# Document Image Dewarping using Robust Estimation of Curled Text Lines

Adrian Ulges
U. of Kaiserslautern
Kaiserslautern, Germany
ulges@iupr.net

Christoph H. Lampert
DFKI GmbH
Kaiserslautern, Germany
chl@iupr.net

Thomas M. Breuel
DFKI GmbH
Kaiserslautern, Germany
tmb@iupr.net

## Abstract

*Digital cameras have become almost ubiquitous and their use for fast and casual capturing of natural images is unchallenged. For making images of documents, however, they have not caught up to flatbed scanners yet, mainly because camera images tend to suffer from distortion due to the perspective and are therefore limited in their further use for archival or OCR. For images of non-planar paper surfaces like books, page curl causes additional distortion, which poses an even greater problem due to its nonlinearity.*

*This paper presents a new algorithm for removing both perspective and page curl distortion. It requires only a single camera image as input and relies on a priori layout information instead of additional hardware. Therefore, it is much more user friendly than most previous approaches, and allows for flexible ad hoc document capture.*

*Results are presented showing that the algorithm produces visually pleasing output and increases OCR accuracy, thus having the potential to become a general purpose preprocessing tool for camera based document capture.*

## 1 Introduction

Due to the rapid development of digital photography, camera based document capture has become a simple and flexible alternative to conventional flat bed scanners. Unfortunately, text in the images delivered by digital cameras often is strongly distorted, and the images cannot simply be used for further processing like image-based document management, database storage or OCR, as this is the case for images scanned e.g. by a flatbed scanner.

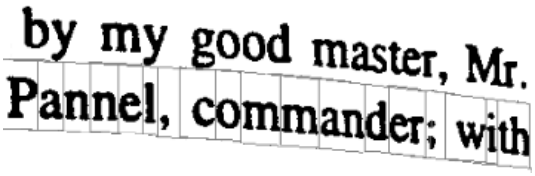
In this paper, we present an image dewarping algorithm that removes this distortion and can therefore be used to enhance picture quality, helping to improve all subsequent processing steps. In particular, OCR yields much better results on the undistorted images, so that our method might be an interesting preprocessing step to be included into existing OCR software packages.

In general, to successfully dewarp images of non-planar documents like books, one needs quantitative information on two major phenomena, namely the distortion due to perspective and the distortion due to non-linear page curl. For a general camera image, both are unknown. To overcome this lack of information, several methods have been proposed, most of them based on the acquirement of depth information in addition to the document image.

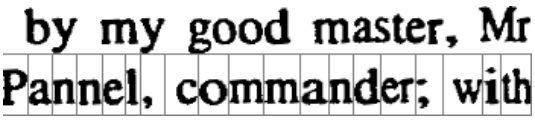
For the special case of documents scanned with flatbed scanners, depth-from-shading techniques have proven useful [8]. However, for images taken by cameras, the lighting conditions are too unpredictable for this, and instead structured light sources [2], lasers scanners [4], and stereo vision (using images by two cameras instead of just one) [5, 7] have been used to reconstruct the shape of the page. All these setups share the disadvantage that they need more or specialized hardware, often require calibration, and therefore reduce the flexibility in taking the pictures. Since its flexibility is the major advantage of camera based document capture, this is a major drawback.

In contrast, we follow the approach to use only one camera image and combine it with a priori side information about document images, namely the presence of text lines which are known to have derived from straight lines on the page surface of the book. This alone however is not sufficient, because straightening of text lines can only deal with distortion in normal direction to the text lines, not tangent to it. It has been tried to estimate the remaining free parameter heuristically [6], or based on parallelism of text lines [3]. Our approach is to rely on the additional fact that line spacing usually is constant all over the document, and that the page surface is smooth, i.e. that and not e.g. torn or wrinkled. All this information is combined to derive depth information on the book surface.

The main practical obstacle is then to reliably track the position and angles of text lines. This is easily underestimated as a problem, since the concept of tracking text lines is very intuitive to a human. However, for western languages the in principle straight base lines of letters are often interrupted by descenders and punctuation, and it turns




(a) Chain of cells along a text line in the input image




(b) Chain of cells along a text line in the output image

**Figure 1. Corresponding text lines before and after the dewarping. The parameters for input box slope and output box width have to be determined for each cell.**

out that simple linear or spline interpolations are too error-prone to be of use in the context of document image dewarping.

## 2 Our approach

Our algorithm performs a line-by-line dewarping of the observed paper surface. Each letter in the input image is enclosed within a quadrilateral cell, which is then mapped to a rectangle of correct size and position in the result image. In Figure 1, the shape of such cells is illustrated.

This construction has two fundamental unknowns: First, to determine the shape of cells in the input image, we need the local slope of the text lines. This information can be extracted from the input image alone. Our method of choice for this is the RAST algorithm, as will be explained later.

Second, the proper width of each destination box in the output image has to be determined. Ideally, it should be equal to the width of the original letter on the printed surface, which, however, is unknown and varies strongly between different characters. Also, the correct width is not necessarily similar to the size of the letter in the input image because of perspective effects.

To overcome this problem, we analyze the camera image with regard to a priori layout information. The underlying idea of this process is that objects or features of equal size appear smaller in the camera image with increasing distance. Consequently, image size information of equal-size features allows for the estimation of depth values. Our feature of choice is line spacing: we assume that line spacing is uniform all over the original document, which is a well justified assumption for many classes of printed documents. In the camera image, however, the distance between adjacent text lines varies due to the perspective, decreasing with increasing distance of the paper surface from the camera.

As result, we obtain a $3D$-model that provides the position and orientation information of each character on the paper surface. From this information, the correct box width can easily be derived. In the following, the individual steps are explained in more detail.

### 2.1 Preprocessing

Preprocessing of the input images starts with an adaptive binarization step. For each pixel, the background intensity $B(p)$ is defined as the $0.8$-quantile in a window-shaped surrounding. The pixel is then classified as background if its intensity is above a constant fraction of $B(p)$. In the binarized image, we identify connected components $C = \{c_1, .., c_n\}$ which we will also refer to as 'letters' in the following, although strictly speaking (e.g. in the case of ligatures or due to binarization errors) they might not contain exactly one letter.

Afterwards, we scan through all boxes to partition $C$ into global text lines. For each letter, the most likely right successor $r_i$ is determined based on distance and overlap between bounding boxes. Two letters are considered to be part of the same text line exactly if they belong to the same tree in the resulting forest $(C, \{(c_i, r_i)\})$.

### 2.2 Local text line approximation

It is crucial for the performance of our approach to very precisely compute the local distances between adjacent base lines in the image. Therefore, the curved geometric shape of the text lines must be estimated to subpixel accuracy. Especially when observing western fonts, this includes that the real base line must be identified robustly against descender letters like 'p' and 'g'.

We achieve this estimation using the RAST algorithm, a fast and flexible method for robust geometric model fitting. Previously, RAST has been used to estimate straight, global text lines in scanned documents [1]. Given a set of character bounding boxes, RAST finds an optimal base line for them, taking into account possible descenders. RAST is known to yield optimal solutions in this situation by exhaustively searching the whole parameter space.

Since in our case text lines are curved, we adapt RAST for a local text line search: For each character $c_i$ a set of neighboring letters within a box-shaped image region around $c_i$ is identified and a RAST search is performed using this local neighborhood only. The result is a local linear text line approximation that passes through the base point of the letter $c_i$. Figure 2 shows how a RAST result for the

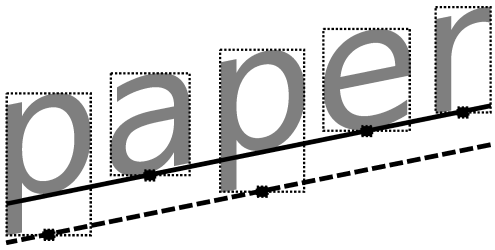



**Figure 2. For a given set of bounding boxes, the RAST algorithm determines the optimal slope parameter of the text line, taking into account that descenders may be present which do not lie the baseline itself, but on a parallel 'line of descenders'.**

central 'p' in the word 'paper' would look like. Note that the line of descenders is correctly identified, whereas a simple approach using splines or least squares regression would result in the line to be drawn down, towards the descender base points, and therefore yield a wrong result.

Using the local slope information, we build quadrilateral cells around the letters in the input image as seen in Figure 1(a). The upper and lower sides of the cells are built from the local base lines in the current resp. previous text line, shifted down by the distance to the line of descenders. The left and right sides are verticals in the center between the neighboring bounding boxes. This way, letters are never cut apart by cell edges, and each cell contains only one letter.

### 2.3 Depth extraction

The local text line approximations delivered by RAST are also used to estimate the line spacing $l_i$ at each character position in the image, by measuring the distance of the character's base point to its top and bottom neighbors. Figure 3 illustrates how a depth value can be derived from the line spacing: assuming objects of constant size (line spacings, in our case) and orientation, their observed size depends on their distance $d_i$ from the focal point of the camera. More precisely, perspective projection yields

$$\frac{f}{d_i} = \frac{l_i}{h}, \tag{1}$$

where $f$ denotes the focal length of the camera and $h$ the object height. Thus, we obtain the object distance as

$$d_i = K \cdot \frac{h}{l_i}, \tag{2}$$

where $K$ is a constant depending on the focal length.
In this way we obtain a depth value $d_i$ for every letter, which

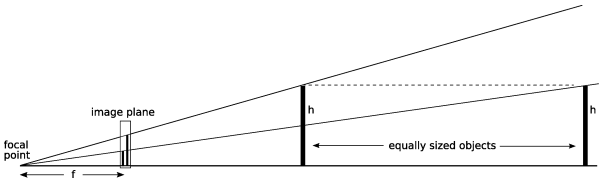



**Figure 3. The size at which two objects of same height are perceived depends inverse linearly on their distance to the observer. The right bar at twice the distance of the left, appears half as big.**

can directly be derived from the local line spacing $l_i$ once $h$ is known. In our algorithm, we estimate $h$ as the average of $l_i$ over all characters. The constant $K$ then has to absorb an additional factor depending on the pixel size of the camera, since $h$ is measured in pixel units.

Due to binarization, line partitioning, and RAST mismatches, outliers can appear in the depth field and have to be filtered out. We use a robust least squares fit in the surrounding of each point, rejecting a point as an outlier if its distance to the fitted plane is larger than a threshold. To compensate for the fact that page curl usually is much stronger in the $x$-direction than in the $y$-direction, we choose an anisotropic neighborhood with larger extent in $y$.

### 2.4 Building a 3D model to estimate cell widths

Using the set of scattered depth values, each corresponding to a character on the page, we obtain $3D$ points using the general principles of perspective projection:

An image point $p$ with pixel coordinates $(u, v)$ and depth value $d$ is back projected onto a 3D-point

$$p' = (\lambda \cdot (u - u_0), \lambda \cdot (v - v_0), d)^t \in \mathbf{R}^3 \tag{3}$$

where $(u_0, v_0)$ is the principle point of the camera, which can be approximated well by just choosing the center point of the camera image, and $\lambda = \frac{l_i}{h}$ again compensates for the perspective shortening.

As described before, we want to use the original width of each character $c_i$ on the curled surface to dewarp the page. This width corresponds to the $3D$-Euclidean distance between the left and right base corner of the letter cell, measured in pixel units. Since this is just the scale we have chosen for building our $3D$-model, we can read off the distance from there, interpolating the position of the left and right bottom corner of the cell halfway between the base points of the cell itself and its left and right neighbors.

Using the width, we can generate a properly dewarped chain of cells in the output image as illustrated in Figure 1(b). Texture warping is then carried out cell-by-cell using bilinear interpolation.

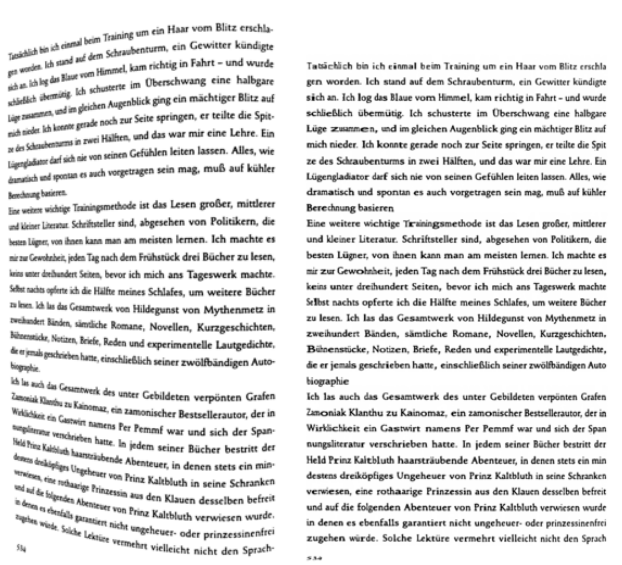


**Figure 4. Page image before and after dewarp (binarized). In the top and bottom rows, text lines are distorted most, and the curl is also gets stronger near the spine (right boundary).**

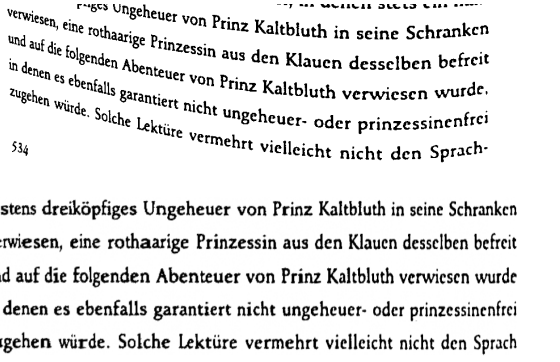


**Figure 5. Enlarged crop of the bottom part of the page, before and after the dewarp. The compensation must correct for the perspective shortening and the curvedness of text lines.**

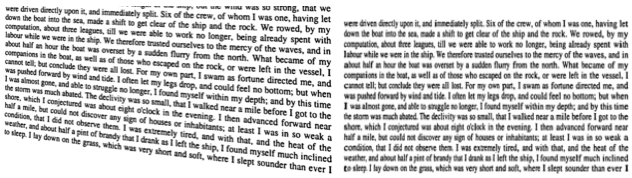


**Figure 6. Image of a planar page before and after dewarp (detail). Only perspective distortion must be removed.**

## 3 Results

To test the performance of our algorithms, we implemented it in the C++ language as a set of command line tools. We then applied it to several images of documents, taken with ordinary consumer digital cameras and showing different kinds of distortion. Since our algorithm so far is only suitable for pages containing single columns of text, we only used documents of that type. All experiments were done on an ordinary 2GHz PC running the Linux operating system. The current version, which is not optimized for speed in any way, takes around 10 seconds to process one page.

Our main target is to dewarp book surfaces, therefore our emphasis in presenting results lies on those. A typical example of such a book image before and after the dewarp is presented in Figure 4. To allow easier visual comparison, we always show the input images after the binarization step. Figure 5 shows an enlargement of this page, from a part close to the bottom of the page where the page curl is especially strong.

Since our method does not rely on any shape model, it works for images of any object that possesses parallel text lines. To demonstrate this, we applied our algorithm to images showing other kinds of distortion that also are frequently studied in the literature: planar documents, where the distortion is only due to the perspective, and book pages scanned in with a flatbed scanner, where only the area close to the spine is distorted due to the inflexible binding. Typical results are shown in figures 6 and 7.

Finally, we tested in how far our method increases OCR accuracy. We created printouts of known ASCII contents in different font styles and sizes. The documents were bent into typical book shape and images were taken the same way as for the real books. On the resulting images, optical character recognition was performed using the commercial OCR software *ABBYY FineReader 7.0*, and the number of words correctly identified was used to measure OCR performance for the input image and the dewarped image. As reference, we used the original ASCII version of the text.

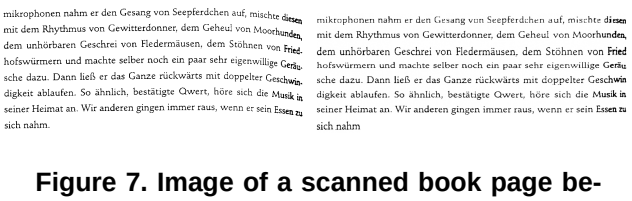


**Figure 7. Image of a scanned book page before and after dewarp (detail). Only close the spine distortion must be removed.**

| input font | error rate on input | error rate on output |
|---|---|---|
| times 9pt | 17.9% | 1.6% |
| times 12pt | 7.2% | 1.5% |
| arial 9pt | 14.5% | 0.8% |
| arial 12pt | 10.7% | 0.0% |
| **average** | **12.6%** | **1.0%** |

**Figure 8. OCR error rates of the FineReader OCR software before and after dewarping.**

Table 3 contains the results for the fonts Arial and Times at font sizes of 9pt and 12pt.

It is however difficult to compare these results with other approaches, since in the field of document image dewarping, no standard database of document images exists so far.

## 4 Discussion

As can be seen from the images presented in the previous section, our algorithm creates output that is visually very pleasing. This is because our main concept is to straighten text lines, and human perception is very sensitive to recognizing straight lines. We therefore believe that our decision to use 1D text lines as underlying geometrical objects of our dewarping approach (contrary e.g. to energy minimizing approaches using 2D-meshes [2, 4]) is justified.

In the closeup it is visible that individual characters are not always restored correctly into upright shape. We believe that apart from imperfect masking of outliers, this is mainly due to the fact that we have chosen the left and right box boundaries in the dewarping step as verticals, which makes it impossible to correct for a slant in $y$-direction. Our plan is to take this into account in a following version. However, this is not a trivial task, since it requires solving the problem of determining which point on a neighboring text line corresponds to a point in the current, and therefore the integration of a partial differential equation.

The result when applying the algorithm to documents where only perspective distortion or only local distortion is present are similar to the general ones: The output is of good visual quality, although not always all distortion is removed. Note that our algorithm is not meant to replace special purpose algorithms in these situations, but as a more general concept that works well also in these cases.

Finally, the result of the OCR experiments encourage us most to continue working in the direction we have taken. It is obvious from the data that a preprocessing step is needed before images captured by cameras can be processed by OCR software. For the unprocessed images, whole sections of text were detected incorrectly or not at all, leading to error rates that make the whole process of OCR useless. The dewarped output showed acceptable performance all over the document, with the remaining errors more often due to bad binarization than to the page curl.

## 5 Conclusion

In this paper we have presented a complete algorithm to remove distortion due to perspective and page curl from images of smooth but non-planar book pages. Contrary to previous approaches, the only data it needs is one camera image and the knowledge that the original document contains parallel text lines of fixed line spacing.

We have shown that our algorithm removes the distortion from different kinds of documents very reliably. Furthermore, the OCR error rate is reduced by up to 90%, which makes it possible to practically perform character recognition on images of curled documents without spending too much time on correcting errors by hand.

As a main future improvement, we plan to incorporate a more general model for the dewarping step itself, and also combine the algorithm with a step of layout analysis, thereby enabling the method to also work on multicolumn documents and more difficult page layouts.

## References


[1] T. M. Breuel. Robust least square baseline finding using a branch and bound algorithm. In *Proceedings of SPIE/IS&T 2002 Document Recognition & Retrieval IX Conf. (DR&R IX)*, pages 20–27, San Jose, California, USA, January 2002.

[2] M. S. Brown and W. B. Seales. Document restoration using 3d shape: A general deskewing algorithm for arbitrarily warped documents. In *International Conference on Computer Vision (ICCV01)*, volume 2, pages 367–374, July 2001.

[3] H. Cao, X. Ding, and C. Liu. Rectifying the bound document image captured by the camera: A model based approach. In *Seventh International Conference on Document Analysis and Recognition - ICDAR2003*, pages 71–75, 2003.

[4] M. Pilu. Deskewing perspectively distorted documents: An approach based on perceptual organization. *HP White Paper*, May 2001.

[5] A. Ulges, C. Lampert, and T. M. Breuel. Document capture using stereo vision. In *Proceedings of the ACM Symposium on Document Engineering*, pages 198–200. ACM, 2004.

[6] C. Wu and G. Agam. Document image de-warping for text/graphics recognition. In *Proc. of Joint IAPR 2002 and SPR 2002 Windsor*, pages 348–357, 2002.

[7] A. Yamashita, A. Kawarago, T. Kaneko, and K. T. Miura. Shape reconstruction and image restoration for non-flat surfaces of documents with a stereo vision system. In *Proceedings of 17th International Conference on Pattern Recognition (ICPR2004), Vol.1*, pages 482–485, 2004.

[8] Z. Zhang, C. L. Tan, and L. Fan. Restoration of curved document images through 3D shape modeling. In *International Conference on Computer Vision and Pattern Recognition (CVPR2004)*, pages 10–15, June 2004.