



Automated Extraction of Labels from Large-Scale Historical Maps

Inga Schlegel^a (corresponding author)

inga.schlegel@hcu-hamburg.de

^aLab for Geoinformatics and Geovisualization (g2lab), HafenCity University Hamburg, Hamburg, Germany

Abstract. Historical maps are frequently neither readable, searchable nor analyzable by machines due to lacking databases or ancillary information about their content. Identifying and annotating map labels is seen as a first step towards an automated legibility of those. This article investigates a universal and transferable methodology for the work with large-scale historical maps and their comparability to others while reducing manual intervention to a minimum. We present an end-to-end approach which increases the number of true positive identified labels by combining available text detection, recognition, and similarity measuring tools with own enhancements. The comparison of recognized historical with current street names produces a satisfactory accordance which can be used to assign their point-like representatives within a final rough georeferencing. The demonstrated workflow facilitates a spatial orientation within large-scale historical maps by enabling the establishment of relating databases. Assigning the identified labels to the geometries of related map features may contribute to machine-readable and analyzable historical maps.

Keywords: historical maps, text detection, text recognition, text extraction, optical character recognition, levenshtein distance, georeferencing

1 Introduction

Automatically extracting labels from historical maps is not as straightforward as it is the case for current maps (Chiang, 2017; Lin and Chiang, 2018). A frequent lack of in-depth information, which is generally implemented by databases within current maps, impairs a simple search or analysis of places, street or building names, and other local designations within historical maps. As a large part of existing attempts are restricted to e.g. a particular cartographic style and therefore not transferable to others, detecting text in these scanned

and often complex maps is an ongoing challenge (Nazari et al., 2016).

The purpose of this study is to demonstrate a universal solution for an automated detection and recognition of text elements from large-scale ($\geq 1:10,000$, Kohlstock (2004)) historical maps without the need of making major individual adjustments for individual maps. With this goal in mind, we have been able to locate and label geographical features which, in general, are not accessible from historical maps. Besides, a contribution to an approximate georeferencing of historical maps has been made.

We present an automated workflow for detecting and recognizing labels from historical maps and comparing them with current street names. This matching enables a spatial referencing of further streets and places so that an initial spatial orientation within a historical map is possible. The gained information may be useful for subsequent database productions or comparisons between different maps, e.g. from various periods.

This paper is structured as follows. In Sect. 2, an overview of current challenges and related work concerning text extraction from historical maps is presented. Section 3 illustrates details on our used data and methodology before experimental results of individual stages (detection, recognition, and comparison of map labels) of our end-to-end approach are reported in Sect. 4. Finally, Sect. 5 concludes the paper by discussing further potential enhancements and future work.

2 Current challenges and state of research

Compared to current digital maps, a simple scan of a historical map represents no more than an ordinary bitmap image consisting of a number of pixels, each holding a color value. It can be seen as a hybrid of similar color regions, textures, and strokes (Ye and Doermann, 2015). An automated distinction between

text and other map elements such as geometries of buildings or roads is considered as major challenge.

For monochrome historical maps, this differentiation cannot solely be based on color information (Iosifescu et al., 2016). The great stylistic variety among historical maps and their individual typefaces raise a claim for further differentiators when applying automated approaches such as machine learning. Recurring patterns and shapes, which are utilized e.g. in the course of automated face detection or the identification of roads for autonomous driving, can rarely be found within old maps and their labelling (Nazari et al., 2016). Other, technically induced drawbacks turning the described issue into a complex task are a low graphical quality or perspective distortions which are caused by scanning processes, for instance.

The mentioned aspects often cause unsatisfactory results when applying (semi-)automated text detection and recognition to historical maps. Manual post-processing becomes necessary as soon as parts of map labels have not been identified or a context to similar words is missing (Chiang et al., 2020; Chiang and Knoblock, 2014).

Both automated and semi-automated processes aiming at the identification of text in historical maps imply a series of advantages and drawbacks. With semi-automated methods a higher recognition rate of a greater variety of map content can be achieved, whereas, at the same time, laborious manual processing is essential. Hence, only a small quantity of maps can be processed by such time-consuming approaches. Previous research also showed limitations to highly specific map types or typefaces (e.g. straight aligned and horizontal labels or uniform text sizes) do exist (Chiang and Knoblock, 2014). A number of authors have suggested the utilization of a Hough transform to extract text from images or maps but have not considered curved labels (Fletcher and Kasturi, 1988; Velázquez and Levachkine, 2004) or even alphabetic characters (Chen and Wang, 1997). Methods employed by Goto and Aso (1999) and Pouderoux et al. (2007) which identify text in maps based on the geometry of individual connected components do not consider characters of various sizes. Cao and Tan (2002) made use of individual thresholds to detach the black map layer consisting of text and contours as well as of connected components to differentiate between those. Although this is considered a much faster approach compared to a Hough transform, their tailor-made size filters cannot handle overlaps between text and other

map features apart from specific line types (Tombre et al., 2002).

An increasing number of studies are based on the early involvement of a gazetteer or a comparable database available from other sources to match place names with those extracted by small-scale maps (Milleville et al., 2020; Simon et al., 2014; Weinman, 2013). However, this so-called *geoparsing* only works with a comprehensive gazetteer and for place names which do not shift over time. These rarely exist for historical large-scale maps.

To properly address the mentioned issues, Laumer et al. (2020) assigned each pixel either to a map's foreground (resp. labels) or background with the help of convolutional neural networks. Within their approach, labels, or rather clusters built up from interrelated characters, were interpreted, manually matched, and corrected by the combined use of Google's Vision API and a local gazetteer. Machine learning approaches may enable a universal solution to automatically detect and extract text from a variety of maps. Although their application requires a large amount of input training data it offers the advantage to process data without any manual intervention (Chiang et al., 2020). With *Strabo*, Chiang and Knoblock (2014) provide a command line tool for detecting text within maps which is not only based on color differences but also on other characteristics such as the similarity of text sizes or distance measures between individual characters. Its application may be promising when examining monochrome maps.

Until now, machine learning has not been widely used to analyze historical maps. Instead, binarized connected components or other bottom-up approaches have been applied onto maps to detect labels (Weinman et al., 2019). So far lacking in the scientific literature, this paper addresses an appropriate combination of automated text detection and recognition from historical large-scale maps with the aim of extracting machine-readable information.

3 Materials and methods

3.1 Data

For demonstrating our suggested approach with an illustrative example, we chose a large-scale historical map of the city of Hamburg from 1841 (exemplary extract in Fig. 1). Map features such as buildings, built-up areas, roads, railroads, stations, drainage, and docks

are illustrated (Hamburg, Germany 1853). Due to the map's salient color composition and texture the human perception of map objects and their differentiation is facilitated (Schlegel, 2019). The dark labels, primarily designating streets, squares, and water bodies are clearly visible on the bright background but frequently connected to or even overlapping textured objects.

According to general recommendations, a high resolution (≥ 300 dpi) of the scanned input map is ideal so that characters are large enough to be readable by automatic text recognition tools (Milleville et al., 2020). With regard to a reduction of computational cost and time, an appropriate map subset illustrating as many differing map features as possible was chosen for further procedure. The input image, as seen in Fig. 1, was stored in lossless PNG format.



Figure 1: Subset of Hamburg, drawn under the direction of Willm. Lindley, Esqr. C.E. April 1841; engraved by B.R. Davies used as exemplary dataset (Hamburg, Germany, 1853).

3.2 Text detection

With the objectives of

- reducing manual user interaction within the entire workflow and
- increasing the number of true positive labels for a subsequent text recognition

a separation of the map's text from non-text elements was performed using the automatic machine learning approach Strabo¹ (Chiang and Knoblock, 2014; Weinman et al., 2019). Being based on OpenCV's EAST text detector, Strabo is able to detect cartographic labels of different typefaces, sizes, orientations, and curvatures and even those overlapping

¹ Li et al. (2019)

with other map elements (Chiang and Knoblock, 2014; Tombre et al., 2002). Also, blurred, reflective, or partially obscured input images can be processed up to a certain point (Rosebrock, 2018a). The open source tool implements functions of available Python libraries (e.g. *NumPy*, *OpenCV*, *SciPy*, *TensorFlow*, *Matplotlib*) for vector and image processing, statistical computation, machine learning, and visualization. It separates a text layer from the rest of an input image based on differences in color, text size ratios, and appropriate text samples (Chiang and Knoblock, 2014). As an output, Strabo supplies a vector dataset including rectangular bounding boxes each holding an (raster) input image area where text was detected (see upper third of Fig. A1 in Appendix).

3.3 Additional adjustments

As is the case with many applications, Strabo regularly detects only parts of map labels or even omits them entirely. Further manual post-processing is necessary for these results (Chiang et al., 2020). While avoiding an individual editing for each map – whether via pre- or post-processing – we focus on a universal solution to this issue. Regardless of a map's apparent condition, year of creation, style, or color composition a transferability to other similar large-scale maps is desirable.

When working with Strabo we could determine the following points which might have prevented an adequate detection of labels:

- **Specific label orientation** due to the lack of corresponding training data (Chiang, 2019). As suggested by Tesseract's (see also Sect. 3.4) user documentation, we addressed this issue by repeatedly rotating the input image (Tessdoc, 2020). Thus, having five input images in total (rotated through 0°, +45°, +90°, -45°, and -90° resp.), the share of true positives of all existing labels throughout the map, called *recall*, could be increased by about 50% (see (b) in Fig. 3). As can be seen in Tab. 1, this also applies for other maps examined.
- **Overlapping map elements** such as textures, lines, or other labels (see examples in Fig. 2). This is assumed to be a main drawback in the course of text detection (Abdullah et al., 2015; Tofani and Kasturi, 1998). A vast amount of existing algorithms operate on the assumption that black text is in contrast to different-

colored features. However, with a fluent transition between labels and other map elements of the same color their differentiation is scarcely possible within typically black and white historical maps. Due to their occasionally recurring patterns, textures are often mistakenly identified as text by automated detection processes. Tofani and Kasturi (1998), Cao and Tan (2002), Chiang and Knoblock (2014), as well as Nazari et al. (2016) defined different thresholds based on connected components to distinguish between text and other map elements. This laborious task is certainly not adaptable to a large variance of maps.



Figure 2: Overlaps between labels and other map elements are supposed to be a major challenge for automated text detection.

These further drawbacks do not or rarely appear within our presented map but may be a general challenge for text detection:

- **Wide character spacing.** Cartographic labeling principles indicate a smaller spacing between characters compared to words (Chiang and Knoblock, 2014; Yu et al., 2017). According to Strabo's specification, the horizontal space between two characters must be smaller than the largest character so that they are connected to one word (Chiang et al., 2016). This is not the case for e.g. 'Alter Wall' within the upper left part of our map subset illustrated in Fig. 1.
- **Extraordinarily curved labels.** Strabo splits labels deviating substantially from a straight alignment into smaller parts in favor of an enhanced recognition of individual characters (Chiang et al., 2016).
- **Differing text sizes within a label.**
- **Low graphical quality** (Abdullah et al., 2015; Yu et al., 2017; Chiang et al., 2016). Efforts to emphasize and make use of the map's whole RGB color range by linear contrast stretching (normalization) and global histogram equalization made only marginal improvements concerning the overall label detection rate (see (c) in Fig. 3 as well as Tab. 1).



Figure 3: Detected text elements: true positives (blue) and false positives (purple). Strabo was applied to the original image subset (a), the combination of the original and the rotated input image through +45, +90, -45, and -90 degrees (b), and the combination of the original, rotated through +45, +90, -45, and -90 degrees, and enhanced (linear contrast stretching and global histogram equalization) input image (c).

Table 1. Quality of text detection by Strabo revealed by recall, precision, and f-score. The results are derived from the original, original + rotated (through +45°, +90°, -45°, -90°), as well as the original + rotated + enhanced (linear contrast stretching and global histogram equalization) input images.

map (subset)	number of pixels	recall ^a			precision ^b			f-score ^c		
		original map	→ original + rotated map	→ original + rotated + enhanced map	original map	→ original + rotated map	→ original + rotated + enhanced map	original map	→ original + rotated map	→ original + rotated + enhanced map
as shown in Fig. 1	1081 x 881	41%	→ 58%	→ 66%	100%	→ 91%	→ 91%	58%	→ 71%	→ 77%
subset of Fig. 1	468 x 380	34%	→ 56%		92%	→ 86%		50%	→ 68%	
complementary map	1056 x 794	37%	→ 70%		76%	→ 78%		50%	→ 74%	

^a $recall = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$ = percentage of correct detected text elements in respect to the total number of existing text elements

^b $precision = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}$ = percentage of correct detected text elements in respect to the total number of detected elements (Pouderoux et al., 2007)

^c $f - score = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}}$ with 100% indicating perfect recall and precision

The algorithm developed by Chiang and Knoblock (2014) frequently generates multiple bounding boxes for individual labels which rather represent an identical one. Consequently, those bounding boxes belonging to one label overlap each other. Figure 4 illustrates how this spatial relation can be used for merging the affected bounding boxes with the aim to effectively separate off each label from the input image hereafter.

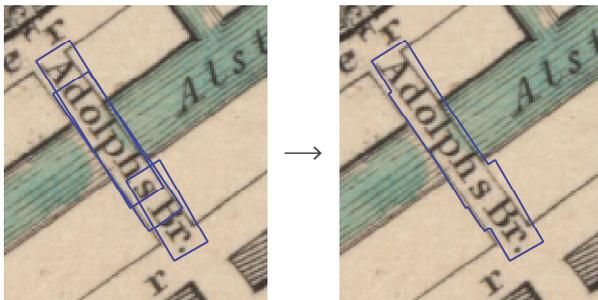


Figure 4: Strabo’s outputted bounding boxes need to be merged per label to effectively separate off them from the map.

In view of the aforementioned causes, overlapping bounding boxes meeting at least one of the following criteria were unified in the order as listed within an iterative procedure:

1. The overlapping area between two bounding boxes is larger than 50% of the smaller bounding box’ area (Fig. 5 (1)).
2. The distance between the centroids of two overlapping bounding boxes is larger than 1.5

times the overall average bounding box height and, at the same time, the difference between their rotation angle is less than 8 degrees (Fig. 5 (2)).

To achieve the desired results, the input data was converted into a local, metric coordinate reference system before calculating each bounding box’ surface area. For criteria (1), the ratio of the overlapping area between two bounding boxes to the area of the smaller one was determined. The two considered polygons were unified into a single one for ratios of at least 50%. Preliminary testing showed that an overlap of 50% or more indicates an incorrect double detection by the algorithm and therefore an identical label. This procedure was iterated until all ratios between two bounding boxes were less than 50%.

Using further Python libraries such as *GeoPandas*, we were able to derive the coordinates of each bounding box’ centroid. NumPy’s *mean()* function helped us to determine the average of the two shortest side lengths over all bounding boxes which was assumed as their initial average height. In combination with their inclination provided by Strabo and normalized to a semicircle covering 0 to 180 degrees, these two variables could be used to find cases exceeding or falling below the thresholds defined from experience for criteria (2). Again, two bounding boxes were unified as long as they fulfilled the mentioned conditions.

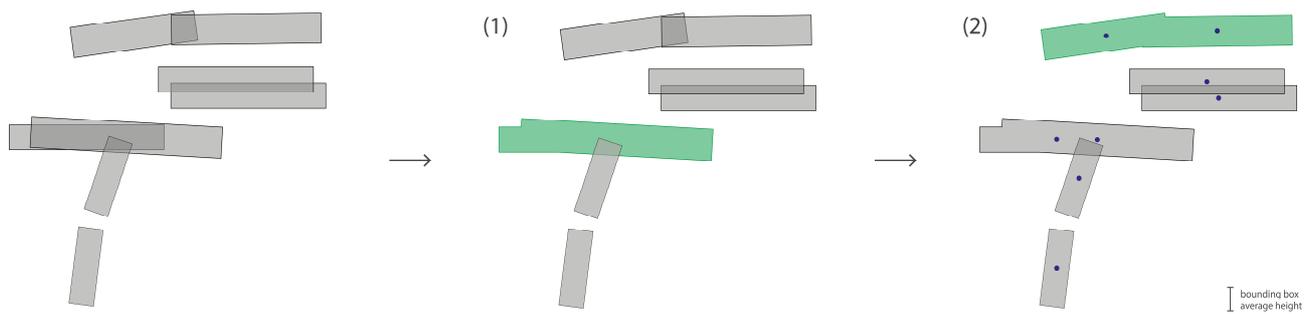


Figure 5: Criteria contributing to a unification of two bounding boxes: overlapping area >50% of the smaller bounding box' area (1) and centroids' distance >1.5 times the overall average bounding box height and rotation angle difference <8° (2).

As a result, each label represented on the input raster map and being localized by Strabo comprised exactly one appropriate bounding box. By applying *ArcPy*, Esri's Python library for spatial data processing, the original input image could therefore be extracted by one of these bounding boxes respectively to generate individual text image areas. Being exported as individual raster files, they were rotated through their averaged rotation angle calculated on the basis of their original bounding boxes. This procedure was implemented to considerably improve the preconditions for the subsequent text recognition (Chiang et al., 2014).

3.4 Text recognition

Available text recognition approaches do rarely achieve satisfactory results regarding raster maps so that additional steps become necessary (Milleville et al., 2020). With the help of a preliminary text detection an early knowledge about the exact location of text can contribute to systematically read content from input images such as historical maps. Combining text detection and recognition in an end-to-end approach not only improves recognition rates but also reduces computing time by focusing on text image areas solely (Ye and Doermann, 2015; Weinman et al., 2019).

To convert the detected text image regions into a machine-readable format, resp. characters and strings, we used the free and open source engine *Tesseract OCR*² (version 4.1.1) which is considered as one of the most accurate tools for optical character recognition (OCR) at present (van Strien, 2020). As all labels within the utilized map subset are in German, this language specification was defined for an improved automatic text recognition by Tesseract. Additionally, each input image should be considered as a single word. The workflow shown in Fig. A2 (see Appendix

starts with an exemplary output from Tesseract for further processing, the string 'Fisch'.

3.5 String similarity

Given a character string for each detected text image area, our aim was to roughly spatially assign them to the input map (Fig. 1). To strengthen the recognition confidence by retaining the one text string turned right way up, Chiang and Knoblock (2013) suggest a juxtaposition of recognized and suspicious characters. However, neither this methodology nor a comparison of similar strings between different maps (such as recommended by Chiang and Knoblock (2014)) considers an appropriate ground truth. In practice, OCR results are rarely precisely identical with a potential ground truth. To attain real names of streets and places, further reference values are necessary. A great variety of existing approaches (e.g. Simon et al. (2014); Weinman et al. (2019)) are based on the comparison with a regional gazetteer. This is, in some cases, available – and therefore efficiently – only for small-scale maps. We take one step further by comparing all recognized strings to an available database³ holding names of current streets and places within the region covered by our map example in Fig. 1, the city center of Hamburg. As certain street names designate e.g. historical events or circumstances and therefore are subject to only minor changes over long periods, this local geodataset could be used as a comparable similarity measure (Hanke, 2014).

To effectively measure the similarity between two strings, namely an OCR output *string_n* indicating a historical street or place on the one hand and a list of current street names (*string_c*) on the other hand, their Levenshtein Distance was defined. We were able to identify street names which are likely to be identical in

² Weil et al. (2020)

³ Freie und Hansestadt Hamburg, Behörde für Wirtschaft, Verkehr und Innovation (2020)

historical and recent maps by applying two different methodologies with the help of the python library *fuzzywuzzy*, which implements the Levenshtein algorithm:

- **ratio** computes the number of character edits (adding, erasing, and replacing) which have to be done to transform $string_h$ to $string_c$ (Yu et al., 2017) and
- **partial ratio** computes the similarity of the shorter substring $string_h$ within parts of the longer $string_c$.

Here, both measures appeared to be of equal value as both individual characters might be recognized incorrectly (\rightarrow ratio) and only parts of strings might be identified (\rightarrow partial ratio).

The output values are defined in percentage ranging from 0 (no similarity) to 100 (identical). Figure A2 (see Appendix) gives several output examples including their percentage value of accordance for the input $string_h$ 'Fisch'. A low score can be an indication of either a poor OCR outcome or a great difference between the historical and current street names $string_h$ and $string_c$. Additional rules being based on various own findings were defined to exclude each $string_h$ from further processing having few (<75%) or multiple identical matching values for $string_c$. On the basis of a defined threshold of 75%, the street names matching those $string_h$ were continued to be used as control points for a subsequent georeferencing.

3.6 Approximate georeferencing

The dataset³ used for allocating current street names helped to perform an initial rough georeferencing of the historical map subset. Since each street within the mentioned geodataset consists of a variable number of linestrings, we defined different rules to find their centroids each representing a street's approximate point-like location: When consisting of only one linestring, the point at half-length was assumed to represent the street's centroid. For those streets comprising two linestrings, the interpolated point at half-length over both lines was specified as the corresponding centroid. For each street being represented by more than two lines, we built the centroid of their common rectangular bounding box. As the bottom section of Fig. A2 (see Appendix) illustrates, these labelled points served as control points for a georeferencing via affine transformation.

4 Experimental results and evaluation

This section points out the results of our methodology as presented in Sect. 3. We primarily conducted tests with the map subset shown in Fig. 1 and complemented other input as necessary.

4.1 Text detection

For the generation of bounding boxes each holding an individual text image area Strabo works best with RGB input images. Own tests confirmed the findings of other authors that there is no difference between lossless PNG and JPEG with smallest possible compression (at least 93% image quality (Mansurov, 2018)) using as an input data format (Milleville et al., 2020; Li et al., 2019). Our results in Tab. 1 reveal that the increase of the label detection rate was not as stark as that of Wilson (2020) when expanding an image's spatial extent.

Various challenges arose when working with Strabo. Due to their frequently similar visual characteristics, the algorithm does not differ between text and similar graphical elements such as textures or edges of map objects, particularly between those being of the same color. Suggested solutions to separate between isochromatic text and lines, such as the inclusion of connected components, may cause negative effects regarding the detection rate (Chiang and Knoblock, 2014).

To facilitate further processes – in particular text recognition and string similarity – the number of detected labels could be increased by own adaptations which were already presented in Sect. 3.3. As shown in Fig. 3 (b), rotating the input image lead to a perceptible increase in the number of correctly found text elements. In reference to a ground truth, the recall could be improved from 41% regarding the original map to 58% after combining it with rotated images through +45, +90, -45, and -90 degrees respectively (Pouderoux et al., 2007). Table 1 shows that examinations with further map subsets revealed an improved recall by up to 50% through this procedure. Initial image enhancements such as linear contrast stretching and global histogram equalization could contribute once more to an improved recall of 66% when regarding Fig. 1. A slight increase of elements falsely detected as text (false positives) and therefore a decrease in the overall precision can be observed in Fig. 3 (c) as well as Tab. 1. As these did not affect the averaged accuracy measure *f-score* to a high degree,

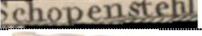
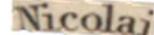
we used the combined input consisting of original, rotated, and enhanced images for further processing. An accurate localization of all text areas is not necessary since the final affine transformation requires only three ground control points.

4.2 Text recognition

Utilizing the derived and unified bounding boxes, the occurrence of text elements within the map could precisely be located. This enabled an improved reading of labels from the input map, the text recognition. As can be seen from Fig. A1 in Appendix, our workflow includes an extraction of all text image areas before

bringing those to a horizontal orientation. Our experiences revealed that Tesseract is incapable of reading text being rotated 10 degrees and more. Recognizing rotated text is an ongoing and still not solved challenge in OCR (Ye and Doermann, 2015; Yu et al., 2017). However, map labels within the bounding boxes might be oriented in two directions. Firstly, right side up in a readable form and secondly, upside down, rotated 180 degrees. The cropped text image areas were consequently rotated through the rotation angle of their associated bounding boxes on the one hand and additional 180 degrees on the other hand.

Table 2. Outputs from Strabo and Tesseract OCR as well as their Levenshtein Distance to current street names³ calculated with the help of the fuzzywuzzy library.

Detected label by Strabo	Rotation angle	Rotated by rotation angle	Recognized string by Tesseract OCR (<i>string_n</i>)	Ground truth string from current street names ³ (<i>string_e</i>)	Average Levenshtein Distance
	179°	no	>Speersort	Speersort	100.0%
	1°	no	Catharineei	Katharinenfleet	33.5%
	179°	no	I Beichei	Siebeneichen	33.5%
	178°	no	chopenstehl	Schopenstehl	98.0 %
	3°	no	Nicola;	Nieland	31.0%
	167°	no	„ame	-	0.0%
	14°	no	ren	Wöhren, Cremon	0.0%
		yes	ud	Hude	33.5%
	54°	no	AN	-	0.0%
		yes	MARKT	Marktweg	33.5%
	55°	no	N	-	0.0%
		yes	Adolphs Br.	Adolphsbrücke	84.0%
	88°	no	-	-	-
		yes	klopfen markt	Hopfenmarkt	87.0%

An appropriate input data pool for an optical character recognition by Tesseract OCR was hereby created. As the map's original lossless PNG format performed poor for text recognition, all files were transferred in TIFF and RGB color mode. Further testing with grayscale

and binary input images did not show any improvement.

Regarding Tesseract's output (examples shown in Tab. 2), a reasonable number of text strings could be

recognized distributed over the entire map. Only minor deviations from a manually prepared ground truth could be identified for horizontal labels. Similar results were also given for further tested input maps. Although Tesseract generally assumes a clean, plain input image and its model is trained on specific typefaces, interfering artifacts such as parts of lines, textures, and other map elements did not considerably deteriorate the outcomes (Rosebrock, 2018b).

4.3 Matching to current data

Several concurring names could be identified between historical and current streets and places. After applying fuzzywuzzy's (partial) ratio the previously derived centroids (see Sect. 3.3) of Tesseract's output on the one hand and the local geodataset³ including current street names on the other hand could be matched in a satisfactory manner for our map example (Fig. 1). As seen in Tab. 2, the average Levenshtein Distance of matching strings such as *Adolphsbrücke*, *Hopfenmarkt*, *Schopenstehl*, or *Speersort* exceeded our defined threshold of 75%. We could continue to use those labels having high matching rates and a good distribution over the raster map. In combination with their centroids they served as reference points for a subsequent allocation of all remaining streets as well as for an initial rough georeferencing of the historical map. By assigning street labels to specific locations within the map, the meaning and context (semantics) of those could be specified (see Fig. 6).



Figure 6: Current names of streets and places spatially assigned to the georeferenced historical map.

5 Conclusions and outlook

This study can be understood as a proof of concept for an automated end-to-end workflow to extract labels from large-scale historical maps. Our findings that

detection and recognition rates are generally low (<80% and <60% on average respectively) are broadly consistent with Weinman et al. (2019) and point out necessary improvements for machine learning approaches (Ye and Doermann, 2015). By combining tools addressing text detection, recognition, and string similarity with further adjustments we were able to not only increase the overall recognition rate but also to provide a base for useful ancillary information such as the names of streets and places. This may be considered a promising aspect of searchable and analyzable historical maps. Furthermore, a georeferencing, which is frequently lacking for historical maps, could roughly be made. For best results, those labels having highest similarity rates and an appropriate scattering over the map should be considered as reference points. A great benefit may be a resulting facilitated comparison between different maps such as between historical and current ones.

We demonstrated the possibility of transferring the suggested approach to a variety of maps due to omitting individual adjustments. Nevertheless, disturbing factors such as interfering artifacts from building corners, textures, or map grids may occur and can therefore still be challenging for different maps. Further testing with additional maps might be helpful to specify and minimize the sources of disturbance more precisely.

To improve the overall accuracy of the presented approach, we suggest connecting identified single words to complete map labels. This may be achieved by looking closely at the adjacency and similarity of rotation angles of detected text image areas. Also, map labels covering multiple lines should be considered. The certainty of true positives may therefore be increased for all substeps within our comprehensive approach.

Future research might continue to use our results to label further map features and to assign those to their related geometries. The identification of geometries such as from streets, buildings, or waterbodies may be facilitated by a preceding elimination of all detected labels within a map. Segmenting and classifying map objects based on their different properties could support the establishment of ancillary, informative databases and therefore enable the analyzability of historical maps. With this kind of feature matching, not only further map objects might be identified but also a more intuitive comparison between historical and current maps would become possible.

6 Data and software availability

All research data and applications produced and applied within this publication can be found at <https://doi.org/10.5281/zenodo.4721174> (Schlegel, 2021). The repository is structured following Sect. 3 of this paper.

The results were generated using QGIS Desktop 3.16.0 (approximate georeferencing, Sect. 3.6), the command prompt in Windows 10 OS (Tesseract OCR, Sect. 3.4), the Linux (Ubuntu 18.04) command line via Windows-Subsystem for Linux (Strabo, Sect. 3.2), as well as several Jupyter Notebooks (additional adjustments, Sect. 3.3 and string similarity, Sect. 3.5) written in Python. These scripts are available under the GNU GPLv3 license.

The workflow underlying this paper was partially reproduced by an independent reviewer during the AGILE reproducibility review and a reproducibility report was published at <https://doi.org/10.17605/osf.io/avn9r>.

References

- Abdullah, A., Abo Alshamat, S. S., Bakhshwain, A. and Aslam, A.: Automatic text removal and replacement in scanned meteorological thematic maps, *JCS.*, 11, 772–783, <https://doi.org/10.3844/jcssp.2015.772.783>, 2015.
- Cao, R. and Tan, C. L.: Text/Graphics Separation in Maps, in: *Graphics Recognition, Algorithms and Applications*, Proceedings of the 4th International Workshop on Graphics Recognition, Kingston, Canada, 7-8 September 2001, 167–177, https://doi.org/10.1007/3-540-45868-9_14, 2002.
- Chen, L.-H. and Wang, J.-Y.: A system for extracting and recognizing numeral strings on maps, in: *Proceedings of the Fourth International Conference on Document Analysis and Recognition*, Ulm, Germany, 18-20 August 1997, 337–341, <https://doi.org/10.1109/ICDAR.1997.619867>, 1997.
- Chiang, Y.-Y.: Unlocking textual content from historical maps – Potentials and applications, trends, and outlooks, in: *Recent Trends in Image Processing and Pattern Recognition*, Proceedings of the International Conference on Recent Trends in Image Processing and Pattern Recognition, Bidar, India, 16-17 December 2016, 111–124, https://doi.org/10.1007/978-981-10-4859-3_11, 2017.
- Chiang, Y.-Y. (University of Southern California, Los Angeles, CA, USA): Personal communication, 2019.
- Chiang, Y.-Y. and Knoblock, C.: Strabo: A Complete System for Label Recognition in Maps, 26th International Cartographic Conference, Dresden, Germany, 25–30 August 2013, 2013.
- Chiang, Y.-Y. and Knoblock, C. A.: Recognizing text in raster maps, *GeoInformatica.*, 19, 1–27, <https://doi.org/10.1007/s10707-014-0203-9>, 2014.
- Chiang, Y.-Y., Moghaddam, S., Gupta, S., Fernandes, R. and Knoblock, C. A.: From map images to geographic names, in: *SIGSPATIAL '14*, Proceedings of the 22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, Dallas, TX, USA, 4-7 November 2014, 581–584, <https://doi.org/10.1145/2666310.2666374>, 2014.
- Chiang, Y.-Y., Leyk, S., Honarvar Nazari, N., Moghaddam, S. and Tan, T. X.: Assessing the impact of graphical quality on automatic text recognition in digital maps, *Computers and Geosciences.*, 93, 21–35, <https://doi.org/10.1016/j.cageo.2016.04.013>, 2016.
- Chiang, Y.-Y., Duan, W., Leyk, S., Uhl, J. H. and Knoblock, C. A. (Eds.): *Using Historical Maps in Scientific Studies. Applications, Challenges, and Best Practices*, Springer, Cham, Switzerland, <https://doi.org/10.1007/978-3-319-66908-3>, 2020.
- Fletcher, L. A. and Kasturi, R. A.: Robust Algorithm for Text String Separation from Mixed Text/Graphics Images, *TPAMI.*, 10, 910–918, <https://doi.org/10.1109/34.9112>, 1988.
- Freie und Hansestadt Hamburg, Behörde für Wirtschaft, Verkehr und Innovation: *Straßen- und Wegenetz Hamburg* (HH-SIB), <https://suche.transparenz.hamburg.de/dataset/strassen-und-wegenetz-hamburg-hh-sib16?forceWeb=true>, 2020.
- Goto, H. and Aso, H.: Extracting curved text lines using local linearity of the text line, *IJDAR.*, 2, 111–119, <https://doi.org/10.1007/s100320050041>, 1999.
- Hamburg, Germany, 1853 (Raster Image), Harvard Map Collection, Harvard College Library, <https://maps.princeton.edu/catalog/harvard-g6299-h3-1853-15>.
- Hanke, C.: *Hamburgs Straßennahmen erzählen Geschichte*, 5th ed., Medien-Verlag Schubert, Hamburg, Germany, 2014.

- Iosifescu, I., Tsorlini, A. and Hurni, L.: Towards a comprehensive methodology for automatic vectorization of raster historical maps, *e-Perimetron.*, 11(2), 57–76, 2016.
- Kohlstock, P.: *Kartographie*, 1st ed., Ferdinand Schöningh, Paderborn, 2004.
- Laumer, D., Gümgümcü, H., Heitzler, M. and Hurni, L.: A Semi-automatic Label Digitization Workflow for the Siegfried Map, in: *Automatic Vectorisation of Historical Maps*, Proceedings of the International Workshop on Automatic Vectorisation of Historical Maps, Budapest, Hungary, 13 March 2020, 55–62, <https://doi.org/10.21862/avhm2020.07>, 2020.
- Li, Z., Chiang, Y.-Y., Banisetti, S. and Kejriwal, L.: *strabo-text-recognition-deep-learning* (Version 0.67), GitHub repository, <https://github.com/spatial-computing/strabo-text-recognition-deep-learning>, 2019.
- Lin, H. and Chiang, Y.-Y.: SRC: Automatic Extraction of Phrase-Level Map Labels from Historical Maps, *SIGSPATIAL Special.*, 9(3), 14–15, <https://doi.org/10.1145/3178392.3178400>, 2018.
- Mansurov, N.: JPEG Compression Levels in Photoshop and Lightroom: <https://photographylife.com/jpeg-compression-levels-in-photoshop-and-lightroom>, last access: 12 April 2021, 2018.
- Milleville, K., Verstockt, S. and van de Weghe, N.: Improving Toponym Recognition Accuracy of Historical Topographic Maps, in: *Automatic Vectorisation of Historical Maps*, Proceedings of the International Workshop on Automatic Vectorisation of Historical Maps, Budapest, Hungary, 13 March 2020, 63–72, <https://doi.org/10.21862/avhm2020.01>, 2020.
- Nazari, H. N., Tan, T. and Chiang, Y.-Y.: Integrating text recognition for overlapping text detection in maps, *Document Recognition and Retrieval.*, XXIII, 1–8, <https://doi.org/10.2352/ISSN.2470-1173.2016.17.DRR-061>, 2016.
- Pouderoux, J., Gonzato, J., Pereira, A. and Guitton, P.: Toponym Recognition in Scanned Color Topographic Maps, in: *Proceedings of the Ninth International Conference on Document Analysis and Recognition*, Curitiba, Brazil, 23–26 September 2007, 531–535, <https://doi.org/10.1109/ICDAR.2007.4378766>, 2007.
- Rosebrock, A.: OpenCV Text Detection (EAST text detector): <https://www.pyimagesearch.com/2018/08/20/opencv-text-detection-east-text-detector>, last access: 27 October 2020, 2018a.
- Rosebrock, A.: OpenCV OCR and text recognition with Tesseract: <https://www.pyimagesearch.com/2018/09/17/opencv-ocr-and-text-recognition-with-tesseract>, last access: 13 November 2020, 2018b.
- Schlegel, I.: Empirical Study for a Deployment of a Methodology for Improving the Comparability Between Historical and Current Maps, *KN – Journal of Cartography and Geographic Information.*, 69(2), 121–130, <https://doi.org/10.1007/s42489-019-00016-0>, 2019.
- Schlegel, I.: Label Extraction: v1.1.0 (Version v1.1.0), Zenodo, <https://doi.org/10.5281/zenodo.4721174>, 2021.
- Simon, R., Pilgerstorfer, P., Isaksen, L. and Barker, E.: Towards semi-automatic annotation of toponyms on old maps, *e-Perimetron.*, 9(3), 105–112, 2014.
- Tessdoc: Tesseract documentation. Improving the quality of the output: <https://tesseract-ocr.github.io/tessdoc/ImproveQuality> last access: 3 November 2020.
- Tofani, P. and Kasturi, R. (1998): Segmentation of text from color map images, in: *Proceedings of the Fourteenth International Conference on Pattern Recognition*, Brisbane, Australia, 16–20 August 1998, 945–947, <https://doi.org/10.1109/ICPR.1998.711391>, 1998.
- Tombre, K., Tabbone, S., Pélissier, L., Lamiroy, B. and Dosch, P.: Text/Graphics Separation Revisited, in: *Document Analysis Systems V.*, Proceedings of the 5th International Workshop on Document Analysis Systems, Princeton, NJ, USA, 19–21 August 2002, 200–211, https://doi.org/10.1007/3-540-45869-7_24, 2002.
- van Strien, D.: Living with Machines OCR hack: <http://livingwithmachines.ac.uk/living-with-machines-ocr-hack>, last access: 25 August 2020.
- Velázquez, A. and Levachkine, S.: Text/Graphics Separation and Recognition in Raster-Scanned Color Cartographic Maps, in: *Graphics Recognition, Recent Advances and Perspectives*, Proceedings of the 5th International Workshop on Graphics Recognition, Barcelona, Spain, 30–31 July 2003, 63–74, https://doi.org/10.1007/978-3-540-25977-0_6, 2004.
- Weil, S. et al.: Tesseract Open Source OCR Engine (main repository) (Version 4.1.1), GitHub repository, <https://github.com/tesseract-ocr/tesseract>, 2020.

Weinman, J. (2013): Toponym recognition in historical maps by gazetteer alignment, in: Proceedings of the 12th International Conference on Document Analysis and Recognition, Washington, DC, USA, 25–28 August 2013, 1044–1048, <https://doi.org/10.1109/ICDAR.2013.209>, 2013.

Weinman, J., Chen, Z., Gafford, B., Gifford, N., Lamsal, A. and Niehus-Staab, L.: Deep neural networks for text detection and recognition in historical maps, in: Proceedings of the 15th International Conference on Document Analysis and Recognition, Sydney, Australia, 20–25 September 2019, 902–909, <https://doi.org/10.1109/ICDAR.2019.00149>, 2019.

Wilson, D.: Finding words in maps: <http://livingwithmachines.ac.uk/finding-words-in-maps>, last access: 25 August 2020.

Ye, Q. and Doermann, D.: Text Detection and Recognition in Imagery: A Survey, TPAMI., 37, 1480–1500, <https://doi.org/10.1109/TPAMI.2014.2366765>, 2015.

Yu, R., Luo, Z. and Chiang, Y.-Y. (2016): Recognizing text in historical maps using maps from multiple time periods, in: Proceedings of the 23rd International Conference on Pattern Recognition, Cancun, Mexico, 4–8 December 2016, 3993–3998, <https://doi.org/10.1109/ICPR.2016.7900258>, 2017.

Appendix

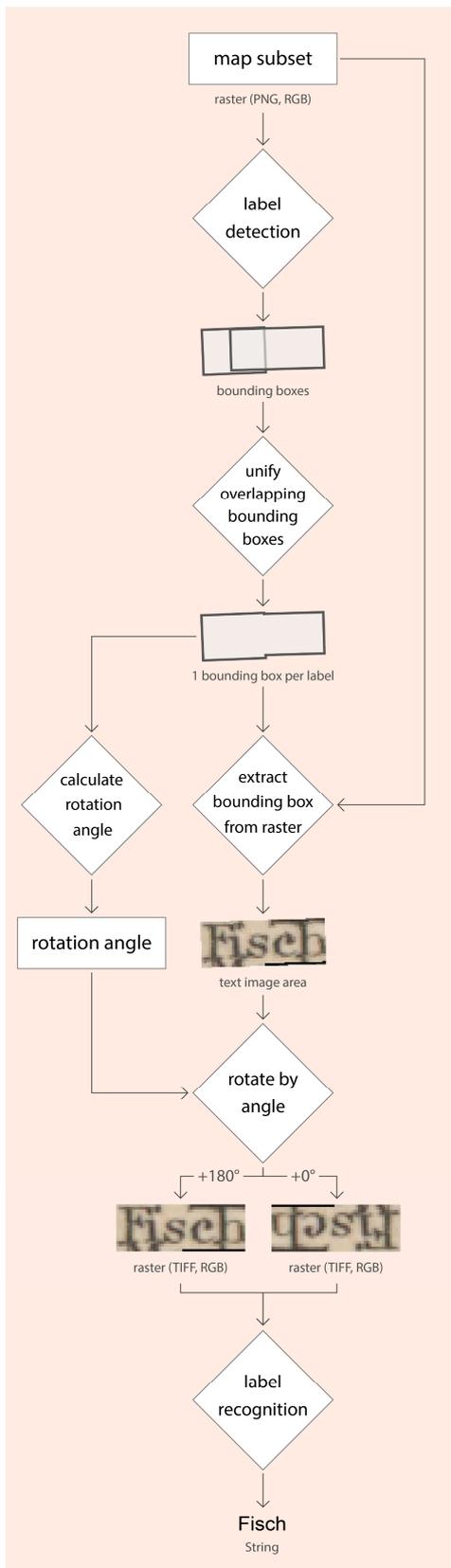


Figure A1: Workflow from label detection to recognition for a map subset including interposed further adjustments.

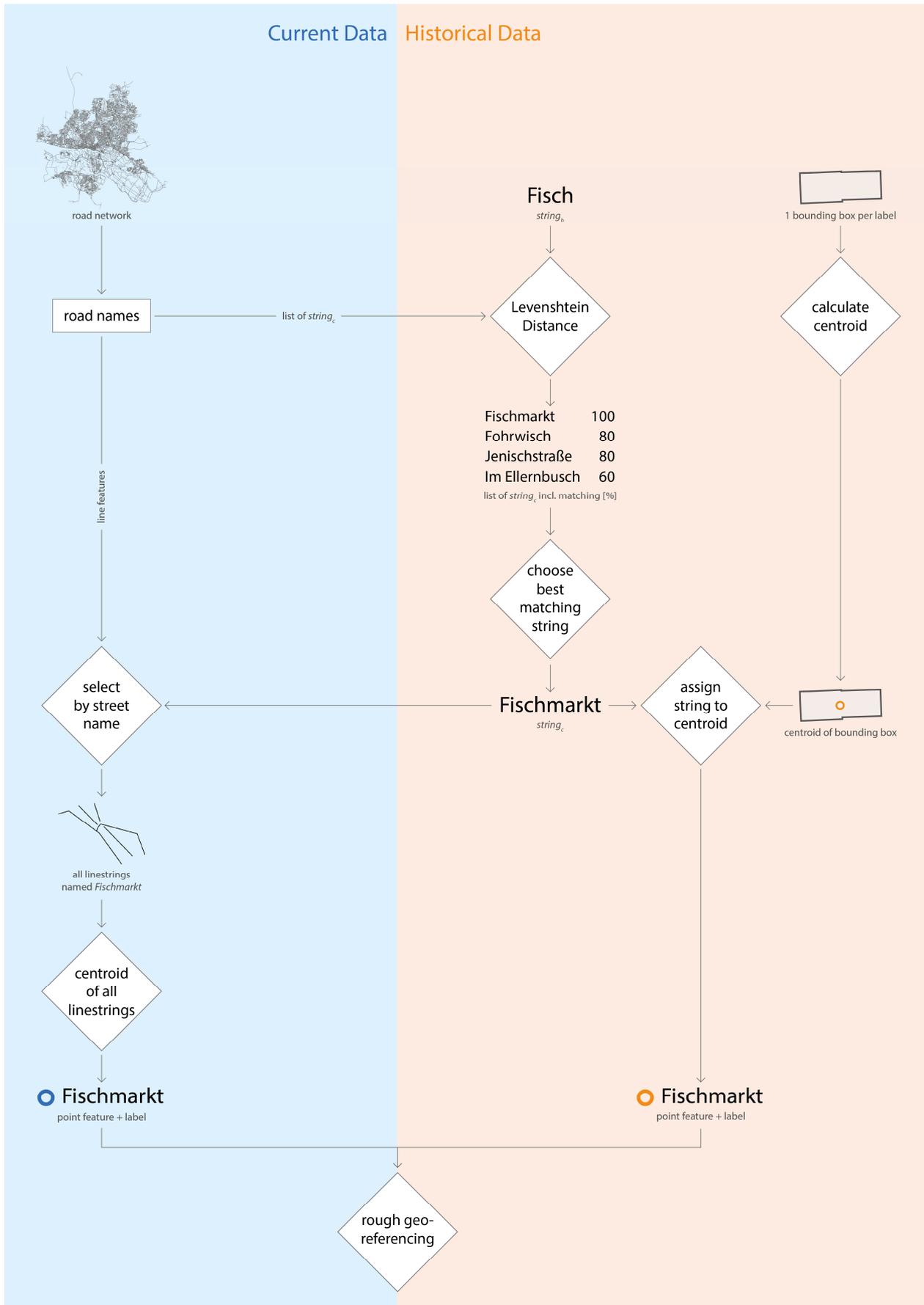


Figure A2: Workflow for matching historical to similar current street names with the aim to perform a rough georeferencing.