Julius-Maximilians-Universität Würzburg
Institut für Informatik
Lehrstuhl für Informatik I
Effiziente Algorithmen und wissensbasierte Systeme

Master Thesis

Algorithmic Analysis of Historical Maps

Benedikt Budig

submitted on September 3, 2014

supervisors:
Prof. Dr. Alexander Wolff
M. Sc. Thomas van Dijk

Zusammenfassung

Historische Landkarten stellen eine Informationsquelle von zunehmender Bedeutung für Forscher verschiedenster wissenschaftlicher Disziplinen dar. Mit der fortschreitenden Digitalisierung in Bibliotheken und Archiven stehen diese Landkarten für einen größeren Kreis von Wissenschaftlern zur Verfügung. Der Digitalisierungsprozess und die Analyse der Inhalte alter Landkarten ist allerdings eine komplexe und langwierige Arbeit. Metadaten, die die Karte und die enthaltenen Informationen beschreiben, müssen extrahiert werden, was derzeit weitestgehend von Hand von Experten erledigt wird.

Im ersten Teil der vorliegenden Arbeit schlagen wir ein neuartiges, ganzheitliches System vor, das eine halbautomatische Informationsgewinnung aus historischen Landkarten erlaubt und auf Benutzerinteraktionen reagiert. Wir skizzieren dieses System, indem wir in einem interdisziplinären Ansatz Methoden aus der Mustererkennung, Linguistik, Kartographie, algorithmischen Geometrie und Graphentheorie verbinden. Dazu regen wir eine Sammlung algorithmischer Werkzeuge an, die die essentiellen Probleme der Informationsgewinnung aus historischen Landkarten lösen und zu einer Verarbeitungspipeline verbunden werden können. Ziel der einzelnen Module des Systems ist eine (weitgehend) automatische Erkennung und Zuordnung von Ortsnamen sowie die Analyse in den Karten enthaltener Geländetopographie.

Im zweiten Teil der Arbeit gehen wir in Form einer Fallstudie auf eines dieser Module im Detail ein: wir entwickeln eine algorithmische Methode, um die Zugehörigkeit von Ortsmarkierungen und ihren Beschriftungen auf historischen Landkarten zu ermitteln. Diese mühsame Aufgabe ist auch für Menschen nicht trivial, liefert aber wertvolle Metadaten und spielt bei der anschließenden Georeferenzierung eine Schlüsselrolle. Wir modellieren das Problem mittels kombinatorischer Optimierung und zeigen eine effiziente Lösungsmethode. Auf zwei historischen Landkarten mit insgesamt mehr als 4000 Ortsmarkierungen und Beschriftungen, die zu Testzwecken manuell extrahiert wurden, ordnet der Algorithmus 99% der Elemente richtig zu. Zusätzlich führt unser Algorithmus eine Sensitivitätsanalyse auf der ermittelten Lösung durch und kann damit ein Maß für das Vertrauen in jede einzelne Zuordnung berechnen. Dies dient als Grundlage für einen interaktiven Dialog mit dem Benutzer, in dem die Resultate durch gezielte Benutzerinteraktionen weiter verbessert werden können. Außerdem wird ein Konzept zum Umgang mit fehlerbehaftetem Input vorgestellt.

Mit der Umsetzung des gesamten vorgeschlagenen Systems wird es möglich sein, den Zeitaufwand drastisch zu reduzieren, der zur Erhebung geographischer Informationen auf historischen Landkarten notwendig ist. Gegenwärtig benötigen erfahrende Konservatoren 15 bis 30 Stunden, um die Orte zu georeferenzieren, die in einer durchschnittlichen Karte mit 1000 bis 4000 Ortsmarkierungen enthalten sind. Unser System setzt sich zum Ziel, das selbe Ergebnis innerhalb einer Stunde zu erreichen (möglicherweise mit der Hilfe von Laien anstatt Experten).

Schlagworte: Historische Landkarten, Algorithmische Werkzeuge, Georeferenzierung, GIS, Digitalisierung, Informationsgewinnung, Metadatenerhebung

Contents

Introduction 4

Part I Proposal of a System for the Analysis of Historical Maps

1 Problem Description 7

2 Related Work 9
2.1 Systems for Metadata Extraction and Management . . . 9
2.2 Research on Related Subproblems . . . 10

3 Methodology and Outline 12
3.1 Segmentation and Image Based Analysis . . . 13
3.2 Matching Markers and Labels . . . 16
3.3 Handwritten Character Recognition . . . 18
3.4 Georeferencing Places . . . 19

4 Outlook 21

Part II Case Study: An Algorithm for Matching Place Markers and Labels

5 Matching Markers and Labels 23
5.1 Optimization Problem . . . 23
5.2 Polynomial-Time Algorithm . . . 24

6 Experiments 26
6.1 Balanced Case . . . 26
6.2 Imbalanced Case . . . 27
6.3 Parameter Choice . . . 29

7 Sensitivity Analysis and User Interaction 31
7.1 Sensitivity-based Classification . . . 31
7.2 Interactive Postprocessing . . . 32
7.3 Qualitative Discussion of Classified Matches . . . 36

8 Matching Markers with Sets of Labels 38
8.1 Optimization Problem . . . 39
8.2 ILP and Proof of NP-Hardness . . . 39
8.3 Polynomial-Time Algorithm for a Restricted Problem . . . 40

Conclusion and Future Work 42

Appendix 43

Bibliography 45

Introduction

Historical maps are highly valuable documents for scholars, as they serve as a rich source of information and contain distinctive geographic features that are rarely preserved in other types of sources. The present trend of digitizing documents, for instance by the Google Books project[1], also increases the availability of historical maps. As a result, such maps are used more frequently as resources for historical and geographic research. Figure 1 shows a section of a historical map containing a variety of topographic information.

Figure 1: Section of a topographic map from 1787, showing the area around Bad Königshofen in the north of Bavaria.

Currently, digitization and content analysis of ancient maps is a very time-consuming process. After the elaborate task of scanning the document using specialized hardware, the contents of the map need to be analyzed in order to create metadata and to make the map more easily retrievable. Sometimes, towns contained in the map are georeferenced during this process, which means that they will be matched with the corresponding town on a modern map (and its geographic position). This step is particularly useful, as it allows interested scholars to search for maps containing certain towns or depicting a specific geographic region. In subsequent steps, researchers and curators might process the map further, for example by creating thematic indices of the contained places, analyzing land usage and forest cover or the course of rivers. There is an interdisciplinary interest in such information, as it can be used for research in the fields of history, geography, geodesy and economy, amongst others. As an example, Schuppert and Dix (2009) reconstructed historic cultural landscapes in several regions of Germany using historical maps, which they had to georeference and evaluate manually.

In Part I of this thesis, we sketch a novel system that holistically extracts relevant information from historical maps in a (semi-)automated fashion. Following an interdisciplinary approach, our proposed system allows to detect and georeference places in historical maps with considerably less user effort required. We also include physical geographic features, which so far have mostly been analyzed manually in independent studies, if at all. Presently, metadata is mainly created by expert map conservators. In contrast, our

[1] `http://books.google.com/`

approach aims at simplifying and automating tasks such that even laypersons with basic geographic knowledge of the map area can achieve valuable results.

In Part II, we present a case study in which we discuss one of the modules proposed in this system in detail: we introduce an algorithmically-assisted method for determining the proper correspondence between place markers and their labels in historical maps. This time-consuming step in the digitization process of historical maps is non-trivial even for humans, but provides valuable metadata and is useful when subsequently georeferencing the map. In order to speed up the manual process, we model the problem in terms of combinatorial optimization, show how to solve that problem efficiently, and how user-interaction can be used to improve the quality of results.

In the course of the project, we leveraged local expertise at the Julius Maximilian University Würzburg. Dr. Hans-Günter Schmidt is Head of Department of Manuscripts and Early Prints at the University Library in Würzburg. The *Franconica* collection[2], containing more than 800 historical maps conserved by his department, was our primary source of maps for this project.[3] Dr. Schmidt was also significantly involved in the development of a distributed digitization workflow system that is now in use at his department; see Schöneberg et al. (2013) and Höhn et al. (2013). The system we propose could be integrated into this workflow at a later time.

[2] `http://franconica.uni-wuerzburg.de/`

[3] All maps shown in this document are part of the *Franconica* collection.

Part I

Proposal of a System for the Analysis of Historical Maps

1 Problem Description

The research problem[1] that is addressed in this thesis is the development of methods to (semi-)automatically extract relevant information from historical maps. This includes physical geographic features like the course of rivers as well as thematic features like a georeferenced index of contained towns. At present, such information is primarily extracted manually, at best assisted by tools such as those described in Chapter 2. This is still a considerably time-consuming task: an expert map conservator needs 15 to 30 hours in order to georeference a historical map containing 3000 to 4000 places. A fully automated system that extracts all relevant information from the maps seems hard to achieve due to the diverse nature of historical maps. Instead, we propose advanced methods to process such documents in a way that requires significantly less user effort. We therefore need to identify steps in the digitization process that can be approached algorithmically.

There is little literature available on algorithmic methods for information retrieval in historical maps, and it is generally restricted to a limited set of features and specific types of maps. In particular, there exists no system that combines methods from different disciplines in order to holistically approach the problem. Due to the difficulty of the problem, it is crucial to establish a dialogue between the system and the user in order to guide him or her and to be able to respond to feedback immediately. Our goal is to create such a system; however, a complete development of all modules required would be beyond the scope of this thesis. Instead, we give an outline identifying important subproblems and offer possible approaches to solve them, see Chapter 3.

In the following paragraphs, we give a short overview on challenging problems in the metadata extraction process that need to be solved algorithmically. Taking the bitmap images of the scanned maps as a starting point, the first step in the workflow is an image-based analysis. This step requires a segmentation method that reliably separates text, place markers and physical geographic features contained in the map. However, these features are hard to recognize and distinguish. This is because of their close arrangement in the map (which, for instance, may lead to arbitrarily rotated text labels) and because text and pictograms are hand-drawn and thus often differ considerably. In addition, due to the wide variety of visual styles in historical maps, elements tend to be different for every map. See Figure 2 for an example of the placement of elements in historical maps and Section 3.1 for our proposed approaches to the problem of segmentation.

Once the segmentation is completed, it is important to correctly match place markers to their corresponding labels. Due to the dense placement of markers and labels, in some cases it is not even apparent to the human reader how to match them. For example, geographic circumstances may lead to labels being placed at a significant distance from their corresponding place markers in the map. However, the correct assignment of markers and labels is necessary for subsequent steps. The topic is dealt with in Section 3.2 and a detailed description of our algorithmic solution to the problem is given in Part II.

To complete georeferencing, the label corresponding to each place marker needs to be analyzed by a system for optical character recognition (OCR) that supports handwritten

[1] Part I of this thesis was used in a research proposal by the author and describes the outline of a system that will be created in a subsequent research project.

Figure 2: Dense placement of elements in historical maps (created in 1600 and 1746).

characters. This allows us to identify places in the historical map with modern places of the same name. The largely varying handwritings as well as the age and condition of the maps make this problem especially complex: it is often the case that labels in historical maps are completely illegible to regular OCR systems. In addition, modern place names (or at least their spellings) often differ considerably from the place names found in historical maps. It will be necessary to augment an existing OCR system for our purpose; see Section 3.3.

We suggest to make our system available to the end user by implementing it as an extension to an existing digitization workflow system. Specifically, we propose to integrate the system into the workflow software by Schöneberg et al. (2013), which is used at the University Library in Würzburg.

2 Related Work

The digitization and software assisted analysis of historical maps is of increasing interest to libraries and collections of historical documents all over the world. For this reason, several systems providing features that simplify this complex process have been developed. We give a critical review of relevant map digitization systems and discuss their advantages and shortcomings, see Section 2.1. In addition, there is research on related topics such as the segmentation of bitmap images of (historical) maps and postprocessing of georeferenced maps, see Section 2.2.

2.1 Systems for Metadata Extraction and Management

Note that not all systems presented in this section have the same scope and address the same problems: some of them can be regarded as academically motivated proof of concepts, while others are commercial software products designed for every day usage by librarians. Most of these systems provide convenient graphical interfaces, but still rely heavily on users to manually georeference or annotate the input maps.

Reference and Annotation Tool (RAT) Schöneberg et al. (2013) introduced a digitization workflow system that was developed for the specific needs of the digitization center of the University Library in Würzburg. It has a modular structure and tries to ensure digitization quality using automated image processing and error detection modules. In addition, modules for data mining and metadata augmentation allow users to extract and save semantic and structural metadata of historical documents.

The *Reference and Annotation Tool (RAT)* by Höhn et al. (2013) is a module for this workflow system that has been designed to extract information about places in historical maps during the digitization process. For the detection of place markers, template matching using normalized cross-correlation is applied to the bitmap image of the map. This process relies on user provided templates for all different types of place markers used in the map. In a second step, the system supports the user in georeferencing detected place markers by suggesting names of modern places in geographic proximity. Possible candidates are identified by calculating a projective transformation between points on the historical map and their coordinates on a modern map, based on previously georeferenced points. A search within a specified radius around these coordinates returns a list of candidate places, from which the user must then pick the correct match.

However, this list of place names is unsatisfactory, as it often contains a confusingly large number of candidates for each place. According to the Department of Manuscripts and Early Prints at the University Library in Würzburg, it still takes the 15 to 30 hours of experts' time to process a large map using RAT. We want to address this problem by taking the actual labels for each place marker into account, which will narrow the list of possible place names significantly. Although the scope of the system by Höhn et al. (2013) is similar to parts of the system we propose, it still relies heavily on users supervising the process and manually performing tasks. Our proposed system, on the other hand, works

on a higher level of automation. It aims at identifying obvious place names directly and providing better assistance for users in difficult cases.

Georeferencer A system specially focused on georeferencing places in historical maps is *Georeferencer* by Fleet et al. (2012). This application is designed as an online platform and leverages the possibilities of collaboration and crowdsourcing. It is currently deployed by several large institutions holding collections, including the *British Library* (London) and the *Nationaal Archief* (The Hague).

The georeferences are achieved by collecting control points on both the historical map and the modern map. These points are specified manually by the users. To simplify this task, corresponding sections of historical and modern maps are displayed side by side. Once there are sufficiently many control points available, historical maps can also be presented as an overlay on modern maps or satellite images. Subsequently, georeferenced maps can be analyzed in terms of geometric accuracy by external applications like *MapAnalyst*, described in the next section.

Although *Georeferencer* works well for pure georeferencing of historical maps, it lacks support of handling the actual contents of the processed maps. There is no support for detecting or adding meta data and no extraction of map features. Instead of relying on a crowd of volunteers who manually georeference points, our approach aims at automatically identifying and thus georeferencing places contained in the map. In addition, this gives us deeper insight into the actual contents of the analyzed historical map.

YUMA The *YUMA Map Annotation Tool* by Simon et al. (2011) has its primary focus on the semantic annotation of historical maps. Annotations can be added by a team of collaborating scholars as well as by crowdsourcing. The system is available online to its users and allows them to add and modify annotations, which are displayed directly in the historical map. From these annotations, structured semantic metadata is created by linking to possibly relevant web resources, visualized by tag clouds. Search queries to an index of all annotations can be used to find maps with specified content. However, there is no automation to retrieve content from maps or to assist users.

2.2 Research on Related Subproblems

Apart from software systems providing general solutions for metadata creation and management, there is also work related to subproblems within or subsequent to possible processing pipelines. For the postprocessing of georeferenced maps, Jenny and Hurni (2011) introduced a tool which is able to analyze the geometric and geodetic accuracy of historical maps and visualizes identified distortions. Even if the projection of the historical map is unknown (which is common), their system is able to find the best fitting projection out of a set of likely map projections. While there is no automated processing of the maps' contents, the output can be used to assess the accuracy of the historical maps themselves and the quality of the automated georeferencing process.

Some research has also gone into image segmentation related to historical maps. Höhn (2013) introduced a method to detect arbitrarily rotated labels in historical maps; Mello et al. (2012) dealt with the similar topic of identifying text in historical maps and floor plans. However, the detection of text labels (and other map elements as well) can in general not be considered a solved problem. Many algorithms for extracting semantic information from bitmap images have precision and recall that is far from perfect. This is to be expected, since these problem are truly difficult for computers. To the curators

of historical map collections, however, the correctness and completeness of metadata is of paramount importance.

There is little research available on algorithmic information retrieval from historical maps. Fully automatic approaches exist, but only for restricted inputs, that is, developed specifically to digitize a particular corpus. For example, Leyk et al. (2006) describe a method to find forest cover in a specific set of 19$^{\text{th}}$ century topographic maps. Arteaga (2013) extracts building footprints from a set of historic maps from the New York Public Library (NYPL), particularly, georectified scans of insurance atlases published in the 19$^{\text{th}}$ and early 20$^{\text{th}}$ centuries. The effectiveness of these approaches is in part due to the homogeneity of these relatively recent maps. The *Franconica* collection contains a great amount of much older maps; the tests in Part II of this thesis are performed on maps from the 16$^{\text{th}}$ and early-18$^{\text{th}}$ century.

3 Methodology and Outline

In this chapter, we give an outline of our proposed system and discuss some preliminary work. Proceeding from an overview of possible data sources and the system structure, we present novel approaches for each task in the processing pipeline.

We first show how to approach segmentation and feature extraction of a bitmap of a historical map; see Section 3.1. Then we discuss the problem of matching place markers with place labels (see Section 3.2); an algorithmic solution for this problem will be presented in Part II. Subsequently, we examine the application of handwritten character recognition systems to retrieve texts in place labels; see Section 3.3. Finally, we discuss how places contained in the map can be georeferenced using information extracted in the previous steps; see Section 3.4.

Data Sources The primary input to our system will be high-resolution bitmap images of historical maps, for example obtained by scanning. In cooperation with the University Library at Julius Maximilian University Würzburg, we have access to almost 100 high-quality scans of historical maps created between the 16^{th} and the 19^{th} century. These maps are focused on the area around Würzburg within a radius of approximately 100 km and are part of the *Franconica* collection[1]. The collection features a total number of more than 800 historical maps.

In addition to this data source, we consider maps from several other collection. Particularly useful for accessing historical maps is *Old Maps Online*[2], which features a georeferenced index containing more than 120 000 maps from 20 university libraries and archives worldwide. Besides the map image as the main input, our system makes use of modern map data both for georeferencing and obtaining a list of modern places in the target area. To cope with historical spelling differences in place names, we will use dictionaries of historical place name variants, for instance the geographic norm data provided by the German National Library[3].

System Structure The proposed system is modular, which means that each step in the processing pipeline has well-defined input and output and can therefore be used and evaluated independently from other modules. This allows us to have the case study on the matching module in Part II without having to implement the remaining modules of the pipeline. Important outputs of our system include georeferencing of the historical map, an index of contained places as well as the position and size of physical geographic features like woodlands and rivers. In all steps, user feedback will be interactively taken into account. See Figure 3 for an overview over the proposed modules and the system structure.

[1] `http://franconica.uni-wuerzburg.de/`
[2] `http://www.oldmapsonline.org/`
[3] `http://www.dnb.de/EN/Standardisierung/GND/gnd_node.html`

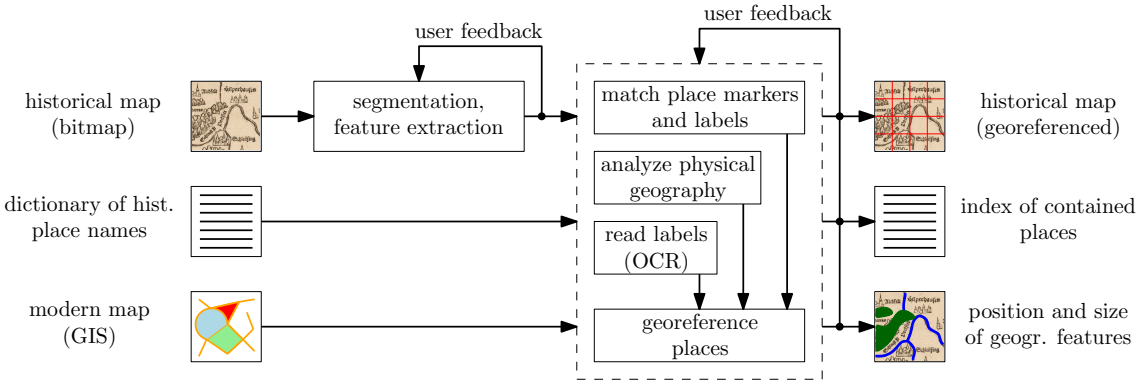

Figure 3: Overview of proposed modules and possible system structure.

3.1 Segmentation and Image Based Analysis

Segmentation is the problem of recognizing and locating features that are contained in the map. It is the first step in our processing pipeline and fundamental to all following steps. Presently, we focus on three different types of features: place markers, place labels and physical geographic features. The properties of these features will be explained (together with our proposed retrieval methods) in the following paragraphs. Based on approaches from the area of computer vision, our solutions take advantage of the special characteristic of historical maps.

Place Markers

With the term *place markers*, we refer to those elements in a map that indicate the position of a place. In many historical maps, marked places are mostly settlements, varying from small hamlets to large towns. Some maps use a variety of pictograms to mark places, indicating the type of settlement. Other maps uniformly use small circles; see Figure 4 for examples. In addition to the diversity of different types of place markers, even markers of the same kind vary to some degree, as the maps are hand-drawn. Furthermore, legends specifying place markers used in a map cannot be expected to be exhaustive or present at all. This renders the detection of place markers in historical maps considerably challenging.

Höhn et al. (2013) propose a method to detect place markers in historical maps using template matching. Their method is based on searching the map for pictograms similar to a set of manually provided place marker templates using normalized cross-correlation. This approach worked well in some of the tests that the authors conducted. However, it is not able to cope with scaled or rotated pictograms and relies heavily on manually provided templates for each type of place marker and manually tuned threshold values.

Figure 4: Examples of various types of place markers used in different historical maps.

For non-technical users, it is challenging to adjust these parameters correctly and to supply an appropriate set of templates such that usable results can be achieved. To tackle this problem, we propose a dialogue-style approach that interactively collects required information from the user. The goal of our protocol is to achieve better results both by automation and by collaborating with the user in a clearly defined way that leverages his or her experience. The user will initially only be asked to select a single place marker on the map, which will be used for a first template matching run (using a default threshold). Resulting matches will be displayed immediately, for instance as an overlay on the map. In subsequent steps, the user can add an additional, previously unmatched place marker by selecting it in the map. Furthermore, he or she may select and remove place markers (false positives). Depending on these user actions, our proposed system adds new (positive) templates, takes negative templates into account, or tunes matching thresholds automatically. This process is iterated until a satisfactory result is obtained. To make efficient use of the user's time, the system directs him or her to uncertain matches by shading or highlighting areas of the map. As a result, only limited parts of the historical map have to be evaluated manually. The uncertainty of each match can be assessed via the measure of similarity used in the template matching.

In addition, we propose to review and evaluate measures of similarity other than the normalized cross-correlation. Our preliminary tests indicate that a method using the normalized correlation coefficient is superior to the approach by Höhn et al. (2013) for certain marker types. With an in-depth evaluation of different measures on different historical maps, we want to get insight into what approaches work best and how to choose an adequate approach for a given set of markers.

Due to the diversity of place markers within a historical map and even more across different maps, a fully automated approach that does not rely on human post-processing does not yet seem feasible. However, as a pre-processing step to the protocol explained above, some place markers in the map could be detected automatically. A possible method to recognize them would be to find elements of reasonable size in the map and cluster them according to their similarity. Together with their frequency of appearance, we could determine the likelihood that the items of each cluster depict place markers in the map.

Place Labels

Generally, place markers in historical maps are labeled with the name of the marked place. This means that most markers have a lettering called *place label* next to them. These labels consist of handwritten characters, sometimes with a line break due to insufficient free space in the map. They differ in size and can be arbitrarily oriented to fit into the respective area of the map, as Figure 5 shows. In some cases, labels are split by other features in the map (e.g. rivers) or overlap with place markers.

Figure 5: Place labels in different historical maps. Note that some labels are rotated, while others are split or consist of several words.

There is some related work dealing with the detection of text labels in maps. Mello et al. (2012) introduced an algorithm that allows the segmentation of historical maps and floor plans by removing non-textual elements. It makes use of edge detection and connected component analysis. Another approach, which is specifically focused on text labels in historical maps, was developed by Höhn (2013). Using connected component analysis, he recognizes text components in a scale- and rotation-independent way. In an unpublished project report[4] we introduced two preliminary approaches for text detection on different types of historical maps. They work with connected components on sparsely labeled maps and combine this with pattern matching (of a small set of characters) on densely labeled maps. While first results look promising, there are still improvements necessary to obtain the quality needed for our application.

Physical Geographic Features

In addition to the geographic locations of settlements, many historical maps contain information about the physical geography in the mapped area. We refer to such map elements as *physical geographic features*. Out of 16 historical maps from the *Franconica* collection, all maps contain rivers and water courses as well as woodlands; all but one map also include mountain areas. The most important physical geographic features are rivers, whose courses are depicted by thick strokes. In contrast, the extent of forest and especially mountain areas is often only hinted at by clusters of hand-drawn pictograms. Examples of these features are shown in Figure 6. River courses can also be relevant to the automatic recognition and georeferencing of the represented landscape: in historical maps, the location of settlements is often represented best in relation to rivers. The one-dimensional subspace of a river is much less distorted then the surrounding two-dimensional map.

The diverse representation of physical geographic features in historical maps demands individual approaches for each feature. For the detection of rivers, both connected component analysis and edge detection mechanisms seem appropriate. Recognizing woodlands is more challenging, as they are often depicted as a dense accumulation of (potentially overlapping) small pictograms. To tackle this problem, we propose an approach that analyzes the ratio of ink and blank paper within a sufficiently large window sliding over the map. This is based on the observation that, due to the dense arrangement of pictograms, forest areas appear darker than open land on the maps. For the recognition of mountain areas, this method could be used as well, provided that mountain pictograms are densely clustered. While this is true for the representation of mountain ranges in many historical maps, there also exist free standing hills, displayed by a single pictogram. These can be recognized using the approach introduced for the detection of place markers in Section 3.1.

[4]Budig, B., Chlechowitz, M., Kauer, J., Löffler, A., and Wisheckel, F. (2014). Abschlussbericht zum Projekt Texterkennung auf historischen Landkarten. Universität Würzburg.

Figure 6: Physical geographic features in different maps, depicting forests, rivers and hills.

An interesting approach for the detection of forests in historical maps has been introduced by Leyk et al. (2006). It focuses on a specific set of topographic maps created in the late 19^{th}. A combination of character recognition, line detection, statistical classification and structural analysis obtains very good results on this specific collection of maps.

A special case of physical geographic features are lakes, which appear less often in historical maps of the *Franconica* collection. This might be due to the relatively small scale used by many of them, which renders most lakes too small to be included. For maps with larger scales that do include lakes, Shaw and Bajcsy (2011) introduced an algorithm specifically designed for the segmentation of lakes from historical maps. However, it relies on templates indicating an approximate shape for each lake that is to be found.

3.2 Matching Markers and Labels

Once place markers, place labels and physical geographic features have been identified, we want to determine the relation between these elements in the map. For our algorithmic approach on georeferencing places contained in historical maps, it is crucial to identify which of the place labels belongs to which of the place markers. We call this task the *matching* of markers and labels.

Although it might appear trivial at first glance, this is actually a hard problem: due to the dense placement of features, it is not always apparent which label belongs to which marker. Often, mostly for reasons of space, labels are split into multiple parts, wrapped around other map features or placed on the opposite side of rivers. See Figure 2 and Figure 7 for examples of dense and occasionally distant placements of labels in a historical map. In this section, we introduce important properties of markers and labels, point out a concept on how to automatically match them (which will be discussed in detail in Part II), and describe means of estimating the quality of the obtained results.

Properties of Markers and Labels

For the task of matching place markers with place labels, we deal with each map element in terms of its bounding rectangle. Most place markers, even if they consist of large pictograms, do specify the position of the depicted settlement as a point in space, which can be important for the quality of subsequent georeferencing. However, in terms of labeling, not the point location but the extent of the entire pictogram is decisive. In other words, labels in historical maps generally refer to place marker pictograms and not to the actual point locations of the indicated places. Therefore, the bounding box of each pictogram is suitable for our matching process. We represent the place labels by their bounding boxes as well.

An immediate observation when dealing with historical maps (and also most modern maps) is the fact that labels are generally positioned *near* their corresponding markers. Based on this, we assume that the Euclidean distance (in image space) between place markers and labels can be used as a measure of how likely they belong together.

Modeling using Minimum Cost Matchings

To automatically determine a corresponding place label for each place marker, we propose an approach utilizing *matchings*, which is a well-known concept from the field of graph theory. Given a weighted graph, a matching is a set of edges without common vertices and its cost is defined as the sum of the weights of its edges. The basic idea of our method described in Part II is to approach the problem of assigning place markers to

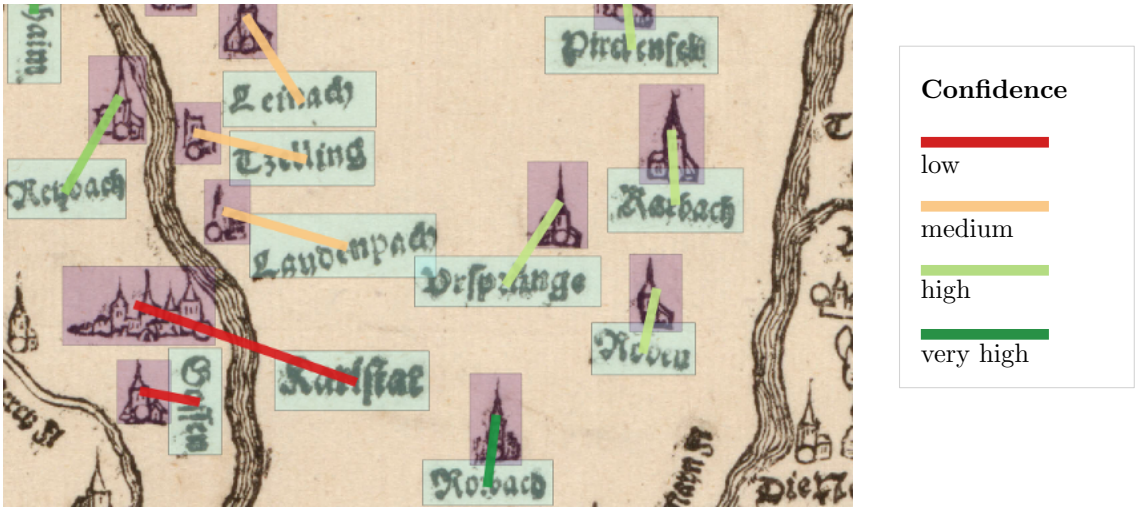

Figure 7: Sensitivity analysis of a matching (as it could be presented to the user). Colors refer to the system's confidence in each assignment, increasing from red to green.

labels by finding a matching of minimum cost. We therefore assign the weights of the edges corresponding to the distance between the map elements. The result is a one-to-one assignment of place markers and labels such that the sum of distances between all matched labels and markers is minimized.

Sensitivity Analysis for Matchings

The large variety of visual styles occurring in our target maps makes it necessary to ask the user for feedback in difficult situations. In order to avoid that users have to inspect the whole map, which often contains thousands of place markers and labels, it helps to point out critical situations. The system's confidence in the resulting matching could be displayed in a user-friendly way, for example using color-coded line segments visually connecting place markers with their matched labels. This allows the user to quickly recognize matches our system has low confidence in.

In this context, the concept of *sensitivity analysis* is relevant, which addresses the question of how sensitively a system reacts to (possibly very small) changes of its inputs. This concept is used in several fields, including Bayesian networks and scheduling. Sensitivity analysis is relevant to us because our inputs are not certain: we cannot expect that the segmentation procedure from Section 3.1 is going to yield a complete set of precise bounding boxes for each label and marker in the map. Instead, some elements might have not been detected in their full extent or even not at all, for example due to the nature of hand-drawn pictograms or the state of preservation of the map. The proposed sensitivity analysis helps to cope with such uncertainties.

Liu and Shell (2011) introduced an extension of the Hungarian method due to Kuhn (1955) to analyze the sensitivity of minimum-cost matchings: For every edge in a matching, the algorithm calculates the interval of possible deviation of its weight such that the given assignment will not to change. Originally developed for multi-robot task assignment, this method can be adapted to our purposes. It allows us to assess how sensitive (to each edge weight) the matching obtained by our method is. The edges with weights relatively near the interval boundaries must be considered less certain and should be displayed to the user for manual inspection. Figure 7 shows the results of a sensitivity analysis we have calculated using this approach on an actual matching.

3.3 Handwritten Character Recognition

Optical character recognition (OCR) is both an important theoretical problem in the field of computer vision and has a significant practical impact. However, it remains a very challenging problem, especially when trying to recognize handwriting. In the context of our project, we will use OCR to recognize text in place labels featured in historical maps.

Difficulty of the Problem

The task of recognizing characters or identifying words in historical maps is very complex. This is due to the variety of handwritings, limited space resulting in a dense placement of characters as well as the age and condition of the maps. Additionally, non-textual map elements as rivers or hills might intersect with the labels. For examples of the place label "Würzburg" in three different historical maps, see Figure 8.

Phonetic Algorithms and Dictionaries

Considering the complexity of handwriting recognition in historical maps, we cannot assume that the text of every label will be recognized correctly. However, it is possible with many OCR systems to provide a dictionary of likely words, which assists the system in recognizing the correct strings. In the context of place labels in historical maps, such a dictionary could contain the names of all places in the covered area. To acquire an index of modern place names in given areas, we can query modern geographic information systems. However, potential spelling difference between historical and modern place names prevent the immediate use of such indices. To overcome this problem, we can match the given index with another dictionary containing historical spelling variants, for example the geographic norm data of the German National Library[5].

In addition, we propose to use phonetic algorithms that identify words by their pronunciation rather than by their spelling. For German words, the *Kölner Phonetik* due to Postel (1969) is an example for such an algorithm. To be useful for our purpose, a phonetic algorithm needs to be included into the OCR system itself. While this approach seems to generalize well, it lacks support for synonyms and place names that have completely changed. For instance, the city of Würzburg is on some old maps referred to by the Greek-Latin name of "Herbipolis" or "Herbapolis".

An integrated approach, first trying to find a place name in the dictionary and then using a phonetic algorithm to identify words with similar pronunciation, seems most promising for our problem. The *tesseract* OCR system[6] could serve as a strong technical basis for the proposed extensions. It is available as open source software and continuously developed by the Google Books project. Identifying historical place names with modern equivalents is not only helpful for the OCR process, but also very important for the task of georeferencing, as we will see in the next section.

[5] `http://www.dnb.de/EN/Standardisierung/GND/gnd_node.html`
[6] `https://code.google.com/p/tesseract-ocr/`

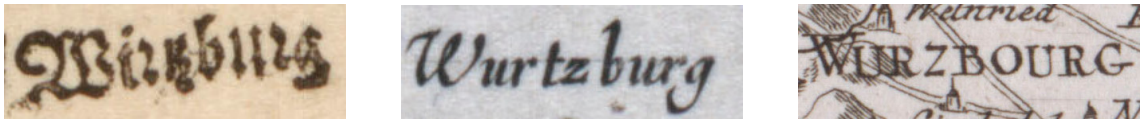

Figure 8: Labels for the city of Würzburg in three historical maps (created in 1533, 1626 and 1787); note the differences in style and spelling.

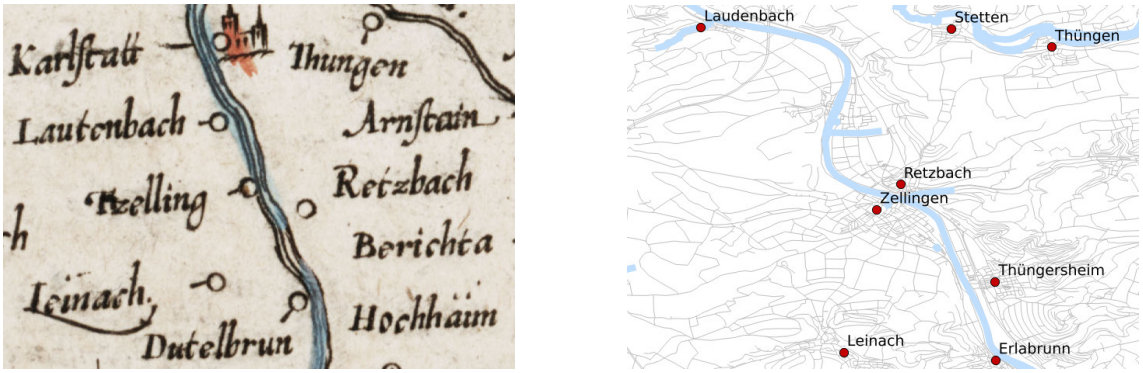

Figure 9: Villages along the river Main, shown in a historical and a modern map[7]. Leinach, Retzbach, Zellingen, Laudenbach and Thüngen can be found on both maps.

3.4 Georeferencing Places

Accurately georeferencing places in the given historical map is one of the key features of the proposed system. For this task, information collected from all system components introduced in this chapter will be combined in order to obtain precise georeferencing results. In a holistic approach, we consider place labels, georeferenced surroundings as well as physical geographic features.

The obvious (human) approach to determine the identity of a place marked in a map is to find and read the corresponding label. With a combination of our segmentation, matching and OCR modules, we can use the same approach: first, place markers and labels will be detected and subsequently matched. Then, the OCR component recognizes the place names in the labels and finds the corresponding modern place name, if necessary. Once a place in the historical map can be identified as a modern place of the same name, georeferencing of that place is already completed, assuming the geo-coordinates of the modern place are known. The geographic norm data of the German National Library introduced above already contains such coordinates for every entry. In addition, modern geographic information systems can be queried to obtain geo-coordinates of a place with given name.

However, as pointed out in the previous sections, we cannot assume that each module always returns perfect results. For instance, place labels might be unreadable to the OCR system (c.f. Figure 8) or not recognized correctly by the segmentation module. Still, we might be able to georeference the place correctly using additional information collected while processing the map. Assuming that its place marker has been detected correctly, the identity of a place could be concluded from its already georeferenced surroundings. Using a modern GIS, we can determine possible candidates within a certain query region. In the historical map in Figure 9, the label "Tzelling" might be too smudged for the OCR system to return good results. Now, assume that Leinach, Lautenbach and Retzbach, which have clearer labels, have been georeferenced correctly. Considering their positions in relation to the position of the uncertain place identifies the latter clearly as Zellingen (using the modern map data shown on the right).

In less certain cases, we may need to ask for user feedback. However, we are still able to provide likely suggestions for the place name and point the user to the specific area of the map. If the label corresponding to the unidentified place has been detected and matched correctly, it is also possible to show the cropped label text to users and have them decide

[7]Modern map data obtained from the Open Street Map project, `http://www.openstreetmap.org/`

whether the text actually reads the name of the most likely candidate place. For this task, expert users are not necessarily required; instead, volunteers or crowdsourcing could be engaged. Note that the New York Public Library has a very successful, comparable crowdsourcing project[8], in which volunteers assess the quality of automatic detection of building footprints in historical city maps.

An alternative technique can be used to conclude the identity of places that are positioned along a river or coast line. In most historical maps, even in those that are heavily geographically distorted, the order of settlements along rivers is correct. Provided place markers and rivers have been detected correctly, we can leverage this fact to identify uncertain places by taking the previous and following places along the river into account.

[8]NYPL Labs Building Inspector, `http://buildinginspector.nypl.org/`

4 Outlook

In addition to the concrete approaches introduced in the previous chapter, we also present further-reaching questions for which we do not have specific ideas yet. The following problems could be addressed as future work or in addition to the aforementioned problems in the course of the development of our proposed system.

The production of maps has always been expensive and time-consuming. This is particularly true for historical maps, as they were handcrafted, for example using copperplate engravings. Engraved copperplates were too valuable to be simply discarded after their first use; instead, some copperplates were used in a variety of maps and constantly extended over several decades. It is therefore of interest to recognize how historical maps are related and whether they share a common source (like the same copperplate). This could be determined by automatically comparing pictograms and the location of common places in various maps.

In this context, it would also be desirable to transfer knowledge from an already analyzed map in order to analyze a similar one. This involves the matching of markers and labels (if the drawing of two maps is related) as well as orthographic characteristics (if two maps were created in the same period). If the style of handwriting is similar over several maps, it could also be feasible to train the OCR system to that specific style during the analysis of the first map. To further improve the results obtained by OCR, we also consider creating custom modifications for Gothic handwriting.

Last but not least, we want to assess which tasks in our proposed pipeline are suitable for crowdsourcing. Creating an attractive smartphone application, we might be able to encourage geographically or historically interested volunteers to casually support the University Library's digitization efforts. The New York Public Library successfully uses a similar concept, as described above.

Part II

Case Study: An Algorithm for Matching Place Markers and Labels

5 Matching Markers and Labels

In the case study[1] presented in this part, we concern ourselves with one specific step in the metadata extraction process: the matching of place labels to place markers. This is in fact a non-trivial problem, even for humans. (See Figure 10 for a tricky situation that we will later discuss in more detail.) We focus on a specific subtask of the digitization process in order to focus on a manageable problem. Our solution solves the matching problem discussed in Section 3.2 and can be implemented as a module in the proposed metadata extraction system. In the next chapter, we will present experiments and evaluate the performance of our approach.

First, we (re-)introduce the map elements we want to work with and give preliminary definitions holding for the remaining part of this thesis:

Definition 1 (Place Marker)**.** *A* place marker *(short:* marker*) is a map element—typically a pictograph—indicating the geographic position of a point of interest. In our model of the problem, a place marker is represented by an axis-aligned bounding rectangle.*

Definition 2 (Label)**.** *A* label *is a piece of text in the map that labels a certain place marker. In our model of the problem, a label is represented by an axis-aligned bounding rectangle.*

Let P denote the set of place markers contained in a historical map and L the set of contained labels, all represented by their axis-aligned bounding box.

5.1 Optimization Problem

Recall that our goal is to identify the correct correspondence between labels and place markers. We assume that this matching has the following two properties: every $p \in P$ is assigned to at most one $l \in L$, and every $l \in L$ is assigned to at most one $p \in P$. Note that with this formulation, we do not demand a *perfect matching*, i.e. a one-to-one assignment of place markers and labels. This is necessary because many historical maps contain a small number of unlabeled markers and stray labels, for example due to the conservation state (faded areas, cracks, etc.) and errors during production.

We have observed in Section 3.2 that labels are generally positioned *near* the place markers they belong to. This is also the basic assumption of our matching model. For

[1] Parts of this case study will be published in a research paper containing joint work with Thomas van Dijk and Alexander Wolff.

Figure 10: A difficult case: without geographic or historical context, it is hard to tell which of these three assignments is correct.

a place marker $p \in P$ and a label $l \in L$, we define the distance $d(p, l)$ as the Euclidean distance between the closest pair of points in p and l. Since p and l are rectangles, this distance can be easily determined. In addition, we assume that labels are never located more than a certain distance r from the marker they are labeling. This parameter has to be chosen somewhat carefully, because an insufficiently large value of r might disallow the correct assignment (see Section 6.3 for a discussion). Semi-formally, our goal is to find a matching M of elements in P and elements in L, such that:

$$\text{No match } (p, l) \in M \text{ has distance } d(p, l) > r. \tag{5.1}$$

$$\text{The sum over } d(p, l) \text{ for all } p, l \in M \text{ is small.} \tag{5.2}$$

$$\text{The size of the matching } M \text{ is large.} \tag{5.3}$$

Finding such a matching is indeed not trivial; a greedy approach that iteratively takes the match with lowest distance does not perform well (see Chapter 6). Instead, we formulate an optimization problem with the following objective function:

$$f_{\text{obj}}(M) = \sum_{(p,l)\in M} r - d(p, l) \tag{5.4}$$

We want to maximize f_{obj} under the constraint that M is a matching. We call this the Assign Labels problem.

Note that constraint (5.1) will always hold in an optimal solution, as adding any pair (p, l) with $d(p, l) > r$ to M decreases the objective value. The parameter r thus has another useful interpretation: it limits the "marginal cost" of adding an additional match (p, l) to M, that is, how much the sum of distances in M is allowed to rise in order to increase the cardinality of the matching by one: if adding (p, l) to M leads to a decrease in $f_{\text{obj}}(M \setminus (p, l))$ by more than r (e.g. due to reassigning elements in order to observe the matching constraints), this pair will not be added.

5.2 Polynomial-Time Algorithm

We solve the Assign Labels problem using a flow-based approach. Let $G = (V, E)$ be a directed acyclic graph with a set of vertices V and a set of edges E. We identify vertices $v_p \in V$ and $v_l \in V$ with place marker p and label l, respectively. G is composed of four layers: the first layer contains the source vertex s, the second layer contains vertices v_p for all $p \in P$, the third layer contains vertices v_l for all $l \in L$ and the fourth layer contains the sink vertex t. Every vertex from each layer is connected to every vertex of the following layer by an edge. Figure 11 gives an overview of the layout of G.

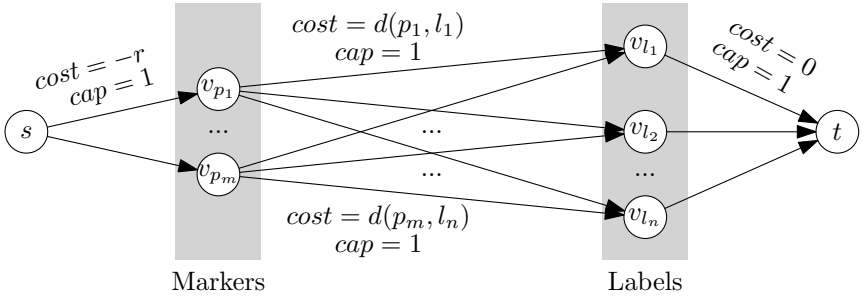

Figure 11: Layout of flow network G used to solve the Assign Labels problem.

We want to apply the concept of minimum cost flows. Thus, we need to translate the maximization into a minimization problem and define edge weights and capacities accordingly. Let edges sv_p connecting the source vertex s to vertices in the second layer have $cost(sv_p) = -r$. For edges v_pv_l connecting vertices in second and third layer, we assign $cost(v_pv_l) = d(p, l)$. The remaining edges v_lt have $cost(v_lt) = 0$. All edges e in E have $cap(e) = 1$.

We can easily derive matchings from flows in this network: place marker p and label l are matched if and only if the flow value of edge v_pv_l is greater than 0. For our flow network G, Lemma 26.9 in Cormen et al. (2009) holds: if M is a matching as defined above, then there is an integer-valued flow f in G with flow value $|f| = |M|$. Conversely, if f is an integer-valued flow in G, then there is a matching M between markers and labels with cardinality $|M| = |f|$. The proof given by Cormen et al. (2009) can directly be applied here, as G is a *corresponding flow network* and the paths induced by edges in M are edge-disjoint in G. Moreover, all capacities in G are integer, so their Integrality Theorem (Theorem 26.10) holds. A maximum matching can thus be found by computing a maximum flow in G; a flow in G correctly models a matching as described above.

The edge costs of an s-t path in G correspond to the terms in the sum of f_{obj}. We can solve the resulting minimum cost flow instance for a given flow value d using any among a number of known polynomial-time algorithms, for example the push-relabel method by Goldberg (1992). Finding a flow of minimum cost over all admissible flows in G gives an optimal solution for the model introduced above. Since all capacities in the flow network are integer, we can also efficiently use linear programming to find an optimal solution (see Schrijver (2003)).

6 Experiments

We have implemented the algorithm described in the previous section using linear programming. The experiments presented in this section have been run on a laptop PC with an Intel® Core™ i5-3427U CPU at 1.80 GHz and 8 GB of main memory. We have used CPLEX v12.5.1 for solving linear programs. To obtain test data, we have manually extracted all place markers and labels contained in two historical maps from the *Franconica* collection, the *Franckenland* map[1] created in 1533 and the *Circulus Franconicus* map[2] from 1706. These maps are displayed in their full extent in the appendix of this document. With both maps, we have assigned place markers and labels by hand and use this as a ground truth for testing. Unless otherwise noted, we have used a fixed value of $r = 150\,\text{px}$ for our experiments on both maps. We discuss this value in Section 6.3; for now, see Figure 16 for an indication of scale.

6.1 Balanced Case

First, we have run experiments with our algorithm on *balanced* input data. We mean by this that the ground truth data is a one-to-one assignment, which admits a perfect matching. This is not the case in all historical maps, even if the input perfectly models the contents of the map: we had to filter a small amount of unlabeled markers and stray labels out of both maps to obtain this property.

The input data based on the *Franckenland* map thus contains 517 place markers and labels. Taking 0.9 seconds of runtime, our algorithm assigns 515 of them correctly and makes 2 incorrect assignments (Exp. F1). The *Circulus Franconicus* map contains a considerably higher number of place markers and labels. In our test run, 1636 out of 1644 markers were assigned correctly, with the remaining 8 markers matched to incorrect labels (Exp. C1). The required runtime was 2.1 seconds; see Table 1 for statistics.

[1] Sebastian von Rotenhan. *Das FranckenLandt = Chorographi Franciae Orien[talis]*, 1533.

[2] Frederik De Wit. *Circulus Franconicus: in quo sunt episcopatus Wurtzburg, Bamberg et Aichstet, Status Equitum Teutonicor(um), Ducatus Coburgensis, Marchionatus Cullembach et Onspach, Comitatus Henneberg, Wertheim, Holach, Reinec, Papenheim, Erpach, Schwartzenberg, et Castel, Baronatus Sensheim et Territorium Norinbergense*, 1706.

	Exp. F1	Exp. F2	Exp. C1	Exp. C2
number of place markers	517	539	1644	1663
number of labels	517	524	1644	1669
correct matches	515	503	1636	1626
incorrect matches	2	14	8	20
error ratio	0.4%	3.5%	0.5%	1.3%
runtime	0.9 s	1.0 s	2.1 s	2.2 s
greedy error ratio	17.8%	17.8%	5.4%	5.9%

Table 1: Statistics of our experimental results.

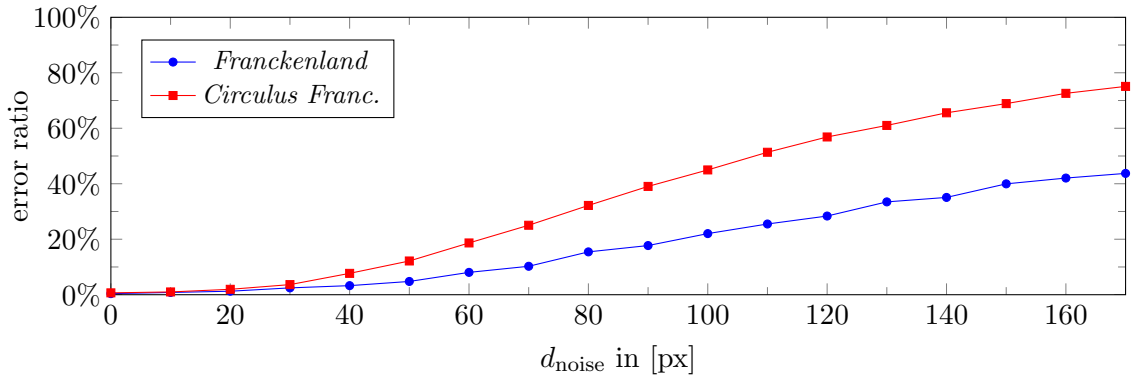

Figure 12: Impact of d_{noise} on the error ratio of our algorithm. Note that for values of $d_{\text{noise}} \leq 30\,px$, the error rate is contained at a low level.

In both experiments, the error ratio is below 1%, which we consider a good result given the dense and sometimes inconsistent placement of elements in the maps. For comparison, we have also implemented a greedy approach, which iteratively adds an assignment with smallest distance to the matching, until all remaining potential assignments exceed cost r. The greedy algorithm has an error rate of 17.8% on the *Franckenland* map and 5.4% on the *Circulus Franconicus* map. This experiment shows that the greedy algorithm is unsuitable for this matching problem, and that the result of our matching algorithm is indeed non-trivial. Also note that our algorithm has similar error rates on both maps, while the greedy algorithm performs significantly worse on the *Franckenland* map. This is due to the comparatively less clear labeling in this map (it has been created more than 170 years before the *Circulus Franconicus* map), which requires a higher combinatorial effort to understand.

Since our algorithm will be used as part of a semi-automatic digitization pipeline, we cannot assume its input to be absolutely accurate. This is especially true for the detection of labels (and areas of text in historical maps in general), where some characters can easily be missed by existing algorithms. For an example of the quality of label detection in historical maps, see Figure 22. In the next set of tests, we take this into account by introducing "position noise." Based on our ground truth data, we shifted each label by distance d_x in the x- and by distance d_y in the y-direction. The distances d_x and d_y were uniformly randomly chosen for each label such that $-d_{\text{noise}} \leq d_x \leq d_{\text{noise}}$ and $-d_{\text{noise}} \leq d_y \leq d_{\text{noise}}$. We have run the algorithm repeatedly with different values for d_{noise} on both maps; Figure 12 shows the results. Observe that our algorithm copes well with position noise as long as the distances by which labels are shifted are not too high. In particular, for $d_{\text{noise}} \leq 30\,\text{px}$, the error ratio stays below 4% on both maps. This is approximately the width of one to three characters in an average label in these maps. The error rate increases faster for the *Circulus Franconicus* map because the placement of its map elements is denser than in the *Franckenland* map.

6.2 Imbalanced Case

Next, we consider *imbalanced* input data. By this, we mean that the number of place markers $|P|$ is not necessarily equal to the number of labels $|L|$. In addition, not every label in L is actually corresponding to a marker in P in the ground truth and vice versa. This is a more realistic assumption for two reasons: First, our historical maps contain a

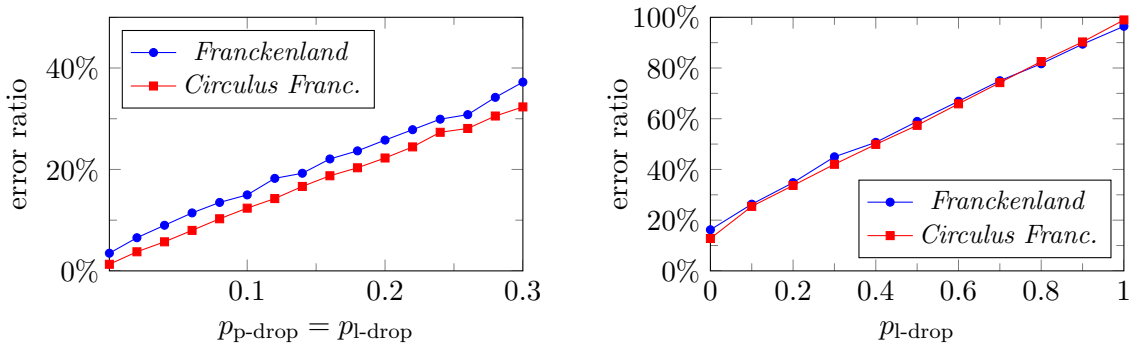

Figure 13: Impact of $p_{\text{p-drop}}$ and $p_{\text{l-drop}}$ on the error ratio in our algorithm. Note that $p_{\text{p-drop}}$ was fixed at 0.1 in the experiment depicted on the right.

small amount of unlabeled place markers and stray labels. Second, when integrating our approach into a (semi-) automated digitization process, the preceding detection modules are likely to miss some of the map elements.

We have run another set of experiments to assess the performance of our algorithm in such situations. For both the *Franckenland* and the *Circulus Franconicus* map, we use input data based on the manually created ground truth. Unlike for the tests in the previous section, we did not remove unlabeled place markers or stray labels from the data set. The input data based on the *Franckenland* map now contains 539 markers and 524 labels; 22 markers and 7 labels do not have counterparts. Our algorithm returns a matching of size 517, which contains 503 correct matches (Exp. F2). Of the 14 incorrect matches, 4 assign actually unlabeled markers and one a stray label, i.e. elements that should not have been assigned at all. The remaining 9 incorrect assignments involve only regular place markers and labels. Conversely, 6 out of 7 stray labels and 18 out of 22 unlabeled markers are correctly left unassigned. We calculate the error rate in the imbalanced case following the concept used in the balanced case, that is, evaluating the correctness of each assignment. However, there is now a second type of "assignments" in the ground truth: place markers and labels that are not assigned at all (assigned to "nothing"). The sum over the incorrect assignments of both types divided by the sum of assignments in the ground truth will be considered the error rate of our algorithm. In Exp. F2, the error rate is below 3.5%, while the required runtime was 1.0 s.

Based on the *Circulus Franconicus* map, we now have input data containing 1663 place markers and 1669 labels; 19 markers are unlabeled and 25 labels are stray. The matching obtained by our algorithm contains 1646 assignments, of which 1630 are correct (Exp. C2). The 16 incorrect assignments consist of 5 actually unlabeled markers that are matched to labels, 3 stray labels that are matched to otherwise labeled markers and 8 incorrectly matched regular markers and labels. The error rate in this experiment is 1.3%, while the required computation time was 2.2 s. Note that the error rate on this map is considerably lower than on the *Franckenland* map. As stated above, the labeling in the latter map requires a higher combinatorial effort, which is interfered with by adding stray map elements. To obtain comparison values, we have also run the greedy algorithm introduced in Section 6.1 on this instance. The returned matchings have error rate 17.8% on the *Franckenland* and 5.9% on the *Circulus Franconicus* map. We observe that in this setting, our approach again clearly outperforms the greedy algorithm. The obtained error rates increased in comparison to the balanced case, but we can still consider the results to be of high quality.

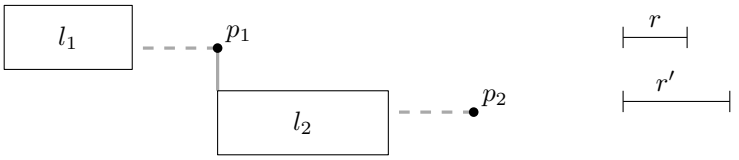

Figure 14: Using distance r, p_2 is too far away from l_2 to be considered, resulting in the wrong matching $M = \{(p_1, l_2)\}$ (solid gray). With distance r', our algorithm returns the correct matching $M' = \{(p_1, l_1), (p_2, l_2)\}$ (dashed gray).

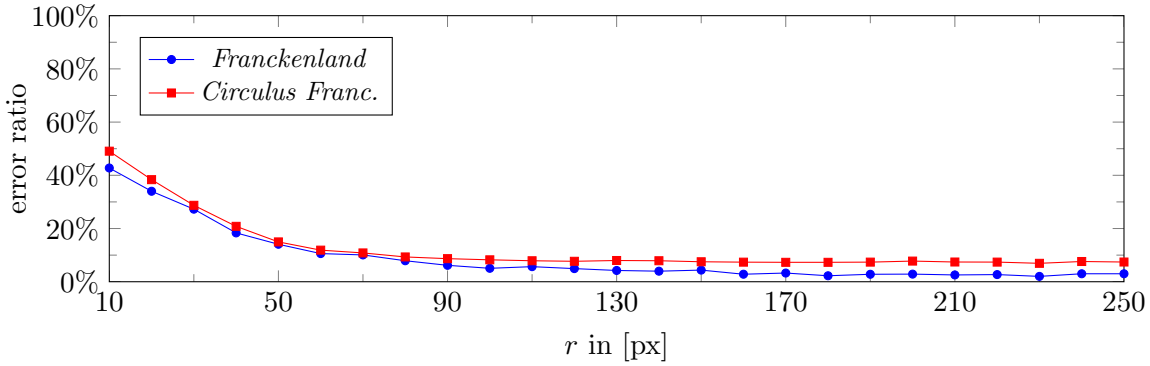

Figure 15: Impact of r on error ratio ($d_{\text{noise}} = 40\,px$)

In addition to the tests on the ground truth, we have also run experiments on incomplete data. Again, we expect that automatic detection modules, which could be deployed in preceding steps, will not detect all elements in the map correctly. In the next test setting, we assume that some place markers and labels have not been identified at all, thus missing in the input of our algorithm. We generated test data sets starting with the ground truth, removing elements from P with probability $p_{\text{p-drop}}$ and from L with probability $p_{\text{l-drop}}$. In several experiments, we varied values for $p_{\text{p-drop}}$ and $p_{\text{l-drop}}$ between 0% and 100%. On both maps, error rates increase linearly with $p_{\text{p-drop}}$ and $p_{\text{l-drop}}$; see Figure 13. Even with both probabilities at low values, the error rate increases (and cannot be contained, as it was in the position noise experiments). This is not surprising, as there remains no possibility of matching a pair of map elements correctly once at least one of them is removed from the input. The experiments show that our algorithm reacts sensitively on incomplete inputs. This must be taken into account when implementing the detection modules proposed in Section 3.1.

6.3 Parameter Choice

The high quality of the matching results relies to some extent on a reasonable choice of the parameter r. If r is chosen too small, our algorithm will not be able to assign labels that belong to markers with distance greater than r. Due to the combinatorial nature of the problem, this can also influence the assignment of markers and labels less than r apart, which is shown in Figure 14.

Picking a value for r that is too high can also lead to a decrease of the quality of our result. Consider a horizontally arranged series of marker-label pairs and an unlabeled place marker far to the right. Now assume that the left-most place marker was not detected correctly by a preceding system and is therefore not part of the input. If r is chosen too high, the whole assignment will "flip", propagating the error over all pairs and matching

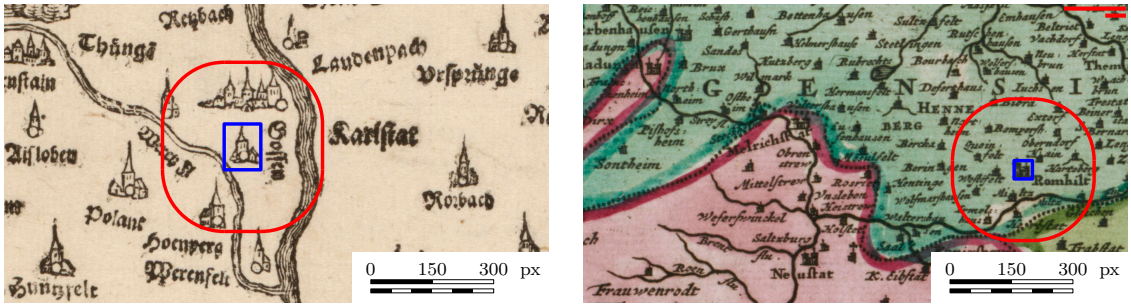

Figure 16: Scale in [px] on *Franckenland* and *Circulus Franconicus* map. The red boundaries mark a distance of 150 px from the blue marker bounding box.

the right most label to the far right place marker. However, if r is chosen correctly, the left most label and the far right place marker will both remain unmatched, leaving the assignments between intact.

Still, in experiments on real maps, our algorithm is not too sensitive to values of r that are picked exceedingly high. In fact, an experiment on balanced input data with fixed position noise $d_{\text{noise}} = 40$ px shows that error rates do not increase significantly for higher values of r (see Figure 15). In this experiment, even for very high values like $r = 1000$ px, the error rates stay at the same level as for $r = 150$ px. If r is chosen too small in this setting (i.e. below 60 px), error rates will greatly increase. However, a high value of r means that the flow network G contains more edges, which leads to an increased runtime. This is based on the fact that edges with $cost > r$ cannot be part of an optimal solution and can therefore be excluded from G. With $r = 150$ px, our algorithm needs 2.1 seconds to calculate an optimal solution for the *Circulus Franconicus* map (balanced case). For comparison, with $r = 1000$ px, it takes the algorithm 11.9 seconds to find the exact same matching. This is another reason why r should not be set to arbitrarily high values.

In our test maps, distances between corresponding labels and markers are typically limited by 2 or 3 times the average text height. This characteristic value of the map can easily be determined by the user. For our experiments, a value of $r = 150$ px was used for both maps. Figure 16 gives an overview of the scale (in pixels) of the *Franckenland* and *Circulus Franconicus* map and shows an area with distance smaller 150 px around a place marker. The dense placement of elements in the *Circulus Franconicus* map would also allow to set r to a lower value without affecting the returned matching, for example to 100 px.

7 Sensitivity Analysis and User Interaction

In both maps there exist situations in which it is unclear even to a human reader how place markers and labels belong together. Changing a single assignment in such situations can affect several other assignments; Figure 10 shows an example where it seems there are three feasible matchings. Note that in the displayed situation, without additional topographic or historic information, the correct assignment of markers and labels is unclear. To meet the high quality standards in digitization required for libraries and archives, it would be very useful to identify such situations and show the computed assignments to a user with domain knowledge for verification and correction.

7.1 Sensitivity-based Classification

As there are several hundred to several thousand assignments to be made, we do not want to show all of them to the user for verification. Instead, we develop a classifier that ranks the computed assignments by our algorithm's confidence into them and presents the T least certain assignments to the user for inspection (for some threshold T). As a measure of confidence for each assignment, we adapt the concept of sensitivity analysis introduced in Section 3.2 and define:

Definition 3 (Objective Sensitivity)**.** *Given an optimal solution M for the* Assign Labels *problem, an assignment $(p, l) \in M$ and an optimal solution M' for the* Assign Labels *problem under the additional constraint that $(p, l) \notin M'$. The* objective sensitivity value *of (p, l) is defined as the difference between $f_{obj}(M)$ and $f_{obj}(M')$. M' is called the* sensitivity matching *$sens_M((p, l)) = M'$ of (p, l) with regard to M.*

The objective sensitivity value of a match $(p, l) \in M$ therefore states how much higher its cost could be such that (p, l) is still in the optimal solution M. Equivalently, it indicates by how much the objective value would decrease if (p, l) was not allowed to be part of the matching. Hence, a high objective sensitivity value means a high confidence of our system into the specific match. Conversely, a low objective sensitivity value means that there is a matching almost as good as M that does not assign p to l. In this case, (p, l) must be considered uncertain. Our classifier sorts the assignments in ascending order of their objective sensitivity values and shows the first T assignments to the user for validation. We present a prototype of a graphical user interface for this in Section 7.3.

For the experiments in this chapter, we have implemented the sensitivity analysis and the described classifier: Starting with a matching M returned by our algorithm, we calculate the objective sensitivity values for each assignment (p, l) by removing the corresponding edge from the flow network G and calculating the new optimum M'. To speed up the computation of M', we use CPlex's "warm start" feature, which allows to start the algorithm solving the LP with the values of the optimal matching M. The difference between $f_{\text{obj}}(M)$ and $f_{\text{obj}}(M')$ is the objective sensitivity value of (p, l). Based on these sensitivity values and T, we sort and truncate the list of assignments in M; these are the assignments that will be presented to the user.

We have run experiments on both the *Franckenland* and the *Circulus Franconicus* map (with imbalanced input). To evaluate the performance of our classifier, we calculated the

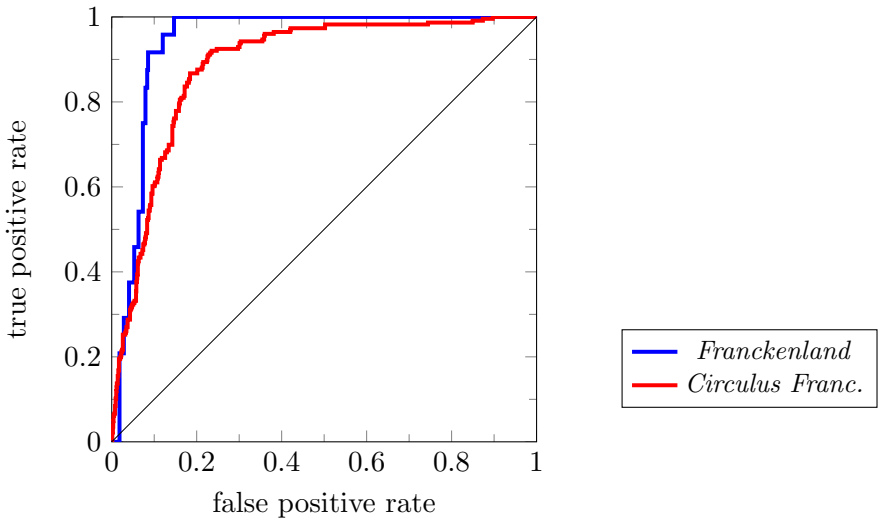

Figure 17: ROC curves for classifying assignments by their index ordered by objective sensitivity values. The area under curve (AUC) equals 0.89 and 0.94.

receiver operating characteristic (ROC) curve using ground truth data: see Figure 17. The concept of ROC curves is often used to visualize the performance of classifiers by showing false and true positive rates while varying their discrimination threshold T. Following the paper on ROC analysis by Fawcett (2006), we calculated the area under the ROC curves (AUC). Hosmer and Lemeshow (2004) state that (in general) an AUC value between 0.8 and 0.9 can be considered excellent, while values over 0.9 are outstanding. The AUC in our experiments is 0.94 for the *Franckenland* map and 0.88 for the *Circulus Franconicus* map. The runtime required to calculate the sensitivities and create the classifier was 3.7 seconds for the *Franckenland* and 78.5 seconds for the *Circulus Franconicus* map.

Next, we introduce errors to our test input by dropping map elements (as introduced in Section 6.2). We set $p_{\text{p-drop}} = 0.1$ and varied $p_{\text{l-drop}}$ between 0.1 and 0.3. On both maps, the AUC values for our classifier stay above 0.8. Based on balanced input, we have also made experiments introducing position noise. On both *Franckenland* and *Circulus Franconicus* maps, the AUC is above 0.8 with $d_{\text{noise}} = 70\,\text{px}$. For an extreme value of $d_{\text{noise}} = 150\,\text{px}$, the AUC value is still above 0.7 for the *Franckenland* map and above 0.6 for the *Circulus Franconicus* map. These experiments clearly show that our classifier is sufficiently robust against erroneous input data to be of practical value.

7.2 Interactive Postprocessing

A positive side effect of calculating the sensitivity analysis is that we obtain for each match $(p, l) \in M$ its sensitivity matching, i.e. an optimal assignment M' such that $(p, l) \notin M'$. When presenting an unclear match (p, l) identified by our classifier to the user, we can immediately show how the next-best matching would look like if (p, l) was indeed incorrect. This can be used to guide the user and improve the quality of our user interface. In Figure 18a, we show an example of how sensitivity results could be presented to the user. Our system's confidence in each assignment increases from red to green. Note that the depicted map contains the unclear situation from Figure 10 and the according assignments have been identified and displayed as uncertain by our sensitivity analysis. Figure 18b shows how we can instantly preview the next-best matching to the user in case he or she

(a) Assignments color-coded by sensitivity. (b) Preview of an alternative matching

Figure 18: Presentation of our system's confidence in each assignment to the user. Note that the unclear situation from Figure 10 has been identified and highlighted.

considers to reject an assignment (here the assignment under the mouse pointer). In the depicted situation, the next-best matching would only differ on three edges (dashed blue).

In fact, in some cases we can also quickly recalculate the objective sensitivity values of the assignments in the new matching M'. This is not trivial (recall that it takes more than a minute to perform our sensitivity analysis on the *Circulus Franconicus* map from scratch), but is crucial for a real-time interactive postprocessing system. Such a system would present assignments with low confidence to the user, let him or her decide whether they are correct or not and immediately show the resulting new matching with its sensitivities. A fast recalculation of sensitivities would also allow to reclassify the assignments and present the remaining most uncertain match to the user. Instead of recalculating sensitivity values for all $m \in M'$, we want to skip those matches in M' that we know to have the same sensitivity as in M. This is (by definition) the case if $f_{\text{obj}}(M) - f_{\text{obj}}(\text{sens}_M(m)) = f_{\text{obj}}(M') - f_{\text{obj}}(\text{sens}_{M'}(m))$. In the following, we discuss situations in which this equality holds.

When presenting a match $(p, l) \in M$ to the user, we consider two possible feedbacks: positive (l actually labels p in the historical map), or negative (p and l do not correspond). First, we want to focus on the positive case. If the user confirms the correctness of a match $(p, l) \in M$, we take the assignment of p and l as certain, do not want any different assignments for p or l in the future and therefore remove both from P and L. In this case, we can utilize our following results:

Lemma 4. *Let M be a solution for the* Assign Labels *problem, that is, a matching of place markers P and labels L that is optimal with respect to f_{obj}. If $(p, l) \in M$, then $M' = M \setminus \{(p, l)\}$ is an optimal matching of $P \setminus \{p\}$ and $L \setminus \{l\}$.*

Proof. We prove this claim by contradiction. By the definition of f_{obj}, each match in a matching adds one term to the total objective value. Thus, $f_{\text{obj}}(M) = (r - d(p, l)) + \sum_{(p',l') \in M'} (r - d(p', l'))$. Now assume M' is not an optimal matching of $P \setminus \{p\}$ and $L \setminus \{l\}$. Then there exists a matching M'' of $P \setminus \{p\}$ and $L \setminus \{l\}$ such that $f_{\text{obj}}(M'') > f_{\text{obj}}(M')$. By adding (p, l) to M'', we obtain an admissible matching of P and L with $f_{\text{obj}}(M'' \cup (p, l)) > f_{\text{obj}}(M)$. This is a contradiction to the optimality of M. □

Lemma 5. *If a match $(p, l) \in M$ is confirmed by the user and thus removed from the instance, then $M \setminus \{(p, l)\}$ is an optimal matching of $P \setminus \{p\}$ and $L \setminus \{l\}$ and no match in $M \setminus \{(p, l)\}$ has a lower objective sensitivity value in regard to $M \setminus \{(p, l)\}$ than it had in regard to M.*

Proof. Since $(p, l) \in M$, the matching $M \setminus \{(p, l)\}$ is an optimal matching of $P \setminus \{p\}$ and $L \setminus \{l\}$ (by Lemma 4). Let $(p', l') \in M \setminus \{(p, l)\}$ be arbitrarily chosen. We distinguish between two cases: first, assume $(p, l) \in \text{sens}_M((p', l'))$. The objective sensitivity value of (p', l') with regard to M is

$$f_{\text{obj}}(M) - f_{\text{obj}}(\text{sens}_M(p', l'))$$

(by Definition 3). Since (p, l) is element of both M and $\text{sens}_M(p', l')$, we can use the definition of f_{obj} and transform the term above into

$$(f_{\text{obj}}(M \setminus \{(p, l)\}) + (r - d(p, l))) - \left(f_{\text{obj}}(\text{sens}_{M \setminus \{(p,l)\}}(p', l')) + (r - d(p, l))\right),$$

which shows that

$$f_{\text{obj}}(M) - f_{\text{obj}}(\text{sens}_M(p', l')) = f_{\text{obj}}(M \setminus \{(p, l)\}) - f_{\text{obj}}(\text{sens}_{M \setminus \{(p,l)\}}(p', l')).$$

Due to Lemma 4, $\text{sens}_{M \setminus \{(p,l)\}}(p', l')$ is an optimal matching for $P \setminus \{p\}$ and $L \setminus \{l\}$ (under the constraint that p' is not matched to l'). Therefore, the objective sensitivity value of (p', l') does not change.

Now, assume that $(p, l) \notin \text{sens}_M((p', l'))$. In this case, we prove our claim by contradiction, so assume (p', l') does have a lower objective sensitivity value in regard to $M \setminus \{(p, l)\}$ than it had in regard to M. This means that there exists a matching $\text{sens}_{M \setminus \{(p,l)\}}(p', l')$ such that

$$f_{\text{obj}}(M) - f_{\text{obj}}(\text{sens}_M(p', l')) > f_{\text{obj}}(M \setminus \{(p, l)\}) - f_{\text{obj}}(\text{sens}_{M \setminus \{(p,l)\}}(p', l')).$$

Following Lemma 4, $f_{\text{obj}}(M \setminus \{(p, l)\}) = f_{\text{obj}}(M) - (r - d(p, l))$. We can thus transform the inequality stated above into

$$f_{\text{obj}}(\text{sens}_M(p', l')) - (r - d(p, l)) < f_{\text{obj}}(\text{sens}_{M \setminus \{(p,l)\}}(p', l')).$$

Applying Lemma 4 again, $f_{\text{obj}}(\text{sens}_{M \setminus \{(p,l)\}}(p', l')) = f_{\text{obj}}(\text{sens}_M(p', l')) - (r - d(p, l))$. This leads to the contradiction that $f_{\text{obj}}(\text{sens}_M(p', l')) > f_{\text{obj}}(\text{sens}_M(p', l'))$.

Since (p', l') was chosen arbitrarily, we have shown that no match in $M \setminus \{(p, l)\}$ has a lower objective sensitivity value in regard to $M \setminus \{(p, l)\}$ than it had in regard to M. □

Theorem 6. *Let $(p, l) \in M$ be a match that is confirmed by the user and thus removed from the instance. For all remaining matches $m \in M \setminus \{(p, l)\}$, if $(p, l) \in sens_M(m)$, then the objective sensitivity value of m does not change.*

Proof. Follows immediately from the proof of Lemma 5 (first case). □

Lemma 5 and Theorem 6 give insight into which sensitivity values stay the same once the user confirms a match. For our system, this means that sensitivity values only have to be recomputed for those matches whose sensitivity matchings did not contain the confirmed match previously. For all other matches, the sensitivity values are still correct and can be transfered to the new matching without additional computation cost. We conducted experiments on the *Franckenland* and the *Circulus Franconicus* map in order to measure the amount of sensitivity matchings that had to be recomputed for every confirmed match. In the experiments, we have used the unbalanced versions of both maps and simulated a confirmation of each assignment that was matched correctly. For the *Franckenland* map, on average 1.0 sensitivity matchings have to be recomputed for each edge; the median is 0 and the maximum number is 20. Similarly, for the *Circulus Franconicus* map, the average

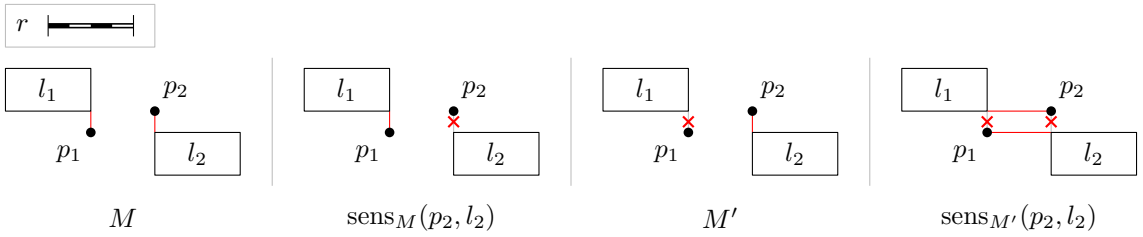

Figure 19: A situation in which the objective sensitivity value of (p_2, l_2) increases if (p_1, l_1) is rejected by the user.

of sensitivity values to be recomputed is 0.8, the median 1 and the highest number 7. These results show that, in practice, we can provide the user with an updated sensitivity analysis within less than a second once he or she confirms a match. This is fast enough to allow the implementation of a real-time interactive system.

Next, we discuss the negative case, in which the user rejects a presented match (p, l). In this case, we have to assume that p and l do not correspond and add a constraint that disallows matching them in the future. Again, we are interested in situations in which we do not have to recalculate sensitivity values in order to save computation time. We can state a theorem similar to Theorem 6 for this case; however, in practice, it rarely applies.

Theorem 7. *Let M be an optimal matching and let $(p, l) \in M$. Let M' be optimal among all matchings that do not contain (p, l). For all $m \in M$, if $(p, l) \notin sens_M(m)$, then $sens_M(m)$ is also a sensitivity matching for m with regard to M'.*

Proof. The matching $\text{sens}_M(m)$ already is an optimal solution for the Assign Labels problem under the constraint that it does not contain m (by definition) and (p, l) (by assumption). Thus, it is a sensitivity matching for m with regard to M'. □

Theorem 7 characterizes matches for which the recalculation of sensitivity matchings is not necessary given negative user feedback. However, its assumption rarely holds, as the following experiment shows. For every incorrect match (p, l) in an optimal solution, we analyze the number of sensitivity matchings that did not contain (p, l). Considering the *Franckenland* map, an incorrect match is not contained in 24 sensitivity matchings at best; the median is 4. For the *Circulus Franconicus* map, in the best case 5 sensitivity matchings do not contain a given incorrect match, while the median is 3. We clearly see that this is not a practical way to speed up sensitivity analysis significantly, as for both maps several hundred sensitivity matchings need to be recalculated.

In contrast to the positive case (and specifically Lemma 5), in the negative case objective sensitivity values might decrease after user interaction: an example is shown in Figure 19. In the depicted situation, we set parameter $r = 4$, while $d(p_1, l_1) = d(p_2, l_2) = 1$ and $d(p_1, l_2) = d(p_2, l_1) = 3$. An optimal solution M for the Assign Labels problem in this situation is $M = \{(p_1, l_1), (p_2, l_2)\}$, which has $f_{\text{obj}}(M) = 6$. To obtain the objective sensitivity value for (p_2, l_2), we consider the sensitivity matching $\text{sens}_M((p_2, l_2))$ with $f_{\text{obj}}(\text{sens}_M((p_2, l_2))) = 3$. The objective sensitivity value of (p_2, l_2) is thus $6 - 3 = 3$. Now consider a negative user feedback that disallows matching p_1 with l_1. In this case, a new optimal solution M' to the restricted instance is $M' = \{(p_2, l_2)\}$ with $f_{\text{obj}}(M') = 3$. The sensitivity matching for (p_2, l_2) is $\text{sens}_{M'}((p_2, l_2)) = \{(p_1, l_2), (p_2, l_1)\}$, which has $f_{\text{obj}}(\text{sens}_M((p_2, l_2))) = 2$. The objective sensitivity value of (p_2, l_2) is now $3 - 2 = 1$, which is a decrease compared to the original value.

Figure 20: Screenshot of a dialog box displayed by our QGIS plug-in. The presented match is highlighted in yellow.

The issues with the applicability of Theorem 7 as well as the possible decrease of objective values render the handling of negative user feedbacks a difficult, open problem. However, we can assume that most negative user feedback will in practice be accompanied by a correct alternative assignment. This is based on the observation that rejecting a given assignment normally requires an in-depth evaluation of its surroundings in the map. In this case, users can easily provide a new, locally adapted matching that does not include the rejected match. A proper solution to the problem of handling negative user feedback will have to take such input into account; this is an open problem.

7.3 Qualitative Discussion of Classified Matches

In the last section of this chapter, we qualitatively assess the performance of the classifier introduced above. In particular, we discuss the highest ranked matches in the *Franckenland* and the *Circulus Franconicus* map. To do this, we have implemented a prototype of our classification system, which provides a basic user interface. Technically, this was achieved by developing a plug-in for the QuantumGIS (QGIS) software package[1]. Provided with a matching M and the corresponding sensitivity analysis, our plug-in ranks each match in M by its objective sensitivity value. By highlighting each match and panning the map accordingly, the plug-in iteratively presents unclear matches to the user. A dialog box allows the user to give positive or negative feedback to each match; Figure 20 shows a screenshot of this system. We have used our system to examine the 20 highest-ranked matches for both maps. The situations in which they occur can be divided into the three different categories presented below.

Map Unclear In this type of situation, it is actually unclear regarding the map how place markers and labels correspond. This is often (but not only) the case if there exist unlabeled markers or stray labels in the neighborhood of the match. Figure 20 shows an unclear situation caused by an unlabeled marker: "Kalenberg" could label the marker to its left or the marker on top. To resolve the situation and find the correct assignment, the user needs additional geographic or historical knowledge. Of the situations presented here, this will be the most time-consuming to clarify for the user.

[1]QGIS is an open source geographic information system, see `http://www.qgis.org/`

(a) Presentation of an incorrect match (yellow). (b) Presentation of an actually correct match.

Figure 21: Matches presented by our plug-in to the user for verification (in yellow). Both were ranked within the 10 most sensitive matches of each map.

Incorrect Match In these situations, a place marker and a label that were incorrectly assigned will be presented to the user; this is a true positive in terms of our classification. Figure 21a gives an example for this: "Erelbach" was incorrectly matched with the marker to its bottom left instead of the (actually corresponding) marker on top. Note that this error propagates over several other matches to the bottom left corner of the displayed map section. Once the user corrects the highlighted match, these matches will also "flip" and assign the correct map elements. In general, the human evaluation process of such situations allows to easily find a new, correct assignment of the affected elements (instead of only rejecting the given match).

False Positive The last type of situations causes our system to present actually correct matches, which are false positives in terms of our classification. This can be due to various reasons; we identify here two relatively common causes. Consider the situation shown in Figure 21b: the assignment of "Oellingen" was highlighted because our algorithm is uncertain if the label corresponds to the marker to the right or the (stray) marker to the bottom right. For a human, the correct correspondence is obvious for two reasons. First, the marker to the right is horizontally aligned with the label, which indicates that they belong together. Second, the style of the lettering hints that a relatively large place is labeled, which also suggests the marker to the right. Such situations can quickly be resolved by the user.

Based on the three types of situations described above, we propose several improvements to our system that could be implemented in the future. For actually unclear situations, it could be helpful for the user to display a modern map next to the historical map for additional geographic background. In case that an incorrect match is presented, it should be possible to provide an alternative, corrected matching through the user interface. Our matching algorithm is able to process such input; however the real-time recalculation of sensitivity values is not yet solved, as explained in the previous section. To improve the quality of the matching and to have the classifier produce less false positives, the two common issues identified above can be approached. First, the distance measure for map elements could be adjusted to favor horizontal alignments. Second, the correspondence of font style and size of corresponding markers could be analyzed and also be taken into account, for example by modifying f_{obj}.

8 Matching Markers with Sets of Labels

In the preceding chapter we have discussed a way to improve the quality of matching results by including user feedback. Now we present a completely different approach to improve the matching, which is to refine our optimization model. Based on the outputs of the label detection approaches by Höhn (2013) and our own experiments[1], we observe that often parts of text that belong to a single label will be detected as separate labels; for an example from our experiments, see Figure 22.

This is partially due to the shortcomings of these algorithms, but also introduced by split labels and inconsistent label placement in the historical maps. However, separately detected text elements that actually form one "logical" label in the map are still located relatively close to each other. We want to leverage this property of our input data to improve our matching results and adapt our optimization problem formulation introduced in Chapter 5.

In order to identify detected labels that actually form a single label in the map, we propose using a heuristic that constructs a family of sets containing labels that possibly belong together. For instance, an appropriate heuristic could be to put labels that are located within a certain distance from each other into one set. On maps that contain mostly horizontal text, one might want to restrict the elements of a set to labels that are aligned horizontally. However, this would not take into account labels that are actually split vertically, for example using a hyphen.

In Figure 22, we see an exemplary family $\mathcal{F}$ of sets that a heuristic could return in the given map situation. Note that we allow labels to be contained in more than one set to deal with unclear situations, for example at the constellation of l_9, l_{10} and l_{12} in the lower right. Labels that do not have other labels in their near neighborhood can as well be the only element in a set, see l_8.

[1] Budig, B., Chlechowitz, M., Kauer, J., Löffler, A., and Wisheckel, F. (2014). Abschlussbericht zum Projekt Texterkennung auf historischen Landkarten. Universität Würzburg.

Figure 22: Labels detected with our own approach used for preliminary experiments. Note that many labels have (incorrectly) been detected as several separate text areas. On the right, we see an example of sets of text areas that might belong together.

8.1 Optimization Problem

Based on the family of sets $\mathcal{F}$ returned by some heuristic, we assume that all elements l in a set $S \in \mathcal{F}$ are likely to form in fact a single label, which should only be matched by a single place marker. This allows us to later interpret assigning one label from a set as assigning the entire group. Recall the three goals introduced in Section 5.1:

$$\text{No match } (p,l) \in M \text{ has distance } d(p,l) > r. \tag{8.1}$$

$$\text{The sum over } d(p,l) \text{ for all } p,l \in M \text{ is small.} \tag{8.2}$$

$$\text{The size of the matching } M \text{ is large.} \tag{8.3}$$

Since we assume that the labels within a set S actually form a single label in the map, we do not want to split S by matching more than one label in S to a place marker. We add a fourth goal which takes this into account:

$$\text{At most one label } l \text{ from each } S \in \mathcal{F} \text{ should be matched in } M. \tag{8.4}$$

We combine these four goals into the following new objective function:

$$f_{\text{obj}}(M) = \sum_{(p,l)\in M} (r - d(p,l)) - \sum_{S\in\mathcal{F}} c(S) \tag{8.5}$$

where $c : S \to \mathbb{R}$ is a cost function that penalizes matching more than one label from each set S. We define

$$c(S) = \begin{cases} 0 & \text{if } |M \cap S| \leq 1 \\ \mathit{penalty} & \text{otherwise} \end{cases} \tag{8.6}$$

and still want to maximize f_{obj} under the constraint that M is a matching. By choosing a positive value for *penalty*, each assignment $(p,l) \in M$ with $l \in S$ lowers the matching value if M contains at least one other assignment (p',l') with $l' \in S$. We call this the Assign Label Sets problem.

8.2 ILP and Proof of NP-Hardness

We can formulate this problem as an integer linear program (ILP), so we can solve it. Let $x_{p,l} \in \{0,1\}$ be decision variables that indicates whether (p,l) is in M or not. Let furthermore $y_S \in \{0,1\}$ be decision variables indicating if more than one element of S is part of the matching M. Let $w(p,l) = r - d(p,l)$. Considering our objective function above, we arrive at the following ILP:

$$\begin{aligned}
\text{maximize} \quad & \sum_{(p,l)\in M} x_{p,l} \cdot w(p,l) - \sum_{S\in\mathcal{F}} y_S \cdot c(S) & \\
\text{subject to} \quad & \sum_{p\in P} x_{p,l} \leq 1 & \forall l \in L \\
& \sum_{l\in L} x_{p,l} \leq 1 & \forall p \in P \\
& \sum_{l\in S} \sum_{p\in P} x_{p,l} \leq 1 + y_S \cdot |L| & \forall S \in \mathcal{F}
\end{aligned}$$

The first two constraints demand that each marker and each label is assigned at most once, which guarantees that the result is indeed a matching. The third constraint forces y_S to be 1 if more than one label in S is matched, thus applying the penalty to the objective value. Interpreting the decision variables $x_{p,l}$, we get $M = \{(p, l) \mid x_{p,l} = 1\}$.

There exist several algorithms that solve integer linear programs; however, none of them calculates a solution *efficiently*. In fact, 0-1 integer linear programming belongs to the list of 21 NP-complete problems stated by Karp (1972). Still, the ILP given above has practical relevance, as it can be used for testing on small instances and to gain deeper insight into the formulated optimization problem. We now show that, assuming P $\neq$ NP, there is no way to solve the Assign Label Sets problem efficiently, as the problem is NP-hard. Our proof uses a polynomial-time reduction from the NP-complete Set Packing problem to a decision variant of the Assign Label Sets problem; the following definition is taken from Karp (1972):

Definition 8 (Set Packing)**.** *Given a family of sets $\{S_j\}$ and a positive integer k, decide whether $\{S_j\}$ contains k mutually disjoint sets.*

Definition 9 (Assign Label Sets)**.** *Given a set of place markers P of size k, a set of labels L of at least the size of P, a family $\mathcal{F}$ of subsets of L, a weight function $w(p, l)$, and a cost function $c(S)$. Decide whether there exists an optimal solution M to the* Assign Label Sets *optimization problem such that $f_{obj}(M) \geq 0$.*

Theorem 10. *The* Assign Label Sets *is NP-complete.*

Proof. We show that the Set Packing problem is polynomial-time reducible to the Assign Label Sets decision problem, i.e. Set Packing $\leq_p$ Assign Label Sets.

Let $f(\{S_j\}, k) = (P, L, \mathcal{F}, w(p, l), c(S))$ be a polynomial-time reduction function, defined as follows: Consider the elements in $\bigcup_j S_j$ to be the set of labels L. Use the family of sets $\{S_j\}$ as $\mathcal{F}$. Introduce k place markers, which form the set P and let $w(p, l) = 0$ for all $p \in P$ and $l \in L$. For $c(S)$, use the function defined in equation (8.6) with $penalty = 1$.

Now, assume $\{S_j\}$ contains k mutually disjoint sets. Then there are also k mutually disjoint sets of labels in $\mathcal{F}$. Matching each of the k place markers to an arbitrary label from a different disjoint set in $\mathcal{F}$ yields an optimal solution M for the Assign Label Sets optimization problem. Since at most one element from each set in $\mathcal{F}$ was matched, no penalties were applied and $f_{\text{obj}}(M) = 0$. However, all place markers have been matched, so M is an optimal solution with $f_{\text{obj}}(M) = 0$.

For the other direction, suppose there is an optimal solution M to the Assign Label Sets optimization problem such that $f_{\text{obj}}(M) = 0$ and $|P| = k$. Since $|P| = k$, k labels were matched, and since $f_{\text{obj}}(M) = 0$, at most one label from each set in $\mathcal{F}$ was matched. The family of sets thus contains k mutually disjoint sets.

By polynomial-time reduction from Set Packing to the Assign Label Sets decision problem, we have shown that the latter is NP-hard. The problem is trivially in NP. □

8.3 Polynomial-Time Algorithm for a Restricted Problem

Since the Assign Label Sets problem can be considered infeasible, we want to focus on a restricted version of the problem. Recall that in the general version of the problem stated above, we allowed labels to be elements of more than one set. If we instead require that each label is only element of one set, we can state a polynomial time solution for the thus restricted Assign Label Sets (disjoint) problem.

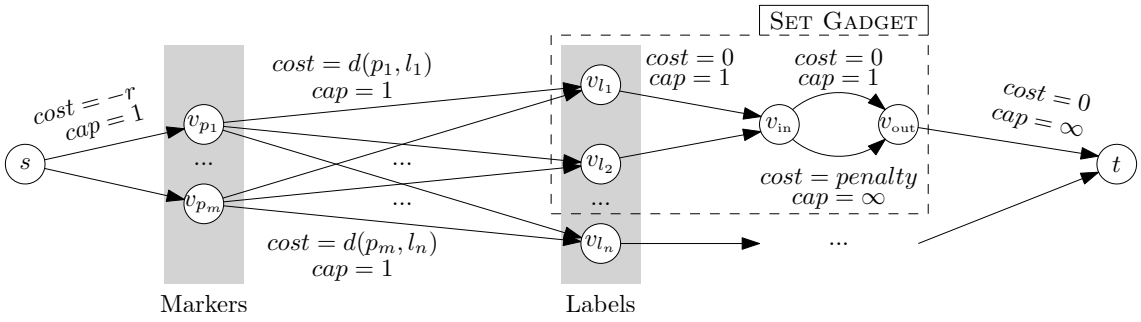

Figure 23: Flow network from Section 5.2, augmented with *set gadget* (dashed box).

Definition 11 (Assign Label Sets (disjoint))**.** *Given a set of place markers P, a set of labels L, a family $\mathcal{F}$ of* disjoint *subsets of L, a weight function $w(p, l)$, and a cost function $c(S)$. Find an optimal solution M to the* Assign Label Sets *problem.*

Partitioning the labels seems reasonable; for example in the situation in Figure 22, we can easily find a partition of detected labels that are likely to belong together. This is particularly the case under the assumption that detected text areas belong together horizontally, which is indeed how the text detection module seems to fail. We can meet the new requirement by applying a different heuristic to the labels of the input, which packs them into disjoint sets. This could for example be done in a greedy fashion, where we still take the distances between the labels into account.

Theorem 12. Assign Label Sets (disjoint) *can be solved in polynomial time.*

Proof (sketch). We solve the Assign Label Sets (disjoint) problem by augmenting the flow network introduced for solving the Assign Labels problem; see Section 5.2 for the original. For every set S returned by our heuristic, we introduce a *set gadget* to the flow network; Figure 23 shows an example. Our construction guarantees that one unit of flow can pass the gadget without increasing costs, whereas every additional unit of flow increases total costs by *penalty*.

Every gadget consists of two additional vertices, v_{in} and v_{out}. The entrance vertex v_{in} can be reached from all labels that belong to the set corresponding to the gadget by directed edges E_{in}. For every edge $e \in E_{\text{in}}$, we set $cost(e) = 0$ and $cap(e) = 1$. This capacity constraint guarantees that every label is matched at most once. From v_{in} to v_{out}, there are two directed edges e_{free} and e_{penalty}, where $cost(e_{\text{free}}) = 0$ and $cap(e_{\text{free}}) = 1$, while $cost(e_{\text{penalty}}) = penalty$ and $cap(e_{\text{penalty}}) = \infty$. The exit vertex v_{out} is connected to sink t with a directed edge e_{out} of $cost(e_{\text{out}}) = 0$ and $cap(e_{\text{out}}) = \infty$.

The amount of flow units that enter each gadget is equal to the amount of labels in S that were matched. This corresponds directly to the behavior of the cost function $c(S)$ defined in (8.6). The augmented flow network thus correctly models the Assign Label Sets (disjoint) problem. Using this flow network, we can apply the method introduced in Section 5.2 to solve the problem in polynomial time. □

Conclusion and Future Work

In the present thesis, we have introduced algorithmic approaches that allow the extraction of information from historical maps. We started by identifying common problems and particularly time-consuming tasks in current map digitization workflows. In a critical review of existing related systems, we observed that most tasks still need to be performed manually even with the help of these systems. With an algorithmically assisted metadata extraction system, the time required to manually analyze a historical map could be significantly reduced. We sketched such a system, described the necessary modules and proposed algorithmic approaches for their implementation.

Afterwards, we showed in a case study that the realization of one of these modules is feasible. We introduced an algorithm for matching place markers and labels and assessed its performance in several experiments on historical maps. Its high-quality results could be further improved by user interaction; to guide the user, the algorithm performs a sensitivity analysis on the resulting matching. In addition, we introduced an approach to better handle imperfect input from preceding text detection modules. As a part of the proposed system, we expect that this module can save a considerable amount of work for the digitization experts.

For future work, we plan to extend the graphical user interface for the matching algorithm introduced in this thesis. Furthermore, the other modules in the proposed information retrieval system need to be implemented. This will be addressed as part of a subsequent research project. In addition, we want to further develop the ideas presented in the technical outlook in Chapter 4. For example, it might be worth implementing a crowdsourcing-based application that will help dealing with difficult tasks in our information retrieval pipeline. We also want to investigate if the sensitivity concept introduced in this thesis is broadly applicable to additional modules of our system.

Finally, we stay in touch with the digitization experts at the University Library in Würzburg to get feedback and valuable suggestions for further improvements. The information retrieval pipeline described here could in the future be integrated into their digitization workflow system.

Appendix

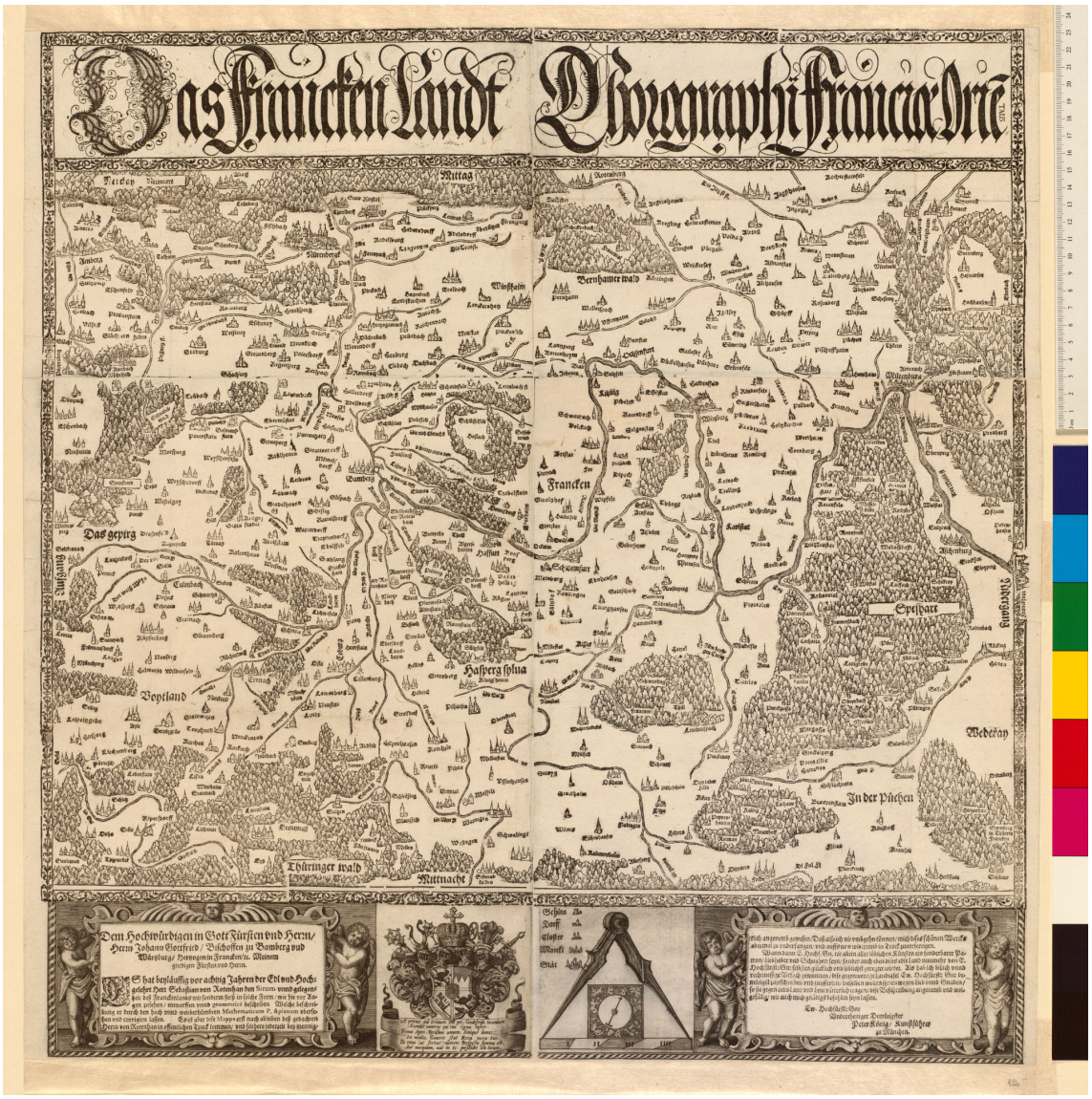

Figure 24: Sebastian von Rotenhan. *Das FranckenLandt = Chorographi Franciae Orien-[talis]*, 1533.

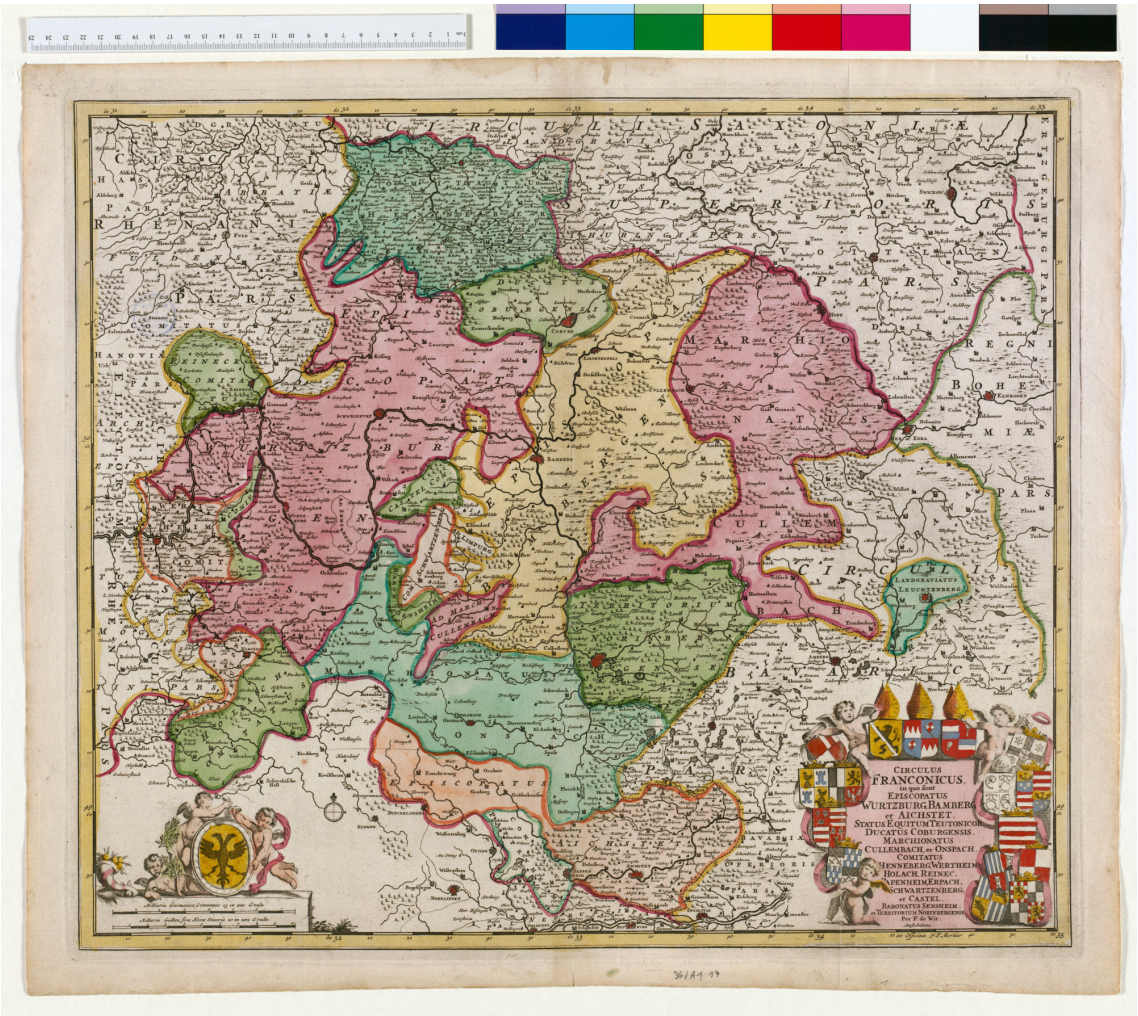

Figure 25: Frederik De Wit. *Circulus Franconicus: in quo sunt episcopatus Wurtzburg, Bamberg et Aichstet, Status Equitum Teutonicor(um), Ducatus Coburgensis, Marchionatus Cullembach et Onspach, Comitatus Henneberg, Wertheim, Holach, Reinec, Papenheim, Erpach, Schwartzenberg, et Castel, Baronatus Sensheim et Territorium Norinbergense*, 1706.

Bibliography

Arteaga, M. G. (2013). Historical Map Polygon and Feature Extractor. In *Proceedings of the 1st ACM SIGSPATIAL International Workshop on MapInteraction*, MapInteract '13, pages 66–71.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2009). *Introduction to Algorithms*. The MIT Press, 3rd edition.

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern recognition letters*, 27(8):861–874.

Fleet, C., Kowal, K. C., and Pridal, P. (2012). Georeferencer: Crowdsourced Georeferencing for Map Library Collections. *D-Lib Magazine*, 18(11/12).

Goldberg, A. V. (1992). An Efficient Implementation of a Scaling Minimum-Cost Flow Algorithm. *Journal of Algorithms*, 22:1–29.

Höhn, W., Schmidt, H.-G., and Schöneberg, H. (2013). Semiautomatic Recognition and Georeferencing of Places in Early Maps. In *Proceedings of the 13th ACM/IEEE-CS Joint Conference on Digital Libraries*, JCDL '13, pages 335–338.

Hosmer, D. W. and Lemeshow, S. (2004). *Applied Logistic Regression*. John Wiley & Sons.

Höhn, W. (2013). Detecting Arbitrarily Oriented Text Labels in Early Maps. In *Proceedings of the 6th Iberian Conference on Pattern Recognition and Image Analysis*, volume 7887 of *LNCS*, pages 424–432.

Jenny, B. and Hurni, L. (2011). Cultural Heritage: Studying Cartographic Heritage: Analysis and Visualization of Geometric Distortions. *Computers & Graphics*, 35(2):402–411.

Karp, R. M. (1972). Reducibility Among Combinatorial Problems. In *Complexity of Computer Computations*, pages 85–103.

Kuhn, H. W. (1955). The Hungarian Method for the Assignment Problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97.

Leyk, S., Boesch, R., and Weibel, R. (2006). Saliency and semantic processing: Extracting forest cover from historical topographic maps. *Pattern Recognition*, 39(5):953–968.

Liu, L. and Shell, D. A. (2011). Assessing Optimal Assignment Under Uncertainty: An Interval-based Algorithm. *The International Journal of Robotics Research*, 30(7):936–953.

Mello, C. A. B., Costa, D. C., and dos Santos, T. J. (2012). Automatic Image Segmentation of Old Topographic Maps and Floor Plans. In *Proceedings of the 2012 IEEE International Conference on Systems, Man, and Cybernetics*, SMC '12, pages 132–137.

Postel, H. J. (1969). Die Kölner Phonetik – Ein Verfahren zur Identifizierung von Personennamen auf der Grundlage der Gestaltanalyse. *IBM-Nachrichten*, 19:925–931. As cited in Höhn et al. (2013).

Schöneberg, H., Schmidt, H.-G., and Höhn, W. (2013). A Scalable, Distributed and Dynamic Workflow System for Digitization Processes. In *Proceedings of the 13th ACM/IEEE-CS Joint Conference on Digital Libraries*, JCDL '13, pages 359–362.

Schrijver, A. (2003). *Combinatorial Optimization: Polyhedra and Efficiency*. Springer.

Schuppert, C. and Dix, A. (2009). Reconstructing Former Features of the Cultural Landscape Near Early Celtic Princely Seats in Southern Germany. *Social Science Computer Review*, 27(3):420–436.

Shaw, T. and Bajcsy, P. (2011). Automation of Digital Historical Map Analyses. In *Proceedings of the IS&T/SPIE Electronic Imaging 2011*, volume 7869.

Simon, R., Haslhofer, B., Robitza, W., and Momeni, E. (2011). Semantically Augmented Annotations in Digitized Map Collections. In *Proceedings of the 11th Annual International ACM/IEEE Joint Conference on Digital Libraries*, JCDL '11, pages 199–202.

Hiermit versichere ich, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen Hilfsmittel und Quellen als die angegebenen benutzt habe. Weiterhin versichere ich, die Arbeit weder bisher noch gleichzeitig einer anderen Prüfungsbehörde vorgelegt zu haben.

Würzburg, den _____, _____

(Benedikt Budig)