Image Preprocessing for Improving OCR Accuracy

Wojciech Bieniecki, Szymon Grabowski and Wojciech Rozenberg

Abstract - **Digital cameras are convenient image acquisition devices: they are fast, versatile, mobile, do not touch the object, and are relatively cheap. In OCR applications, however, digital cameras suffer from a number of limitations, like geometrical distortions. In this paper, we deal with the preprocessing step before text recognition, specifically with images from a digital camera. Experiments, performed with the FineReader 7.0 software as the back-end recognition tool, confirm importance of image preprocessing in OCR applications.**

Keywords – **Image preprocessing, OCR, digital cameras.**

I. INTRODUCTION

Optical character recognition belongs to the most important image analysis tasks. Its main applications are in building digital libraries (containing text, math formulas, music scores etc.), recognizing objects on digitalized maps, localization of vehicle license plates [1], text readers for visually impaired people, understanding hand-written office forms, bank checks etc. A typical OCR system consists of the following steps:

- image preprocessing, e.g. noise attenuation, correction of image orientation;
- image binarization, usually performed adaptively [2];
- segmentation [3], usually hierarchical (recognizing page layout, detecting text areas (and tables, figures etc.), then text paragraphs, individual lines, then segmenting lines into words, and finally words into characters);
- actual recognition (supervised or unsupervised) [4, 5];
- spellchecker-guided postprocessing;
- saving output in some popular format (html, PDF, LaTeX).

We noticed that mainstream commercial OCR tools were optimized for scanner-captured rather than camera-captured documents [6]. This goes against the trends of recent years, in which digital cameras dominated the market of acquisition devices. Digital cameras are fast, versatile, mobile, and relatively cheap. Moreover, they do not touch the photographed object, which is sometimes very convenient. However, cameras also have some drawbacks. They suffer from geometrical distortions (e.g., when photographing a thick volume), focus loss and uneven document lighting.

In our work we assume typewritten documents, acquired from a digital camera.

II. GOALS OF IMAGE PREPROCESSING BEFORE OCR

In our previous work [6] we ran a simulation of OCR detection under controlled influence of noise, geometric distortions, varying image resolution and lighting conditions. The experiments, measuring OCR accuracy of the FineReader 7.0 software, were carried out on the following images:

- a formatted page of a MS Word document saved as a 600 dpi bitmap (simulation of a "perfect" image, ideal conditions);
- same image, image resolution decreased to 300 dpi and 150 dpi, respectively;
- same image (600 dpi and 300 dpi), affected with 10%, 20% and 30% Gaussian noise (simulating low-lighting conditions and resulting CCD matrix noise);
- previous image filtered with the classic median noise removal filter;
- original image (600 dpi) under typical geometric distortions: rotation, skew or "fish-eye" (simulating photos taken from a bad angle, wide zoom);
- original image (600 dpi) affected with uneven lighting.

A couple of conclusions were drawn from those experiments. The OCR process was not much hampered by uneven lighting. Noise was much more detrimental, but for 10% amount of noise and high resolution (600 dpi) the accuracy still remained viable. The median filter appeared totally inappropriate for our scenario, resulting in further accuracy loss. The FineReader accuracy dramatically drops for image resolution below 300 dpi. But, interestingly, upsampling a low-resolution image results in significant accuracy improvement.

Nonetheless, OCR recognition was most sensitive to geometric deformations. These types of artifacts make it difficult to trace text lines, which may imply, that further OCR processing of an image rotated by e.g. 10 degree totally fails. Unlike noise and low resolution obstacles, geometric deformations may be (in theory) eliminated once we are aware they may appear.

III. IMAGE AUTOROTATION

A common problem in an early stage of OCR preprocessing is to adjust the orientation of text areas. The text lines should be parallel to image boundaries. In this section we present an algorithm solving this problem. Our idea is to find a straight line supporting the left margin of the found text area (Fig. 1). The slope of this line defines the page rotation angle.

In the beginning, the acquired image is converted to binary (using *im2bw* function from Matlab's Image Processing Toolbox), where value 0 indicates a black pixel, to speed up further processing. Then, every *k*-th horizontal line is scanned, from left to right, and the first occurring black pixel on each scanned line is stored in the array Tab.

W. Bieniecki, Sz. Grabowski, W. Rozenberg – Computer Engineering Dept. Technical University of Lodz, al. Politechniki 11, Lodz, POLAND. E-mail: wbieniec@kis.p.lodz.pl

$$Tab = \left((x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\right) \quad (1)$$

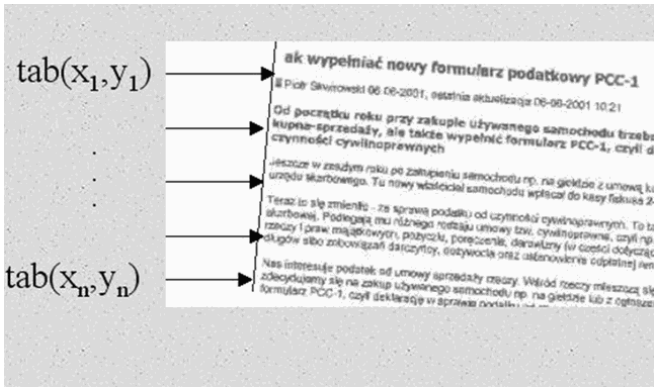


Fig. 1. Margin line detection

Smaller values of k improve the accuracy but naturally slow down the processing. We experimentally set k to 5.

To the next stage of the procedure we must select pixels belonging to the left boundary only. The image may be rotated clockwise or counterclockwise. In the first case we should remove points belonging to the bottom line of the text (Fig. 2).

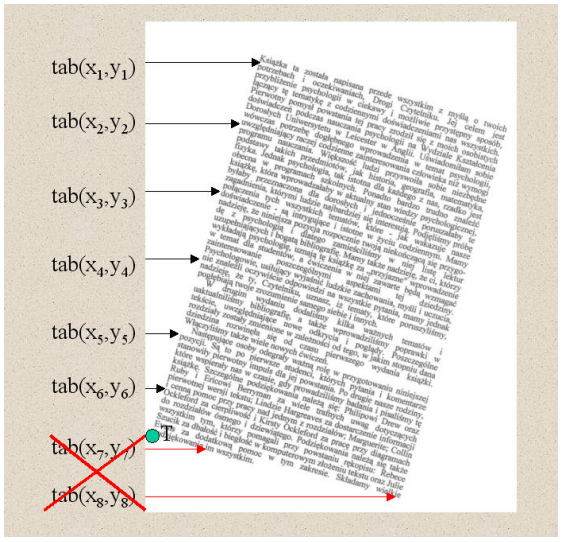


Fig. 2. Selection of the points belonging to the bottom line. The text is clockwise rotated.

In a common case (assuming clockwise rotation of the text) we initially build two sets of points satisfying the condition:

$$Tab_1 = \{(x_k, y_k) : \forall i = 1 \ldots n,\ x_{i-1} > x_i\}$$
$$Tab_2 = Tab \setminus Tab_1 \quad (2)$$

Then we choose the set which represents the line that is "more vertical", i.e. the larger of those two sets, assuming the text orientation is "portrait" rather than "landscape" one.

$$Tab_N = \begin{cases} Tab_1 & if \quad |Tab_1| > |Tab_2| \\ Tab_2 & otherwise \end{cases}. \quad (3)$$

In the text step, we eliminate the points indicated red in Fig. 3.

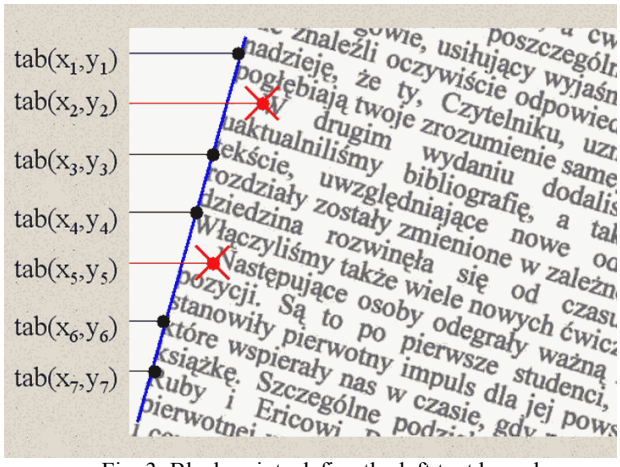


Fig. 3. Black points define the left text boundary. Red points are misleading and should be eliminated.

The criterion for these points is the average displacement of x coordination for the successive points from Tab_N. We denote the remaining set as Tab_S.

$$dx = \underset{i=1\ldots k}{mean}(x_i - x_{i-1}),\ (x_i, y_i) \in Tab_N \quad (4)$$

$$Tab_S = \left\{ \begin{matrix} (x_k, y_k) \in Tab_N : \\ \forall i = 1 \ldots k, |x_i - x_{i-1} - dx| < 0.1 \cdot dx \end{matrix} \right\} \quad (5)$$

Then we select two points from Tab_S, p_{min} and p_{max}, having respectively the minimum and the maximum x coordinate in Tab_S. Having found them, we evaluate the line slope and the angle for the rotation.

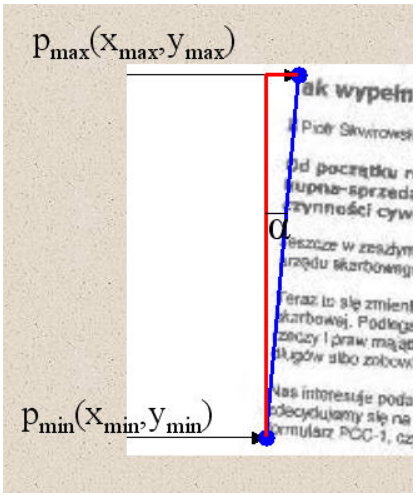


Fig. 4. Knowledge of the extreme points p_{min} and p_{max} makes the rotation trivial

IV. Perspective Correction

Sometimes the image acquisition device (esp. a digital camera) is positioned not orthogonally to the paper sheet. As a result, the acquired area is not a rectangle, but rather a trapezoid or a parallelogram. Experiments show that

especially the latter sort of distortion is harmful for the OCR process.

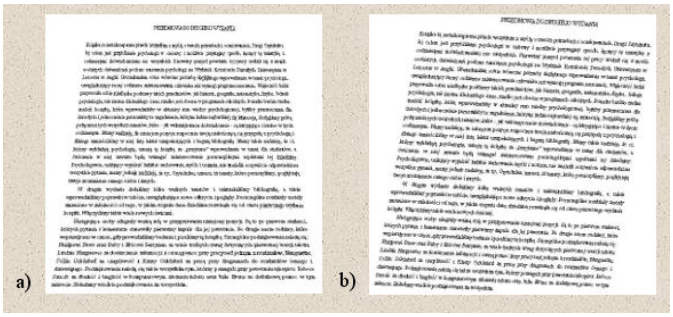


Fig. 5. Image distortion due to improper camera positioning

The first step toward image perspective correction is pointing out the quadrilateral containing the text scope. This operation can be carried out manually or with aid of the technique from the previous section.

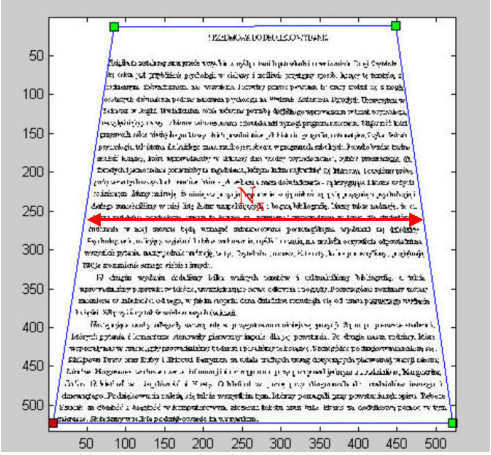


Fig. 6. Finding the bounding rectangle

Let I be the image containing the whole page, and define the following points of the quadrilateral:

$$p_1 = (x_{p1}, y_{p1}) = left\_top \qquad p_2 = (x_{p2}, y_{p2}) = right\_top$$
$$p_3 = (x_{p3}, y_{p3}) = left\_bottom \qquad p_4 = (x_{p4}, y_{p4}) = right\_bottom \tag{6}$$

Initially, the new dimensions of the rectangle bounding the text area are evaluated:

$$N_X = \frac{\Delta p_{12} + \Delta p_{34}}{2}, \quad N_Y = \frac{\Delta p_{23} + \Delta p_{41}}{2}. \tag{7}$$

where Δp_{ij} is the length of the line segment $p_i p_j$.

For all of the points of the new image containing the text area $I_N = \{(i,j) : i = 1...N_X, j = 1...N_Y\}$ the color of the pixel is interpolated.

$$p_{1j} = \frac{j \cdot x_{p2} + (N_X - j) \cdot x_{p1}}{N_X}, \; p_{1i} = \frac{j \cdot y_{p2} + (N_X - j) \cdot y_{p1}}{N_X}$$
$$p_{2j} = \frac{i \cdot x_{p2} + (N_Y - i) \cdot x_{p3}}{N_Y}, \; p_{2i} = \frac{i \cdot y_{p2} + (N_Y - i) \cdot y_{p3}}{N_Y}$$
$$p_{3j} = \frac{j \cdot x_{p3} + (N_X - j) \cdot x_{p4}}{N_X}, \; p_{3i} = \frac{j \cdot y_{p3} + (N_X - j) \cdot y_{p4}}{N_X} \tag{8}$$
$$p_{4j} = \frac{i \cdot x_{p1} + (N_Y - i) \cdot x_{p4}}{N_Y}, \; p_{4i} = \frac{i \cdot y_{p1} + (N_Y - i) \cdot y_{p4}}{N_Y}$$

Formula (9) evaluates the coordinates (x,y) of the point from the original image which will be copied onto the (i,j) pixel in the new image.

$$x = (p_{3j} \cdot (p_{3i} - p_{1i}) \cdot (p_{4j} - p_{2j}) - p_{3i} \cdot (p_{3j} - p_{1j}) \cdot (p_{4j} - p_{2j}$$
$$p_{4j} \cdot (p_{4i} - p_{2i}) \cdot (p_{3j} - p_{1j}) + p_{4i} \cdot (p_{3j} - p_{1j}) \cdot (p_{4j} - p_{2j})) /$$
$$((p_{3i} - p_{1i}) \cdot (p_{4j} - p_{2j}) - (p_{4i} - p_{2i}) \cdot (p_{3j} - p_{1j})),$$

$$y = (p_{3i} \cdot (p_{3j} - p_{1j}) \cdot (p_{4i} - p_{2i}) - p_{3j} \cdot (p_{3i} - p_{1i}) \cdot (p_{4i} - p_{2i}) \tag{9}$$
$$p_{4i} \cdot (p_{4j} - p_{2j}) \cdot (p_{3i} - p_{1i}) + p_{4j} \cdot (p_{3i} - p_{1i}) \cdot (p_{4i} - p_{2i})) /$$
$$((p_{3j} - p_{1j}) \cdot (p_{4i} - p_{2i}) - (p_{4j} - p_{2j}) \cdot (p_{3i} - p_{1i})).$$

V. NON-LINEAR IMAGE TRANSFORMATION

The need for a non-linear image transformation is easily understandable if we look at a photograph of an open book (Fig. 7).

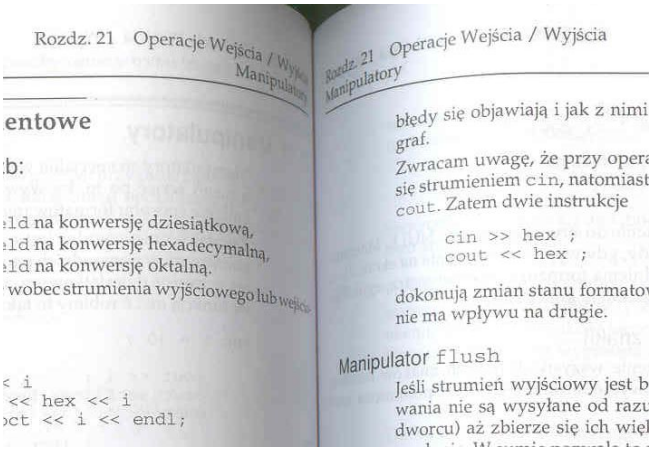


Fig. 7. The image of scanned book

The OCR system fails, because the text lines are curved and near the centre the letters are narrower.

The first step toward the correction is to trace the curves bounding the upper and lower borders of the text area. We assume that the left and right borders are straight lines. We interpolate the curves using splines with four knot points. In the currect implementation the lines are adjusted manually.

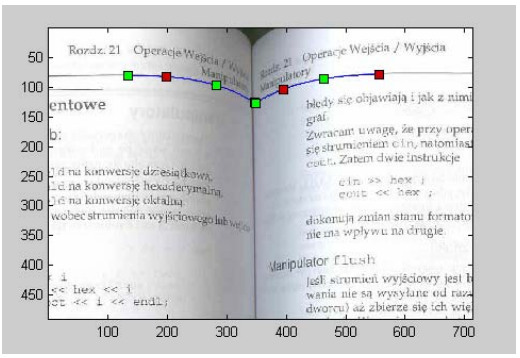


Fig. 8. The process of adjustment of the bounding curves

Having pointed out both upper and lower curve, the actual correction procedure is run. Each point of the spline curve is described by parameter $t \in [0;1]$. For a specific value *t*, the straight line between upper and lower curve is evaluated. The length of the line between starting points $(x_u(0), y_u(0))$ and $(x_l(0), y_l(0))$ is the pattern height of the new image. Each line $(x_u(t), y_u(t))$ is then transformed to the perpendicular line using the interpolation equations (see previous section). As a result, a rectangle image is created.

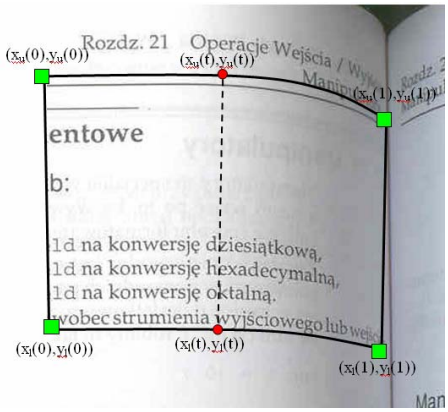


Fig. 9. Finding the bounding curves of the region to process

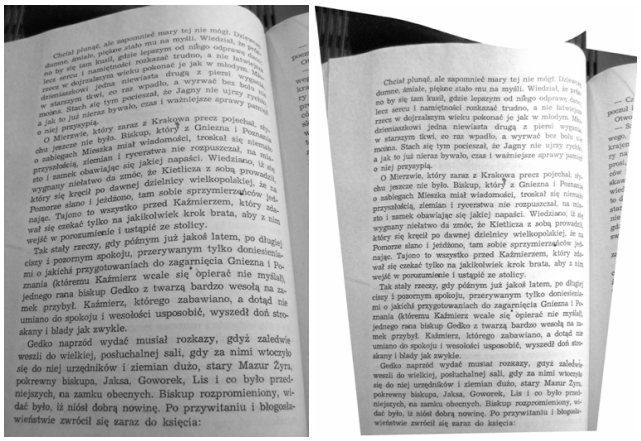


Fig. 10. The result of image restoration

VI. Experimental evaluation

In all the tests, FineReader 7.0 was used as the OCR software.

The first experiment was conveyed on a scanned A4 page, containing text from a Polish book. There were 2698 characters (excluding whitespaces) on the page. In these tests, defects were added artificially, with aid of Paint Shop Pro 9 image processing software, and Matlab scripts.

Table 1 shows how the image resolution affects the OCR accuracy. Note that, suprisingly, very high resolution (600 dpi) leads to accuracy deterioration, which is probably some peculiarity of the used software (we suspect that the FR 7.0 recognition algorithm is tuned for 300 dpi images).

In the next experiment noise of varying intensity was added (*imnoise* Matlab's IPT function) to the test image of 300 dpi. We tried: speckle, salt & pepper (s&p) and white Gaussian noise. The deteriorated images were then processed with the filters. Detailed results are presented in Table 2. The median filter for samples affected with speckled noise as well as Gaussian gives unstable results, sometimes hampering further OCR process significantly. Only s&p noise may be successfully attenuated.

TABLE 1

OCR errors for different image types and resolutions

image	resolution (dpi)	error [%]
scanned	75	0.96
scanned	100	0.14
75dpi, interp.	150	0.59
scanned	150	0.03
100dpi, interp.	200	0.07
scanned	200	0.22
150dpi, interp.	300	0.11
scanned	300	0.07
600dpi, interp.	300	0.11
200dpi, interp.	400	0.07
scanned	600	0.40

TABLE 2

OCR errors for noisy images, filtered or unfiltered

noise	filter	error [%]	notes
s&p, 10%	–	–	text area error
s&p, 10%	median, 3x3	0.03	–
s&p, 10%	Wiener, 5x5	0.07	–
s&p, 20%	–	–	text area err., language misrecogn.
s&p, 20%	median, 5x5	0.14	–
s&p, 30%	–	–	unrecogn. text
s&p, 30%	median, 5x5	0.44	–
s&p, 30%	median, 7x7	1.63	–
speckle, 10%	–	0.81	–
speckle, 10%	median, 3x3	0.07	–
speckle, 10%	median, 5x5	0.11	–
speckle, 10%	Wiener, 5x5	0.18	–
speckle, 10%	min, 3x3	1.03	–
speckle, 10%	thresh. 140 + max, 3x3	0.14	–
speckle, 20%	–	–	unrecogn. text
speckle, 20%	median, 5x5	0.40	
speckle, 20%	median, 7x7	1.18	im. too dark
speckle, 20%	Wiener, 5x5	0.37	–
speckle, 20%	Wiener, 7x7	1.14	im. too dark
Gauss., 30%	–	0.11	–
Gauss., 40%	–	–	text area error
Gauss., 40%	median, 3x3	–	text area error
Gauss., 40%	min, 3x3	0.14	–
Gauss., 40%	median, 3x3 + intensity adjustment	0.11	*imadjust*([0.8 1], [0 1])
Gauss., 40%	median, 3x3 + intensity adjustment	0.00	*imadjust*([0.9 1], [0 1])

In the next experiment the input image was rotated by a known angle before scanning. Three correction types have

been considered: FineReader's automatic one, "manual" correction (i.e. we explicitly set the rotation angle) and automatic in our application. The results are shown in Table 3.

The FR 7.0 can easy cope with slightly rotated images, but if the rotation angle exceeds about 10º, then the text area is completely misdetected.

TABLE 3

OCR ERRORS FOR A ROTATED AND THEN CORRECTED IMAGE

image	correction method	error [%]	notes
right rot., 3º	FR, auto	0.07	some chars in italic
right rot., 3º	auto	0.03	–
right rot., 3º	manual	0.03	–
right rot., 5º	FR, auto	0.59	–
right rot., 5º	auto	0.11	–
right rot., 5º	manual	0.03	–
right rot., 15º	–	–	recogn. area misdetected
right rot., 15º	auto	0.07	–
right rot., 15º	manual	0.07	–
left rot., 15º	–	–	recogn. area misdetected
left rot., 15º	auto	0.11	–
left rot., 15º	manual	0.11	–
left rot., 90º	–	–	unrecogn. text
left rot., 90º	manual	0.03	–

Perspective correction was the goal of the next test. We used Paint Shop Pro 9 to transform a text page into a trapezoid or a rhomboidal shape. Fig. 11 illustrates the three considered perspective destortions. OCR errors on (corrected with the algorithm from Section IV) image affected with these types of distortions are given in Table 4.

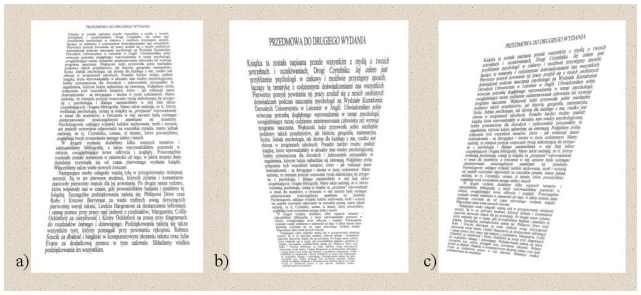


Fig. 11. Perspective distrortions considered in the tests: a) trapezoid, b) reverse trapezoid, c) rhomboid.

TABLE 4

OCR ERRORS AFTER PERSPECTIVE DISTORTIONS

distortion	correction method	error [%]	notes
trapezoid	–	0.70	–
trapezoid	our alg.	0.07	–
rev. trapezoid	–	–	recogn. area misdetected
rev. trapezoid	our alg.	0.11	–
rhomboid	FR's auto	–	recogn. area misdetected
rhomboid	our alg.	0.03	–

Non-linear distortions were also added programmatically. Table 5 refers to the three cases showed in Fig. 12 (a, b, c).

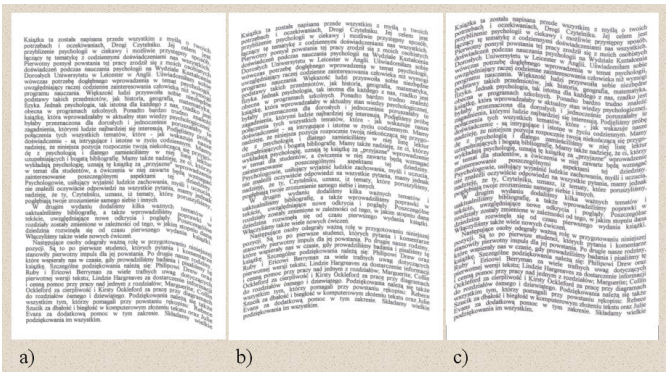


Fig. 12. Non-linear distrortions

TABLE 5

OCR ERRORS AFTER NON-LINEAR DISTORTIONS

distortion	correction method	error [%]	notes
Fig. 12a model	–	–	recogn. area misdetected
Fig. 12a model	our alg.	0.11	–
Fig. 12b model	–	–	recogn. area misdetected
Fig. 12b model	our alg.	0.14	–
Fig. 12c model	–	–	recogn. area misdetected
Fig. 12c model	our alg.	0.14	–

It is clear from Tables 4 and 5 that FR 7.0 cannot operate on images affected with these types of distortion. On the contrary, our algorithm removes those distortions almost perfectly (practically no accuracy loss in the further OCR stage).

The final experiment was probably the most valuable one. We preprocessed a real photograph taken from a digital camera (Fig. 13), presenting an open book. The image resolution was 2288 x 1712 px. The text to recognize was in Polish language, it contained 1651 characters (excluding whitespaces). The image contained several defects: wrong perspective, 90º right rotation, non-linear geometrical distortions, bad contrast, smudges in the background (paper).

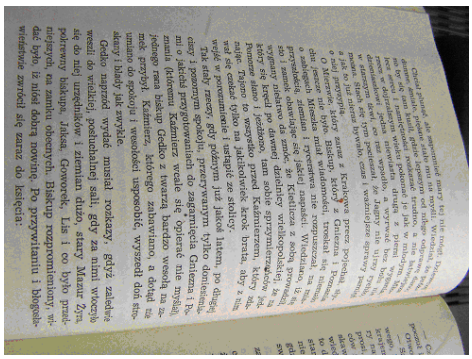
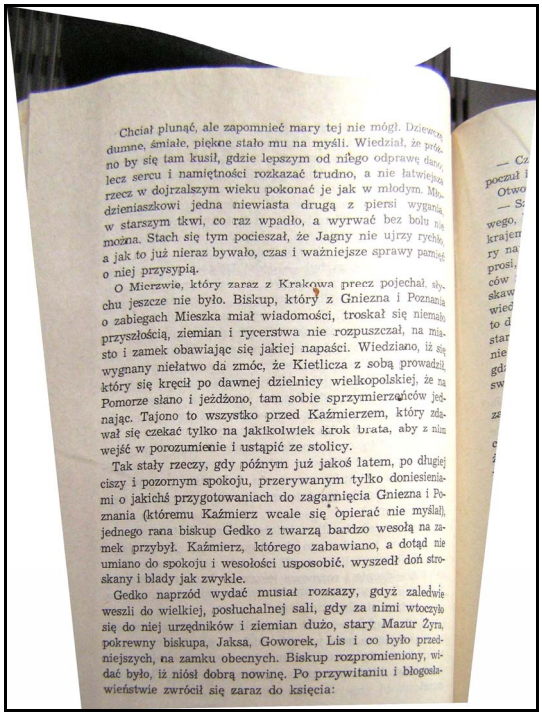


Fig. 13. Photograph of an open book. Many defects visible

The correction procedure contained the following steps (in this order):

- 90º left rotation,
- perspective correction,
- non-linear distortion correction,
- image intensity adjustment.

The output result is presented in Fig. 14.



Chciał plunąć, ale zapomnieć mary tej nie mógł. Dziewczę dumne, śmiałe, piękne stało mu na myśli. Wiedział, że prędzej no by się tam kusił, gdzie lepszym od niego odprawę dano, lecz sercu i namiętności rozkazać trudno, a nie łatwiejsza rzecz w dojrzalszym wieku pokonać je jak w młodym. Mło dzieniaszkowi jedna niewiasta drugą z piersi wygania, w starszym tkwi, co raz wpadło, a wyrwać bez bolu nie można. Stach się tym pocieszał, że Jagny nie ujrzy rychło, a jak to już nieraz bywało, czas i ważniejsze sprawy pamięć o niej przysypią.

O Mierzwie, który zaraz z Krakowa precz pojechał, słychu jeszcze nie było. Biskup, który z Gniezna i Poznania o zabiegach Mieszka miał wiadomości, troskał się niemało przyszłością, ziemian i rycerstwa nie rozpuszczał, na miasto i zamek obawiając się jakiej napaści. Wiedziano, iż się wygnany niełatwo da zmóc, że Kietlicza z sobą prowadził, który się kręcił po dawnej dzielnicy wielkopolskiej, że na Pomorze słano i jeżdżono, tam sobie sprzymierzeńców jednając. Tajono to wszystko przed Kaźmierzem, który zdawał się czekać tylko na jakikolwiek krok brata, aby z nim wejść w porozumienie i ustąpić ze stolicy.

Tak stały rzeczy, gdy późnym już jakoś latem, po długiej ciszy i pozornym spokoju, przerywanym tylko doniesieniami o jakichś przygotowaniach do zagarnięcia Gniezna i Poznania (któremu Kaźmierz wcale się opierać nie myślał), jednego rana biskup Gedko z twarzą bardzo wesołą na zamek przybył. Kaźmierz, którego zabawiano, a dotąd nie umiano do spokoju i wesołości usposobić, wyszedł doń stroskany i blady jak zwykle.

Gedko naprzód wydać musiał rozkazy, gdyż zaledwie weszli do wielkiej, posłuchalnej sali, gdy za nimi wtoczyło się do niej urzędników i ziemian dużo, stary Mazur Żyra, pokrewny biskupa, Jaksa, Goworek, Lis i co było przedniejszych, na zamku obecnych. Biskup rozpromieniony, widać było, iż niósł dobrą nowinę. Po przywitaniu i błogosławieństwie zwrócił się zaraz do księcia:

Fig. 14. Corrected image

TABLE 6

OCR ERRORS ON THE REAL DIGITAL CAMERA IMAGE

input image	correction method	error [%]	notes
original	–	–	text area not found
original	90° left rotat.	–	recogn. area misdetected
rotated	pers. corr.	2.72	–
rotated + pers. corr.	non-linear defect corr.	0.96	–
rotated + pers. corr. + non-linear defect corr.	intensity adjustment	0.88	*imadjust* ([0 0.8] , [0 1])

FineReader 7.0 was unable to process the original input image, most importantly due to perspective distortions. As Table 6 shows, with each successive algorithm layer, the accuracy improves, to reach less than 1% error.

VII. CONCLUSION

The optical character recognition is an active research area, but most efforts concern the actual recognition, not the preprocessing phase. In this work, we showed that appropriate image preprocessing for OCR is extremely important especially for digital camera acquired images. Non-professional digital cameras are rapidly becoming the main source of image information, also in case of text scanning. We predict that the sort of algorithms proposed in this paper will be embedded in the future versions of OCR software.

REFERENCES

[1] Bubliński Zb., Mikrut Zb.: *Lokalizacja tablic rejestracyjnych pojazdów*. Automatyka, t. 7, zeszyt 3, Uczelniane Wydawnictwa Naukowo-Dydaktyczne, Kraków 2003, pp. 303–312 (in Polish).

[2] Thillou C., Gosselin B.: *Robust thresholding based on wavelets and thinning algorithms for degraded camera images*. Proceedings of ACIVS 2004, (2004).

[3] Mao S., Rosenfeld A., Kanungo T.: *Document structure analysis algorithms: a literature survey*. Proc. SPIE Electronic Imaging. Jan. 2003; 5010: 197–207.

[4] Tanghva K., Borsack J., Condit A.: *Evaluation of model-based retrieval effectiveness with OCR text*. ACM Transactions on Information Systems, 14(1): 64–93, Jan. 1996.

[5] Kauniskangas H.: *Document image retrieval with improvements in database quality*. Oulu University Library, ISBN: 951-42-5313-2. ISBN: 951-42-5313-2, 1999.

[6] Bieniecki W.: *Analiza wymagań dla metod przetwarzania wstępnego obrazów w automatycznym rozpoznawaniu tekstu*. Automatyka, t. 9, zeszyt 3, Uczelniane Wydawnictwa Naukowo-Dydaktyczne, Kraków 2005, pp. 525–532 (in Polish).