Dense Correspondences and Ancient Texts

Tal Hassner, Lior Wolf, Nachum Dershowitz, Gil Sadeh, and Daniel Stökl Ben-Ezra

Abstract This chapter concerns applications of dense correspondences to images of a very different nature than those considered in previous chapters. Rather than images of natural or man-made scenes and objects, here, we deal with images of texts. We present a novel, dense correspondence-based approach to text image analysis instead of the more traditional approach of analysis at the character level (e.g., existing optical character recognition methods) or word level (the so called *word spotting approach*). We focus on the challenging domain of historical text image analysis. Such texts are handwritten and are often severely corrupted by noise and degradation, making them difficult to handle with existing methods. Our system is designed for the particular task of aligning such manuscript images to their transcripts. Our proposed alternative to performing this task manually is a system which directly matches the historical text image with a synthetic image rendered from the transcript. These matches are performed at the pixel level, by using SIFT flow applied to a novel per pixel representation. Our pipeline is robust to document degradation, variations between script styles and nonlinear image transformations. More importantly, this per pixel matching approach does not require prior learning of the particular script used in the documents being processed, and so can easily be applied to manuscripts of widely varying origins, languages, and characteristics.

T. Hassner (✉)
Department of Mathematics and Computer Science, The Open University of Israel, Raanana, Israel
e-mail: hassner@openu.ac.il

L. Wolf • N. Dershowitz • G. Sadeh
Tel Aviv University, Tel Aviv, Israel
e-mail: wolf@cs.tau.ac.il; nachum@cs.tau.ac.il; gils@cs.tau.ac.il

D. Stökl Ben-Ezra
École Pratique des Hautes Études, Paris, France
e-mail: daniel.stoekl@ephe.sorbonne.fr

T. Hassner, C. Liu (eds.), *Dense Image Correspondences for Computer Vision*, DOI 10.1007/978-3-319-23048-1_12

1 Introduction

Recent large scale digitization and preservation efforts are producing vast image collections of historical manuscripts. Such manuscripts provide important records and artifacts from our shared cultural heritage. Taking European history as an example, close to one million manuscript books along with countless archival documents have survived to modern times from a period stretching back for over a millennium. Taken together, these documents serve as valuable sources of history, literature, philosophy, and scientific and medical literature, as well as art history (considering illuminations often present in such manuscripts). They are additionally important subjects of scientific inquiry in their own right, as they reflect scribal and monastic culture, the history and development of writing systems, languages and dialects, and the evolution of texts over time. Although efforts to digitally store these manuscripts provide new means for both safeguarding the information buried in them and making them widely accessible, searching through such manuscript image archives remains a challenge.

Manuscript images, unlike printed text images, can be extremely difficult to read by anyone other than experts skilled with a specific script or language type. They are often written in old languages and hence training computer systems to recognize or process them is challenging due to limited access to relevant training data. Further limiting their accessibility are abbreviations and scribal signs, commonly used by scribes. Many manuscripts have been severely degraded in quality over the years and are often corrupted by dirt and moisture. The text itself often fades, requiring specialized, sensitive imaging systems to capture. In other cases, ink may bleed through from one side of a parchment to another, making it difficult to differentiate between different texts on the same page. These and other concerns have made optical character recognition (OCR) of historical documents notoriously difficult [10].

All these challenges, as well as many more, are reflected in some of the most valuable manuscript collections, recently digitized and made available online. Some examples of these include the Dead Sea Scrolls (www.deadseascrolls.org.il), Greek papyri (www.papyrology.ox.ac.uk/Ancient_Lives), Codex Sinaiticus (codexsinaiticus.org), some of the Cairo Genizah documents (www.genizah.org), much of the Tibetan Buddhist Canon (www.tbrc.org; idp.bl.uk), Taiwanese deeds and court papers in the Taiwan History Digital Library (thdl.ntu.edu.tw), medieval Latin and English manuscripts (scriptorium.english.cam.ac.uk/manuscripts), the Early English Laws project (www.earlyenglishlaws.ac.uk), and the Papers of George Washington (rotunda.upress.virginia.edu/founders/GEWN.html) and many others. Our goal is to facilitate searching such manuscript archives, by proposing a system for the task of determining letter-by-letter mappings between transcription texts and their matching image regions in scanned manuscripts. By doing so, we provide access on a character level to these manuscript images. To our knowledge, no fully automatic method has previously been described for this task.

In previous chapters of this book, dense correspondences were used to transfer semantic information (e.g., depth, segmentation, scene labels) from reference examples to query images. Here, the same approach is used to transfer character labels from a synthetically rendered reference image to the manuscript image. We describe a system which is general in the sense that it does not attempt to learn how to identify graphemes in the manuscript. By using a robust "optical-flow" technique to directly match the pixels of the historical image with a synthetic image created from the text, it also avoids the assumption that one is provided with letter segmentation, a challenge in itself, particularly in images of historical texts. Instead, by transferring the (known) letter labels of pixels in the reference image to those in the historical document image, the extents of each letter (i.e., its segmentation) are obtained as part of our output.

The capabilities of our system were tested on images of a number of manuscripts from a range of scripts, writing directions, writing styles, and languages. As a possible extension to our method, we discuss how manual corrections of false correspondences can be used to improve the correspondence estimation quality from one line to the next. Beyond the specific application discussed here, our method provides an idea of the potential capabilities of a per pixel, correspondence-based approach in handling even extremely challenging text images.

This chapter is based on work which has previously been published in [11, 30].

2 Previous Work

Although previous work exists on the subject of matching text with images of the same text [13, 34], this problem has received very little attention compared to related problems of automatic text processing. In broad terms, existing systems take one of the following general approaches to this problem. OCR can be applied to the manuscript image in order to automatically extract the text appearing in it. Following character recognition, alignment of text, and image is a straightforward task. Although a number of effective, commercial quality OCR systems exist, applying them to manuscript images is well known to be extremely challenging (see, e.g., [3, 5, 28, 36] and many others).

One of the problems faced by OCR systems when applied to historical documents is the challenge of segmenting individual letters in texts. To avoid this problem, some have proposed systems which learn to recognize entire words rather than single characters. Some such examples include the systems of [15, 22] and more recently [2]. These systems all involve training on a collection of example word images. These are not available in many cases, particularly when examples from the same script and scribal hand do not exist.

Like us, some attempt to align text lines in their entirety. In [16, 19], as well as in other works, a sequence of word images and the transcript are treated as time series. Dynamic time warping (DTW) is used to align the sequence of word images

and its transcript. Alternatively, hidden Markov models (HMM) have been used to represent these sequences and then align them in [7, 29, 44]. These methods too require sufficient examples of similar texts in order to produce models tuned to the manuscripts being processed.

Character sizes, inter-character gaps, and other text punctuation characteristics have been modeled using geometric models, in order to improve the quality of the text segmentation, thereby providing better alignment quality. This approach has been applied, for example, on Japanese texts in [43] and Chinese texts in [40]. These methods are typically designed for a specific language and character properties, and it is not clear how to generalize them from one language to another, where these properties change.

The methods above are all fully automatic. A different approach which has been gaining in popularity is to offer manuscript scholars convenient software systems for transcript alignment. Several such tools include the text–image linking environment (TILE) [33], the UVic image markup tool project [14], and the TextGrid system [21]. Though these software tools can be of immense help towards cataloging the text in manuscript images, applying them to the huge collections of digitized manuscript images may be too labor intensive.

In contrast to the methods mentioned above, ours does not use expert knowledge of the properties of the language or typeface used by its characters. Furthermore, our method does not require training data to adapt to a manuscript type. Instead, we use direct, image-to-image, per pixel matching techniques between the manuscript image and a rendered image produced using a font sufficiently similar to the one used in the text. Though the quality of our alignments can depend on the suitability of this user-selected font, our system is designed to be robust even in cases when it is only a rough approximation to the typeface used by the scribe when writing the manuscript. Finally, compared to the software tools described above, our system is designed to be applied automatically, though it can be extended to allow for manual corrections. Moreover, when processing long manuscripts, images, manual corrections can be used to learn how to better estimate correspondences and improve alignment from one manuscript line to the next.

3 Method Overview

Our pipeline is illustrated in Fig. 1. For an input manuscript image and a text file containing its line-by-line transcript, our system performs the following steps. We begin processing the manuscript image by applying the line detection method of Wolf et al. [39] in order to obtain individual lines of text in the image. This method first binarizes the manuscript image and then projects the binary values onto the vertical axis. Peaks are then detected on these projected values in order to localize each line. Lines are then individually trimmed at their horizontal boundaries, detected by projecting the binary values of each line onto the horizontal axis (Fig. 1,

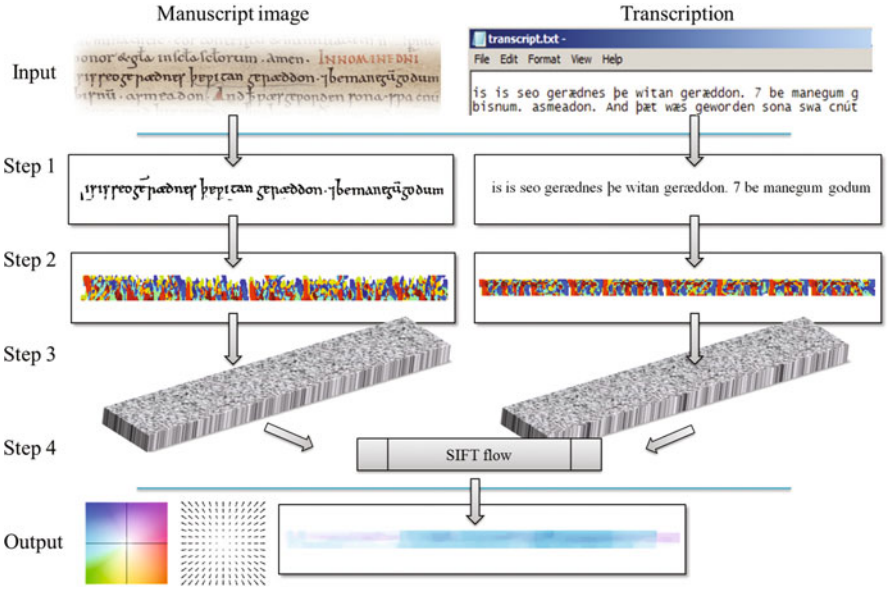

Fig. 1 The stages of our transcript alignment method. *Top row*: Our input is a historical manuscript image (*left*), along with a line-by-line transcript of the text in the image (*right*). Step 1, *left*: Each manuscript line is horizontally projected and trimmed; *right*, each matching line of the transcript is rendered using a suitable typeface, producing a matching reference image. Step 2: FPLBP codes [37] are computed for the pixels of each image (codes color coded). Step 3: Each pixel is represented by the frequency histogram of the codes in its elongated neighborhood. Step 4: Dense correspondences are established between these two histogram matrices, using SIFT flow [23]. *Bottom row*: the output flow (shown here color coded) matches each pixel in the manuscript line with a pixel in the rendered, reference image, providing access to its known character label

Step 1, left). The output of this step can then be manually adjusted using a graphical user interface designed for this purpose.

Once lines are isolated, our problem becomes that of assigning transcript character labels to each of the pixels of ink and of inter-word spaces in the line. To this end we render the matching transcript text line, producing a synthetic, reference image of the text (Fig. 1, Step 1, right). This is performed using a suitable font chosen for the purpose. Font selection is performed in order to produce a reference image in the same script and style (e.g., cursive, and print). In practice, the reference image fonts are not expected to be identical to those in the manuscript and, as we later show, can often be very different.

We maintain the spatial location of each character, as it is rendered onto the reference image. We therefore have a per pixel character label for all the reference image pixels. These labels are illustrated in Fig. 2 where a zoomed-in view of the reference image is shown above the pixel labels, visualized as colors. The transition from colder colors to warmer colors reflects the indices to the letters of the transcript; colder colors represent lower indices and warmer colors, higher indices.

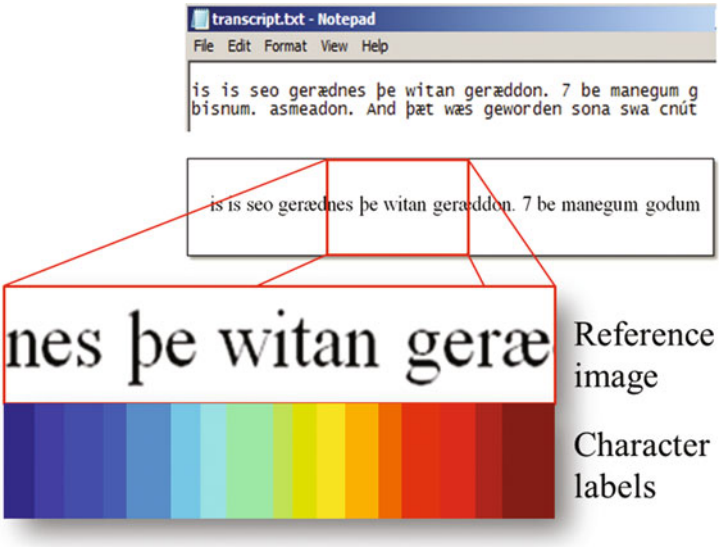

Fig. 2 Reference image and matching character labels. Focusing on Fig. 1, Step 1, *right*: A transcript line is rendered to produce a reference image along with a per pixel index to the characters in the original transcript. Index values are visualized here by *colors*. Note that spaces between words are also assigned with indices, matching them to the spaces appearing in the original transcript. Please see text for more information

Note in particular that spaces and punctuation marks are also labeled with indices to transcript letters.

Step 2 of our pipeline (Fig. 1) converts both the binarized manuscript image and the synthetic reference image to Four-Patch LBP (FPLBP) code images C and C', respectively. These FPLBP codes [37] are discussed in detail below (Sect. 4). This transformation converts each pixel to an integer value in the range [0..15]. In Step 3 of our pipeline, both code images are used to produce dense, per pixel descriptors for both images. This is accomplished by replacing the code assigned to each image with a histogram of the codes in the immediate neighborhood around the pixel. In order to capture the horizontal ambiguity of horizontal scripts, these histograms are computed using a vertical ellipse as their spatial support.

Finally, in Step 4 we use a robust dense correspondence estimation method, the SIFT flow of Liu et al. [23] in order to match the pixels of the two images. Where SIFT flow originally used Dense-SIFT (DSIFT) descriptors [35] to represent pixels, we replace it with the FPLBP code histograms produced in Step 3. Correspondences computed by SIFT flow from manuscript image to the reference image allow transferring the per pixel character labels of the reference back onto the manuscript, thereby providing the alignment from manuscript pixels to transcript letters. Key steps of our method are detailed in the following sections.

4 Pixel Encoding

4.1 Local Binary Patterns and Their Variants

Local binary patterns (LBP) [25–27] were originally designed as texture descriptors characterized by an invariance to monotonic photometric changes. Recent years have shown these representations to be highly effective in domains other than texture recognition, particularly in face recognition tasks[1]. The success of LBP motivated the development of LBP variants, which were later applied to a range of additional applications, including object localization [42] and action recognition [17]. The original LBP codes are computed at each pixel location by applying a threshold over the 3×3 neighborhood around the pixel, using the central pixel's intensity value as the threshold value. The resulting pattern of eight bits is then treated as a binary string and stored as an 8-bit number. An entire image is then represented by counting occurrences of every LBP code, in nonoverlapping regions of the image.

4.2 Four-Patch LBP Codes

We employ an LBP variant called Four-Patch LBP (FPLBP) [37]. Its design was motivated by a previous LBP variant called the center-symmetric LBP (CSLBP) [12]. CSLBP compares four pairs of intensities arranged in a circle around a central pixel. These four comparisons are represented as binary values reflecting which of the two pixels in each pair has the higher intensity value. FPLBP combines a similar circular sampling strategy, with the patch based approach of another LBP variant, Multi-Block LBP [41], which provides better spatial support for each comparison by sampling square patches, rather than pixels, and comparing their mean values.

Though motivated by these previous methods, FPLBP is a very different representation. Rather than encoding the relation between pixel or pixel-patch intensities, it uses short binary strings to encode local self-similarities [32] of pixel patches. FPLBP and the related Three-Patch LBP codes (TPLBP) [37] have been shown to capture valuable local information which complements the information reflected by the pixel-based descriptors, including the original LBP codes [38].

FPLBP encoding is performed by considering two concentric rings around each pixel. Each ring is sampled at S evenly distributed $w \times w$ pixel patches. Two opposing pairs consisting of an inner ring patch and an outer ring patch, separated by α patches between them, are then compared by evaluating the L2 distances between the patches in each pair. A single bit in the representation is set according to which of these two pairs holds more similar patches (i.e., has a smaller L2 distance between its two patches). The central pixel is then assigned with the $S/2$ bits resulting from these comparisons.

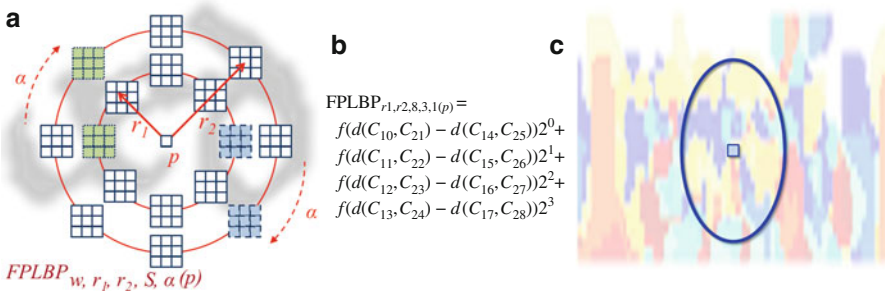

Fig. 3 (**a**) Visualizing how a single bit in the FPLBP code of pixel p is set (Step 2 of our pipeline, Fig. 1). Two patch pairs are compared: one in *dotted lines* and *green fill* and the other in *dashed lines* and *blue fill*. Patches in each pair are separated by $\alpha = 1$ patches. (**b**) Formal expression for FPLBP codes with parameters $S = 8$, $w = 3$, and $\alpha = 1$. The C_{ij} denote the various 3×3 patches; the first subscript indicates the ring (inner or outer) and the second denotes their location on the ring, from index 0 at twelve o'clock. (**c**) Visualizing the vertical ellipse support region used to construct the histogram of FPLBP codes (visualized in *washed out colors*) at each pixel

This entire process is summarized in Fig. 3a, b. It shows the standard case of using $S = 8$ patches, resulting in only four bits per pixel. Despite this small size, these codes have been shown to be extremely effective representations. In [8], for example, FPLBP codes were used for face recognition and were shown to perform nearly as well as the significantly more expensive collections of SIFT [24] descriptors.

4.3 Why FPLBP?

Our choice of using the FPLBP codes here requires justification, especially in comparison to the standard DSIFT [35] typically used with SIFT flow [23] or alternative LBP code variants.

The Three-Patch LBP (TPLBP), presented with the FPLBP codes in [37], is also a patch-based local binary pattern descriptor in which a central patch is compared to a pair of patches in a manner designed to capture local self-similarities. It sets the value of a code bit to 1 following a comparison of the similarity of a central patch to two patches, α patches away from each other on a ring around pixel p. Eight patches are distributed on this ring, resulting in eight comparisons and an eight-bit code or a number in the range [0..255].

Previous work has shown that TPLBP captures more information than FPLBP and produces superior representations when used for face recognition [37] and texture recognition [6]. Our experiments with both these descriptors have shown that this advantage is minor, providing only slightly better performance than SIFT flow using DSIFT. The original LBP codes perform worst than both. This is unsurprising, when considering that LBP is computed by comparing pixel pairs, rather than

patches, making them more sensitive to noise—a major problem when considering degraded manuscript images.

FPLBP, by comparison, has several appealing properties, making it ideal for the task considered here. First, it is substantially smaller than any of the other representations. As we will show next, this property is significant when extracting descriptors on a dense grid; other representations require storage and processing time that can quickly become unreasonable. Second, and more importantly, when computed for document images, the range of codes produced by TPLBP (and also LBP) is only partially represented in the codes computed in practice. This is due to the far more restrictive nature of such images, which results in a smaller local pattern variations. Thus, the comparisons performed by FPLBP of local appearances between pairs in left/right, top/down, and both diagonals suffice to capture meaningful information.

These observations are verified empirically in Fig. 4. It compares the variability of code values computed by TPLBP vs. FPLBP. Evidently, the TPLBP histogram is far more sparse than its FPLBP counterpart. It therefore uses different values more efficiently in order to capture appearance variations.

4.4 From Codes to Dense Descriptors

Both manuscript and reference images are converted to code images C and C' respectively (Sect. 4.2). We next convert the codes in these images to dense, per pixel representations (Fig. 1, Step 3). This is performed in a manner similar to the *explode* operator used by the distribution fields representation for object tracking in [31]. Specifically, in order to account for local displacements of pixel codes, each pixel is represented by the histogram of code frequencies in its immediate neighborhood.

When considering horizontal scripts, the uncertainty of code appearances is greater horizontally than vertically. This is because letters tend to preserve their vertical appearance (similar image features would likely appear above or below each other), whereas, the variability of letters appearing before or after an image location implies greater horizontal feature variability. To account for this, our histograms are produced by considering a vertical ellipse support region (Fig. 3c). Of course, when considering vertical scripts, this may conceivably be changed to reflect vertical uncertainty, though we did not perform such tests here. Additional uncertainty in code values is introduced by weighing the influence of codes further away from the central pixel. In practice, we use a 2D Gaussian filter, with sigmas 2.5 (horizontally) and 1 (vertically), for this purpose.

We produce the dense, per pixel representation from FPLBP code images as follows. First, each FPLBP code is converted to a 16D binary vector, with the value 1 in the cell indexed by the code itself. We then consider the 16 binary maps, representing each dimension of these binary vectors, taken over the entire image. Each such map is processed by applying the elongated Gaussian filter

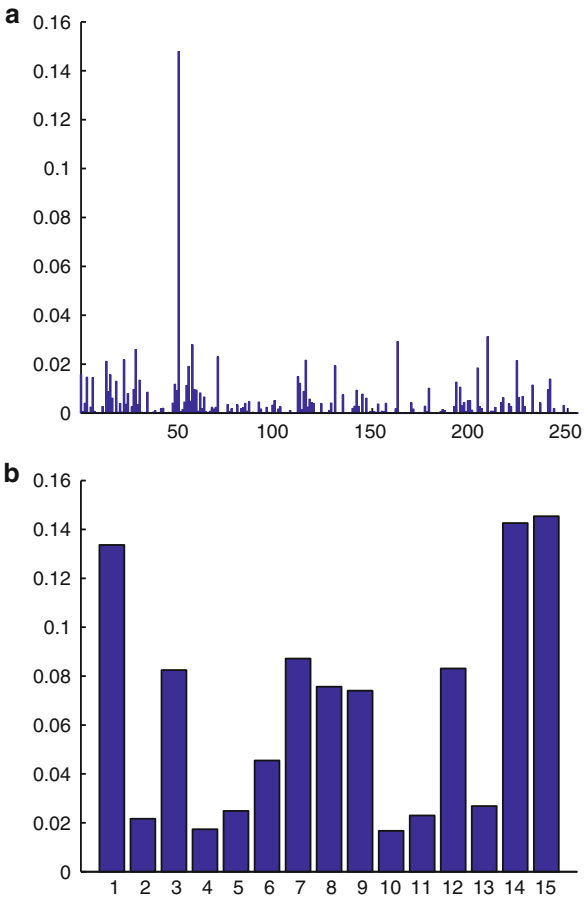

Fig. 4 (**a**) A histogram of TPLBP codes computed for a set of document images. (**b**) FPLBP code histogram computed for the same images, demonstrating a far more uniform distribution of code values

described above. This is performed efficiently by using the fast, integral image-based approximation to Gaussian filtering, described in [20]. Each pixel is then represented by its values following this filtering.

Given the two histogram fields produced by the process described above, we form dense correspondences between them using SIFT flow as a matching engine [23], applied to our own representation. Previous chapters describe SIFT flow in detail. Unlike other related methods (e.g., PatchMatch [4] and its variants [9, 18]), it explicitly seeks smooth correspondences with small displacements. These requirements reflect a desire to match similar regions of the manuscript image to the reference image and is therefore ideally suited for our purposes. Note that its complexity is dominated by the size of the per-pixel representation. Here, by using the FPLBP, the resulting representation is 16D, substantially smaller than other representations, allowing for faster processing.

5 Experiments

Experiments were performed to evaluate the effectiveness of our approach both qualitatively and quantitatively. Quantitative tests were performed using a synthetic data set produced in order to obtain an exact measure of the accuracy of our method, when applied to a wide range of varying fonts. Qualitative tests were performed on real data—historical documents from a wide range of sources.

5.1 Empirical Tests

We produced a synthetic data set from the Charles Dickens book, *A Tale of Two Cities*. We rendered a page from the book using all 274 standard fonts installed on our MS-Windows PC. This large collection of font types was then used to test the robustness of our method to differences between appearances of text in manuscript and reference images. With each rendered text we produced also the reference, ground truth character labels, similar to those visualized in Fig. 2. We scaled all fonts to the height of 19 pixels and the resulting image was at a resolution of 1140×716 pixels. The average width of a character was about nine pixels.

Our tests used a single reference image, which was the one produced using the Times New Roman font. All other fonts, excluding non-English graphemes such as Webdings, Wingdings, and Euclid Math, were used to represent manuscript images. In total, 269 query images were tested, each one providing 50 lines of text. These were processed automatically as described in Sect. 3.

Accuracy was empirically evaluated by considering the x, y pixel coordinates of the center of each letter. We measure the distance between the center of each manuscript image letter to the position of the corresponding reference letter following warp. We note that this error measure cannot be zero, even when perfect alignment is produced, since the center of mass of each character varies depending on the font. Still, this value is expected to decrease as the alignment quality improves. We report the mean ($\pm$SD) distance over all letters in a document as our alignment error rate.

Our results are summarized in Table 1. Besides the mean and standard deviation of the per document distances, we provide the median distance as well as the percent of manuscript fonts for which each method obtained the best results (smallest distances). We compare the following methods: A baseline method, in which text lines are linearly stretched or shrunk to match their extreme ends from reference to manuscript; the original SIFT flow using DSIFT as the per pixel representation; finally, our proposed method. Though we have also tested LBP-based and TPLBP-based representations, neither one was competitive with those reported here and so are excluded from the table.

We visualize our empirical tests in Fig. 5. It shows four example lines aligned using our method. All these examples were produced using the same reference font

Table 1 Summary of empirical results

Method	Baseline	SIFT flow	Proposed
Mean error ± SD	10.21 ± 4.2	8.38 ± 3.8	6.18 ± 3.1
Median error	11.23	7.42	5.27
Percent best error (%)	6	17	77

Each manuscript font is matched to the reference font using three tested methods. Accuracy was measured by considering the mean displacement error between letter centroids. The table reports mean (±SD) distance per method over all 269 fonts tested. Also provided are median displacement errors and the percentage of fonts for which each method performs best compared to all others. As a baseline we applied linear stretching or shrinking of the reference text to match the horizontal dimensions of the tested text

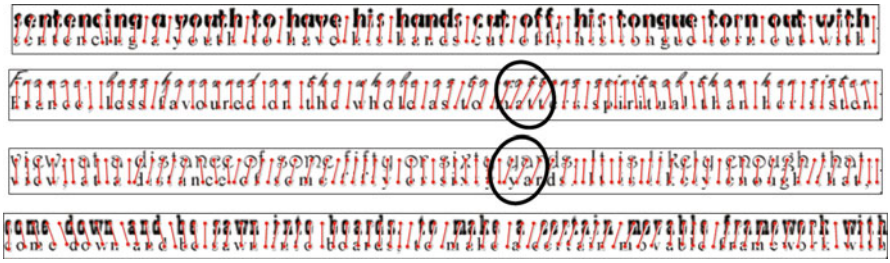

Fig. 5 Empirical tests visualized. Four examples from our empirical tests, produced using our system. Each example shows the synthetic (test) manuscript font on *top* and the reference at the *bottom*. Lines connect the center of each reference font character to its estimated position in the test font. Mistaken alignments are *highlighted* in the two middle examples. Note that by establishing per pixel correspondences, we obtain alignments for the spaces between letters as well as for punctuation marks

(bottom) and different test fonts, representing the manuscript. Lines connect the centers of reference characters to their estimated locations in the test lines of text. Evidently, for the most part, alignment succeeds, despite the differences in fonts used. Noteworthy are the links established between spaces and punctuation marks, and not just the letters themselves. We highlight a few alignment mistakes in the two middle examples.

5.2 *Qualitative Results*

We used our system to align manuscripts from a variety of sources, representing different languages and writing characteristics. Many such results are reported in [11].

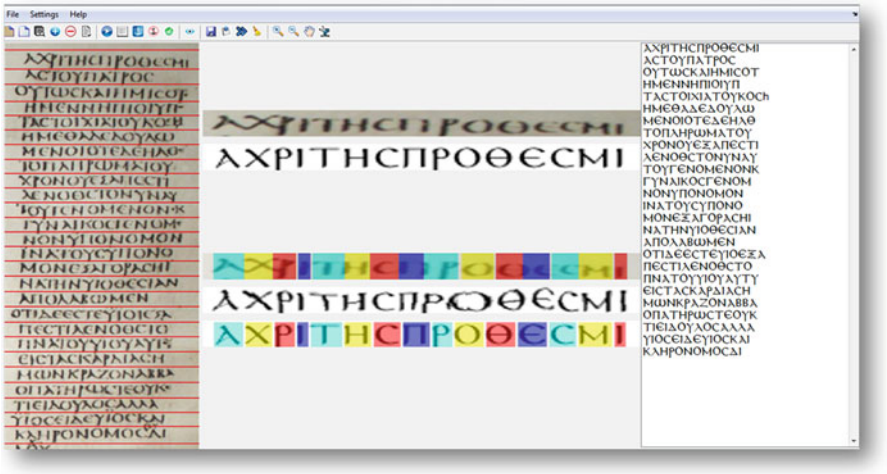

Fig. 6 Graphic user interface for transcript alignment. A page from the Codex Sinaiticus following processing of the first line using our graphical user interface. *Left panel*: The manuscript image. *Horizontal lines* denote automatically detected text rows. *Middle panel*: Alignment result shown for the current (first) line of text. *From top to bottom* are the current manuscript line; the rendered reference image produced for the manuscript line; the current manuscript line, color coded according to the character labels assigned through the correspondences; the synthetic reference image, warped according to the correspondences from manuscript to reference; finally, the synthetic reference image, color coded according to character labels. *Right panel*: Synthetically produced reference lines for the entire manuscript page

We have developed a graphical user interface in order to provide convenient means of loading manuscript and transcript images, as well as present alignment results. Beyond row-by-row application of the method described in this chapter, the interface allows human operators to correct alignment errors. Our system uses these corrections to train a letter spotting mechanism that provides manuscript-specific cues for alignment, improving the quality of alignment, and reducing the required manual labor, from one line to the next. These novel features are described in depth in [30].

A screen shot of our interface is provided in Fig. 6. It shows an alignment result for the first row of a page from Codex Sinaiticus. Alignment results are visualized in two ways. First, by warping the synthetic reference image, according to the inferred correspondences. Here, good results should warp the reference to match the appearance of the manuscript line, and in particular, reference letters should appear beneath their corresponding manuscript lines. Second, reference character labels are shown, color coded, over their matched manuscript image regions. The final result of aligning all text lines of the same page is provided in Fig. 7. An additional example in Hebrew MS Kaufmann A50 of the Mishnah is presented in Fig. 8.

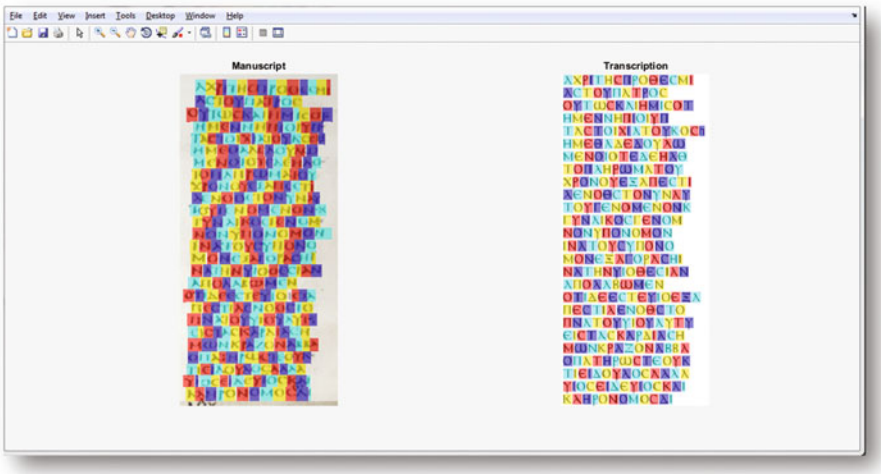

Fig. 7 Alignment result for an entire page from the Codex Sinaiticus. *Left panel*: The manuscript image with per character labels visualized by the graphical user interface as colors overlaid on image regions corresponding to each character. *Right panel*: Synthetically rendered reference rows with their per character labels visualized as colors overlaid on each character. Good results show the same color assigned to a transcript character on the right and to its corresponding manuscript character (image region) on the left

6 Conclusion

This chapter presents a dense correspondence-based solution to the very specific problem of transcript alignment. Despite the ostensibly narrow scope of the problem considered here, the method we present suggests far wider prospects for correspondence-based approaches in processing text images. Specifically, it marks a change from processing text documents on a per character or per word level to a per pixel level. It further demonstrates that with a suitable per pixel representation and a robust correspondence estimation engine, even challenging texts may be automatically processed. It remains to be seen if a similar approach can be applied to more traditional text processing applications. Unlike the settings considered here, these often do not have the privilege of a known transcript to match with the text image. Rather than relying on transcripts, future extensions can instead assume a fixed (possibly very large) vocabulary of known words and abbreviations using them as potential references.

Acknowledgements MS Kaufmann A50 by courtesy of the Oriental Collection of the Library and Information Centre of the Hungarian Academy of Sciences. This research was initiated at the Dagstuhl Perspectives Workshop 12382, “Computation and Palaeography: Potentials and Limits” and seminar 14302 on “Digital Palaeography: New Machines and Old Texts.”

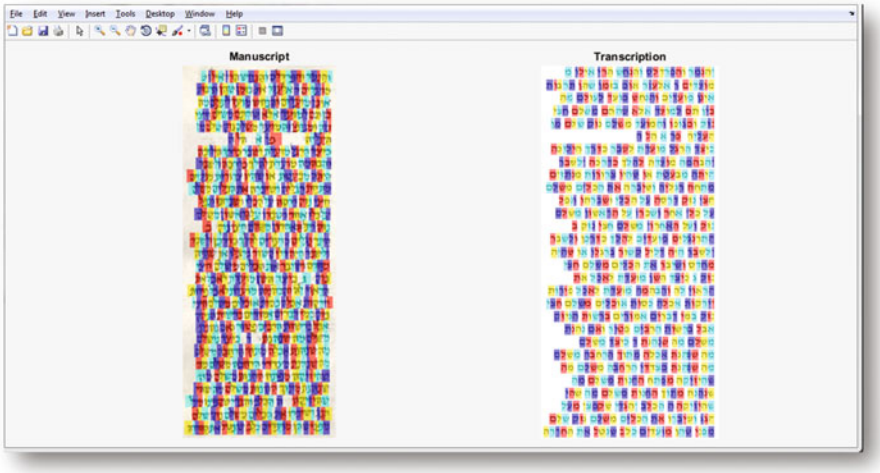

Fig. 8 Alignment result for an entire page from the MS Kaufmann A50 with the manuscript image (*left*) and the synthetically rendered characters of the transcription (*right*). Corresponding characters in the transcript and in the manuscript are labeled with similar color boxes

References

1. Ahonen, T., Hadid, A., Pietikainen, M.: Face description with local binary patterns: application to face recognition. IEEE Trans. Pattern Anal. Mach. Intell. **28**(12), 2037–2041 (2006)
2. Al Azawi, M., Liwicki, M., Breuel, T.M.: WFST-based ground truth alignment for difficult historical documents with text modification and layout variations. In: IS&T/SPIE Electronic Imaging, International Society for Optics and Photonics (2013)
3. Asi, A., Rabaev, I., Kedem, K., El-Sana, J.: User-assisted alignment of arabic historical manuscripts. In: Proceedings of Workshop on Historical Document Imaging and Processing, pp. 22–28. ACM, New York (2011)
4. Barnes, C., Shechtman, E., Goldman, D.B., Finkelstein, A.: The generalized PatchMatch correspondence algorithm. In: Proceedings of ECCV (2010)
5. Dovgalecs, V., Burnett, A., Tranouez, P., Nicolas, S., Heutte, L.: Spot it! Finding words and patterns in historical documents. In: Proceedings of International Conference on Document Analysis and Recognition, pp. 1039–1043. IEEE, New York (2013)
6. Ebert, S., Larlus, D., Schiele, B.: Extracting structures in image collections for object recognition. In: Proceedings of ECCV (2010)
7. Fischer, A., Frinken, V., Fornés, A., Bunke, H.: Transcription alignment of Latin manuscripts using hidden Markov models. In: Proceedings of HIP (2011)
8. Guillaumin, M., Verbeek, J., Schmid, C., Lear, I., Kuntzmann, L.: Is that you? Metric learning approaches for face identification. In: Proceedings of ICCV (2009)
9. HaCohen, Y., Shechtman, E., Goldman, D.B., Lischinski, D.: Non-rigid dense correspondence with applications for image enhancement. ACM Trans. Graph. **30**(4), 70:1–70:9 (2011)
10. Hassner, T., Rehbein, M., Stokes, P.A., Wolf, L.: Computation and palaeography: potentials and limits. Dagstuhl Manifestos **2**(1), 14–35 (2013)
11. Hassner, T., Wolf, L., Dershowitz, N.: OCR-free transcript alignment. In: Proceedings of International Conference on Document Analysis and Recognition, pp. 1310–1314 (2013)

12. Heikkilä, M., Pietikäinen, M., Schmid, C.: Description of interest regions with center-symmetric local binary patterns. In: Indian Conference Computer Vision, Graphics and Image Processing (2006)
13. Hobby, J.D.: Matching document images with ground truth. Int. J. Doc. Anal. Recognit. **1**(1), 52–61 (1998)
14. Holmes, M.: The UVic image markup tool project. Available: http://tapor.uvic.ca/~mholmes/image_markup (2008)
15. Huang, C., Srihari, S.N.: Mapping transcripts to handwritten text. In: Proceedings of the 10th International Workshop on Frontiers in Handwriting Recognition, pp. 15–20 (2006)
16. Jose, D., Bhardwaj, A., Govindaraju, V.: Transcript mapping for handwritten English documents. In: Yanikoglu, B.A., Berkner, K. (eds.) DRR, SPIE Proceedings, vol. 6815, SPIE (2008)
17. Kellokumpu, V., Zhao, G., Pietikainen, M.: Human activity recognition using a dynamic texture based method. In: Proceedings of BMVC (2008)
18. Korman, S., Avidan, S.: Coherency sensitive hashing. In: Proceedings of the IEEE International Conference on Computer Vision (2011)
19. Kornfield, E.M., Manmatha, R., Allan, J.: Text alignment with handwritten documents. In: Proceedings of Document Image Analysis for Libraries (DIAL), pp. 195–211. IEEE Computer Society, Cambridge (2004)
20. Kovesi, P.: Fast almost-Gaussian filtering. In: Proceedings of International Conference on Digital Image Computing: Techniques and Applications, pp. 121–125 (2010)
21. Kuster, M., Ludwig, C., Al-Hajj, Y., Selig, T.: Textgrid provenance tools for digital humanities ecosystems. In: Proceedings of Conference on Digital Ecosystems and Technologies Conference, pp. 317–323. IEEE, New York (2011)
22. Lavrenko, V., Rath, T.M., Manmatha, R.: Holistic word recognition for handwritten historical documents. In: Proceedings of Document Image Analysis for Libraries (DIAL), pp. 278–287 (2004)
23. Liu, C., Yuen, J., Torralba, A.: Sift flow: dense correspondence across scenes and its applications. IEEE Trans. Pattern Anal. Mach. Intell. **33**(5), 978–994 (2011)
24. Lowe, D.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. **60**(2), 91–110 (2004)
25. Ojala, T., Pietikainen, M., Harwood, D.: A comparative-study of texture measures with classification based on feature distributions. Pattern Recognition **29**(1), 51–59 (1996)
26. Ojala, T., Pietikäinen, M., Mäenpää, T.: A generalized local binary pattern operator for multiresolution gray scale and rotation invariant texture classification. In: Proceedings of ICAPR (2001)
27. Ojala, T., Pietikäinen, M., Mäenpää, T.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. Pattern Anal. Mach. Intell. **24**(7), 971–987 (2002)
28. Rabaev, I., Biller, O., El-Sana, J., Kedem, K., Dinstein, I.: Case study in Hebrew character searching. In: Proceedings of International Conference on Document Analysis and Recognition, pp. 1080–1084. IEEE, New York (2011)
29. Rothfeder, J.L., Manmatha, R., Rath, T.M.: Aligning transcripts to automatically segmented handwritten manuscripts. In: Bunke, H., Spitz, A.L. (eds.) Document Analysis Systems. Lecture Notes in Computer Science, vol. 3872, pp. 84–95. Springer, Berlin (2006)
30. Sadeh, G., Wolf, L., Hassner, T., Dershowitz, N., Ben-Ezra, D.S., Ben-Ezra Stökl, D.: Viral transcription alignment. In: Proceedings of International Conference on Document Analysis and Recognition (2015)
31. Sevilla-Lara, L., Learned-Miller., E.: Distribution fields for tracking. In: Proceedings of CVPR (2012)
32. Shechtman, E., Irani, M.: Matching local self-similarities across images and videos. In: Proceedings of CVPR, pp. 1–8 (2007). doi:10.1109/CVPR.2007.383198
33. Terras, M., Cayless, H., Noel, W.: Text-image linking environment (TILE). Available: http://mith.umd.edu/tile (2009)

34. Tomai, C.I., Zhang, B., Govindaraju, V.: Transcript mapping for historic handwritten document images. In: Frontiers in Handwriting Recognition, pp. 413–418 (2002)
35. Vedaldi, A., Fulkerson, B.: Vlfeat: an open and portable library of computer vision algorithms. In: Proceedings of International Conference on Multimedia, pp. 1469–1472 (2010)
36. Wei, H., Gao, G.: A keyword retrieval system for historical Mongolian document images. Int. J. Doc. Anal. Recognit. **17**(1), 33–45 (2014)
37. Wolf, L., Hassner, T., Taigman, Y.: Descriptor based methods in the wild. In: Post-ECCV Faces in Real-Life Images Workshop (2008)
38. Wolf, L., Hassner, T., Taigman, Y.: Effective unconstrained face recognition by combining multiple descriptors and learned background statistics. Trans. Pattern Anal. Mach. Intell. **33**(10), 1978–1990 (2011)
39. Wolf, L., Littman, R., Mayer, N., German, T., Dershowitz, N., Shweka, R., Choueka, Y.: Identifying join candidates in the Cairo Genizah. Int. J. Comput. Vis. **94**(1), 118–135 (2011)
40. Yin, F., Wang, Q.F., Liu, C.L.: Integrating geometric context for text alignment of handwritten Chinese documents. In: Proceedings of International Conference on Frontiers in Handwriting Recognition, pp. 7–12. IEEE, New York (2010)
41. Zhang, L., Chu, R., Xiang, S., Liao, S., Li, S.: Face detection based on multi-block LBP representation. In: IAPR/IEEE International Conference on Biometrics (2007)
42. Zhang, J., Huang, K., Yu, Y., Tan, T.: Boosted local structured HOG-LBP for object localization. In: Proceedings of CVPR, pp. 1393–1400 (2011)
43. Zhu, B., Nakagawa, M.: Online handwritten Japanese text recognition by improving segmentation quality. In: Proceedings of 11th International Conference on Frontiers in Handwriting Recognition, Montreal, pp. 379–384 (2008)
44. Zimmermann, M., Bunke, H.: Automatic segmentation of the IAM off-line database for handwritten English text. In: Proceedings of ICPR, vol. 4, pp. 35–39 (2002)