes to averaging of learning images' fragments corresponding to the same character. The use of obtained templates as parameter vector of recognition algorithm with the dissimilarity function of the form (10) will be called *a method of templates averaging*.

The second approach is a multifont one. We took a popular commercial OCR program **FineReader** as a representative of this approach.

We present the results of testing learning algorithms 1 and 2 on real images like one presented in the fig. 1. The procedure of images creation was the following: computer text was printed using a dot-matrix printer with an old printing band that caused a nonuniform noise in the image. To avoid errors connected with text lines detection and skew

Table 1. Percentage of errors committed by the proposed algorithm (learning), algorithm of templates averaging and multifont commercial OCR program FineReader.

algorithm	Percentage of errors
learning	3,8
averaging	13
FineReader	17,5

all the image text lines have been separated and unskewed manually.

Then for each line that was used for learning a corresponding segmentation (i.e. names and exact horizontal position of each character) was entered by a teacher. Producing such a segmentation manually is a hard work, so in our experiments it was done automatically. For an automatic creation of the segmentation just a text line without information about character positions was entered by the teacher. The character positions were then produced with an unsupervised learning algorithm like one presented in [9].

A Kozinec algorithm with the local dissimilarity function of the form defined with formulae (16) and (17) was used to learn character templates.

We used 100 segmented text lines for testing. Ten on these lines have been used to learn templates. About thirty lines including all the learning lines labelled by ticks are stacked in a single image in the fig. 1. In the fig. 2 some of input lines and lines containing recognition results are stacked together. Learning lines are also ticked off. It is easy to see that lines used for learning are recognized without errors as it is demanded by the problem 1 formulation.

The results of testing are presented at the table 1. Lines used for learning have not been considered. You can see that the proposed approach surpasses template averaging approach more then in three times and multifont recognition software more then in four times.

The approach presented in the article has been tested also on artificial images. These images were of two types. Images of the first type were noised according to formulae (8) and (9). But the difference in recognition results between the proposed approach and the approach of averaging templates for such images was negligible.

Images of the second type were noised in completely another manner. Namely parts of character images were shifted or deformed such that the averaging gave incorrect recognition results in contrast to our approach, that gave error-free results.

The simplest example of such an artificial tests is presented in the fig. 3 and 4. After learning on the image 3 with averaging approach the image 4 was recognized as CCE in

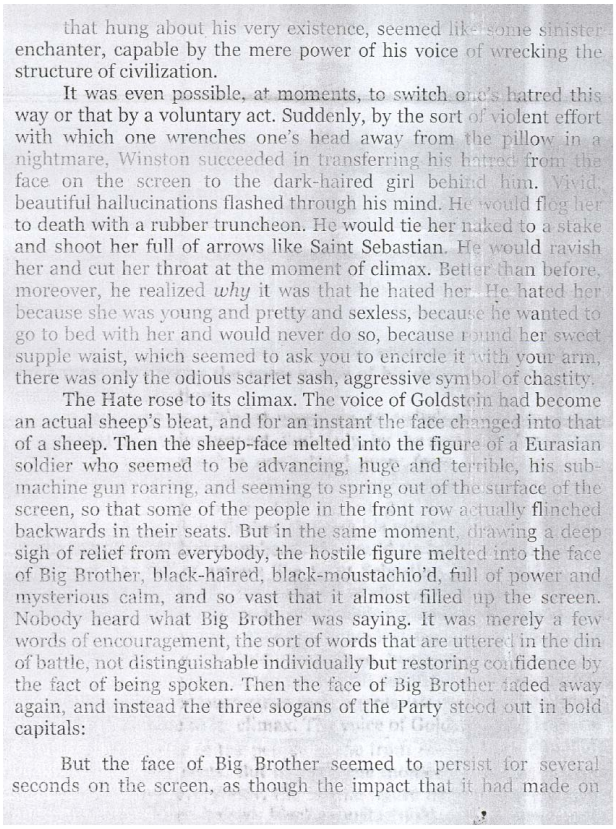

Figure 1. Some of image lines used to test the proposed learning approach. Lines used for learning are ticked off.

contrast to our approach that leaded to error-free recognition.

Quantitatively but not qualitatively more complicated example is presented in the fig. 5 and 6. Degraded characters are labelled with a grey background. Recognition results after learning with the averaging approach are presented in the fig. 6, while our approach gives error-free results.

6. Conclusions

Theoretical and practical analysis of the proposed approach argues:

- the proposed approach leads to approximately the same recognition quality as an averaging approach in a case of known image noise, described with formulae (8) and (9);
- in cases when noise distribution is far from being described with formulae (8) and (9) the proposed ap-

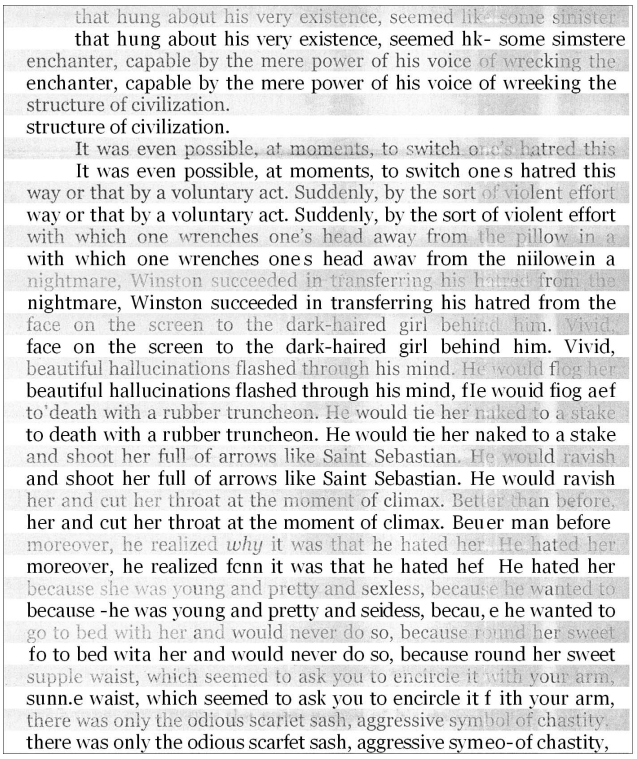

Figure 2. The result of recognition of the first half of image lines in the fig. 1. Lines used for learning ticked off and according to problem 1 formulation are recognized without errors.

proach surpasses the averaging approach and other approaches based on a maximum likelihood estimation of character templates;

- learning time can be significantly decreased if the proposed learning algorithm starts from a good approximation of the character templates received for example by averaging approach;
- a learning quality in the approach depends not on the number of samples but on that how learning samples cover all the cases that appear during recognition phase. Good recognition results were achieved in the example presented in the fig. 1, because lines from learning sample contained as character images from

Figure 3. A very simple example of artificial image. The last character E is degraded.

Figure 4. After learning on the image 3 with averaging approach this image was recognized incorrectly as CCE in contrast to our learning approach which leaded to a error-free result.

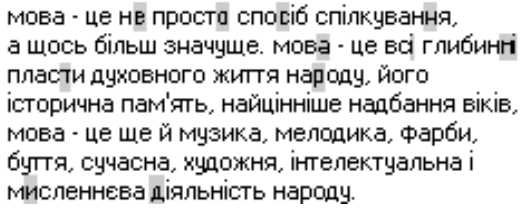

Figure 5. An artificial image for learning. Degraded characters labelled with a grey background.

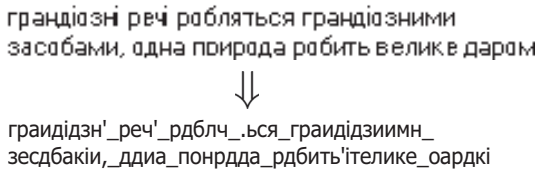

Figure 6. An input image is at the top, the result of recognition is at the bottom. The recognition result was achieved after learning with the averaging approach on the image 6.

the left, relatively "good" part of the whole image, so and character images from the right, relatively "bad" one;

- the basis of the local dissimilarity function must be chosen carefully. In our case the use of orthonormalized Chebyshev polynomials enabled a possibility to work with real images;
- the proposed learning approach can be used not only for text recognition, but also for a wide range of structural recognition problems. The only condition is that the problem must be formulated in a form (2) with a linear dissimilarity function (3) and recognition problem itself must be solvable too.

References

[1] P. A. Chou and G. E. Kopec. A stochastic attribute grammar model of document production and its use in document image decoding. In H. B. L. Vincent, editor, *Document Recognition II*, volume 2422 of *SPIE Proc.*, pages 66–73, 1995.

[2] G. E. Kopec and P. A. Chou. Document image decoding using markov sourcemodels.

[3] G. Kopec and L. M. Document-specific character template estimation. In *IS&T/SPIE 1996 Intl. Symposium on Electronic Imaging: Science & Technology*, San Jose, CA, Jan. 27–Feb. 2, 1996.

[4] V. Kovalevsky. Korreliacionnyj metod raspoznavaniya izobrazhenij; in russian (Correlation method of image recognition). *Journal vychislitelnoj matematiki i matematicheskoj fiziki*, 2(4), 1962.

[5] V. Kovalevsky. Optimalnyj algoritm raspoznavania nekotorych posledovatelnostej izobrazhenij; in russian (Optimal algorithm recognizing some sequences of images). *Kibernetika*, (4):75–80, 1967.

[6] V. Kovalevsky. *Metody optimalnych reshenij v rasposnavanii izobrazhenij; in russian (Methods of optimal solutions in image recognition)*. Nauka, Moskow, 1976.

[7] U. Marti and H. Bunke. Handwritten sentence recognition. In *Proceedings of 15 th International Conference on Pattern Recognition*, volume 3, pages 467–470, Barcelona, 2000.

[8] R. Plamodon and S. N. Srihari. On-line and off-line handwriting recognition: A comprehensive survey. *IEEE Trans. on PAMI*, 22(1):63–84, January 2000.

[9] P. Sarkar, H. S. Baird, and X. Zhang. Training on severely degraded text-line images. In *IAPR 7th International Conference on Document Analysis and Recognition (ICDAR03)*, pages 38–43, Edinburgh, Scotland, August 2003.

[10] M. I. Schlesinger and V. Hlavač. *Ten lectures on statistical and structural pattern recognition*. Kluwer Academic Publishers, Dordrecht/Boston/London, 2002.

[11] Y. Xu and G. Nagy. Prototype extraction and adaptive OCR. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21(12):1280–1296, 1999.