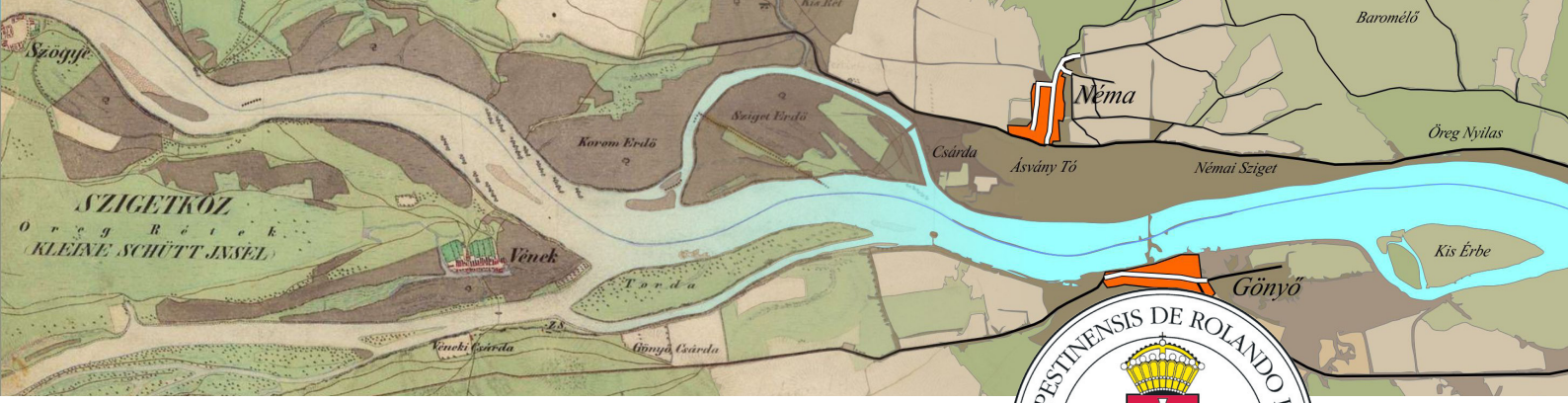


ICA ACI

Cartoheritage into the Digital

UNIVERSITAS BUDAPESTINENSIS DE ROLANDO EÖTVÖS NOMINATA 16 35

AUTOMATIC VECTORISATION OF HISTORICAL MAPS

International workshop organized by the ICA Commission on Cartographic Heritage into the Digital

13 March, 2020
Budapest

Department of Cartography and Geoinformatics
ELTE Eötvös Loránd University

SZÉCHENYI 2020
HUNGARIAN GOVERNMENT
European Union
European Social Fund
INVESTING IN YOUR FUTURE

AUTOMATIC VECTORISATION OF HISTORICAL MAPS
International workshop organized by the ICA Commission on Cartographic Heritage into the Digital

Budapest – 13 March, 2020

Conference board
Scientific committee
Mátyás Gede (ELTE Eötvös Loránd University, Budapest)
Angeliki Tsorlini (Aristotle University, Thessaloniki)
Krisztina Irás (ELTE Eötvös Loránd University, Budapest)
Tomasz Panecki (University of Warsaw/Polish Academy of Sciences)
Evangelos Livieratos (Aristotle University, Thessaloniki)

Local organising committee
Mátyás Gede (ELTE Eötvös Loránd University, Budapest)
Krisztina Irás (ELTE Eötvös Loránd University, Budapest)

Venue: Lágymányos Campus Northern Block, ELTE Eötvös Loránd University
1117 Budapest, Pázmány Péter sétány 1/a.
Contact: avhm.workshop@gmail.com
Info: http://lazarus.elte.hu/avhm/

Proceedings edited by Krisztina Irás, Department of Cartography and Geoinformatics,
ELTE Eötvös Loránd University, 2020, Budapest

The Project is supported by the Hungarian Government and co-financed by the European Social Fund.
EFOP-3.6.3-VEKOP-16-2017-00001: Talent Management in Autonomous Vehicle Control Technologies

Contents

Preface 5

Programme 6

DROLIAS GARYFALLOS CHRYSOVALANTIS, TZIOKAS NIKOLAOS

Building Footprint Extraction from Historic Maps utilizing Automatic Vectorisation Methods in Open Source Gis Software 9

MARGUERITE LE RICHE

Identifying Building Footprints in Historic Map Data using Opencv and Postgis 18

MÁTYÁS GEDE, VALENTIN ÁRVAI, GERGELY VASSÁNYI, ZSÓFIA SUPKA, ENIKŐ SZABÓ, ANNA BORDÁCS, CSABA GERGELY VARGA, KRISZTINA IRÁS

Automatic Vectorisation of Old Maps using Qgis – Tools, Possibilities and Challenges 37

ANNA PIECHL

(Semi-)Automatic Vector Extraction of Administrative Borders from Historical Raster Maps 45

ELŐD BISZAK, GÁBOR TIMÁR

First Vector Components in the Mapire: 3D Building Modelling and Virtual Reality View .. 53

DANIEL LAUMER, HASRET GÜMGÜMCÜ, MAGNUS HEITZLER, LORENZ HURNI

A Semi-Automatic Label Digitization Workflow for the Siegfried Map 55

KENZO MILLEVILLE, STEVEN VERSTOCKT, NICO VAN DE WEGHE

Improving Toponym Recognition Accuracy of Historical Topographic Maps 63

ALEXANDRE NOBAJAS

Targeted Crowdsourced Vectorisation of Historical Cartography 73

JONAS LUFT

Automatic Georeferencing of Historical Maps by Geocoding 75

GEOFF GROOM, GREGOR LEVIN, STIG SVENNINGSEN, MADS LINNET PERNER

Historical Maps – Machine Learning Helps Us over the Map Vectorisation Crux 89

Preface

Historical maps are important sources of information about past ages. Exploring ways of making this information available for further research is among the main tasks of the *Commission on Cartographic Heritage into the Digital* of the *International Cartographic Association (ICA).* In the first part of the Commission's fifteen year long history, the main focus was on georeferencing those old maps in order to be able to embed them in GIS systems for further analysis. Later more and more papers started to discuss the methods of extracting information from these documents, mainly by manual vectorisation.

Nowadays, dramatic increase of computing power and advances in the field of computer vision make possible to partially or fully automatize map vectorisation process. But historical maps raise special challenges for these efforts – the peculiarities of those old documents often trick algorithms that work well on recent maps. More and more researchers work on finding solutions for this automatization problem, and there is a demand to discuss the results with fellow researchers.

Recognising this need, the Commission organised an international workshop entitled "Automatic Vectorisation of Historical Maps" in Budapest, 13 March, 2020.
This volume contains the papers submitted to the workshop and shows the diversity of problems in this field. While there are several promising results, the need of further research and discussion is also inevitable.

We hope that readers of this volume will find answers, ideas and inspiration for their work in the realm of cartographic heritage.

Mátyás Gede
chair, ICA Commission on
Cartographic Heritage into the Digital

Krisztina Irás
local organizer, editor

Programme

9:00 Opening

Mátyás GEDE, Chair, ICA Commission on Cartographic Heritage to the Digital
Zoltán HORVÁTH, Dean, Faculty of Informatics, ELTE
László ZENTAI, Vice President, ICA

9:30 Coffee break

10:00 Session 1. Chair: Tomasz PANECKI

10:00 Drolias Garyfallos CHRYSOVALANTIS, Tziokas NIKOLAOS:
Building footprint extraction from historic maps utilizing automatic vectorization methods in open source GIS software

10:15 Marguerite LE RICHE:
Identifying building footprints in historic map data using OpenCV and PostGIS

10:30 Chenjing JIAO, Magnus HEITZLER, Lorenz HURNI:
Extracting Wetlands from Swiss Historic Maps with Convolutional Neural Networks

10:45 Mátyás GEDE, Valentin ÁRVAI, Gergely VASSÁNYI, Zsófia SUPKA, Enikő SZABÓ, Anna BORDÁCS, Csaba Gergely VARGA, Krisztina IRÁS:
Automatic vectorisation of old maps using QGIS – tools, possibilities and challenges

11:00 Anna PIECHL:
(Semi-)automatic vector extraction of administrative borders from historical raster maps

11:15 Előd BISZAK, Gábor TIMÁR:
First vector components in the MAPIRE: 3D building modelling and Virtual Reality view

11.30 Discussion

12:00 Lunch break

13:00 Session 2. Chair: Krisztina IRÁS

13:00 Daniel LAUMER, Hasret GÜMGÜMCÜ, Magnus HEITZLER, Lorenz HURNI:
A semi-automatic Label Digitization Workflow for the Siegfried Map

13:15 Kenzo MILLEVILLE, Steven VERSTOCKT, Nico VAN DE WEGHE:
Improving Toponym Recognition Accuracy of Historical Topographic Maps

13:30 Alexandre NOBAJAS:
Targeted crowdsourced vectorisation of historical cartography

13:45 Jonas LUFT:
Automatic Georeferencing of Historical Maps by Geocoding

14:00 Geoff GROOM, Gregor LEVIN, Stig SVENNINGSEN, Mads Linnet PERNER:
Historical Maps – Machine learning hel[illegible]er the map vectorisation crux

14:15 Discussion

15:00 Technical visit to Arcanum Ltd.

Drolias Garyfallos Chrysovalantis[1], Tziokas Nikolaos[2]

Building footprint extraction from historic maps utilizing automatic vectorisation methods in open source GIS software

Keywords: automatic vectorisation; historical maps; open source GIS; shape recognition

Summary: Historical maps are a great source of information about city landscapes and the way it changes through time. These analog maps contain rich cartographic information but not in a format allowing analysis and interpretation. A common technique to access this information is the manual digitization of the scanned map but it is a time-consuming process. Automatic digitization/vectorisation processing generally require expert knowledge in order to fine-tune parameters of the applied shapes recognition techniques and thus are not readily usable for non-expert users. The methodology proposed in this paper offers fast and automated conversion from a scanned image (raster format) to geospatial datasets (vector format) requiring minimal time for building footprint extraction. The vector data extracted quite accurate building footprints from the scanned map, but the need for further post-processing for optimal results, is imminent.

Introduction

Historical maps contain a lot of information about a region in a certain period, depict the condition of a settlement and toponyms of the time the map was created. Not only those maps are a part of our cultural heritage, but some of them are of great interest to researchers from various academic fields. Maps record the geographical information that is essential to reconstruct past places, town-wide, region-wide or even nation-wide depending on the map's scale and extent. These maps often contain information retained by no other written source, such as toponyms, boundaries, and geomorphological features that have undergone any type of change or even elimination through time. A map's degree of accuracy is a genuine source of information about the state of technology and scientific understanding at the time of its creation. By incorporating information from historical maps and open-source GIS software, researchers are now able to extract a vast amount of information, analyze and interpret with other geospatial data.

Most of those maps are found in analogue form, thus they need to be processed in order to extract geospatial data for further analysis and interpretation. Scanning of historical maps is usually the first step towards converting them in a format allowing analysis in GIS platforms. When a map is scanned, a raster image is created. Raster images are a matrix of pixels with each pixel having a data value. A raster image can be black-and-white, containing a single band, or colored containing multiple bands (red, green, blue). A black-and-white image often is a binary image, with each pixel having the value of 1 or 0, but a multicolor image means that each pixel has a data value for each band. Conversely, vector data are in the form of points, lines, and areas (polygons). Points represent discrete locations, lines represent discrete linear features, and polygons represent discrete bounded areas. Raster images resulting from scanning methods need to be converted in vector format in order to extract further information from the historic map.

[1] University of the Aegean [akisd@hotmail.com]
[2] University of the Aegean [nikos.tziokas@gmail.com]

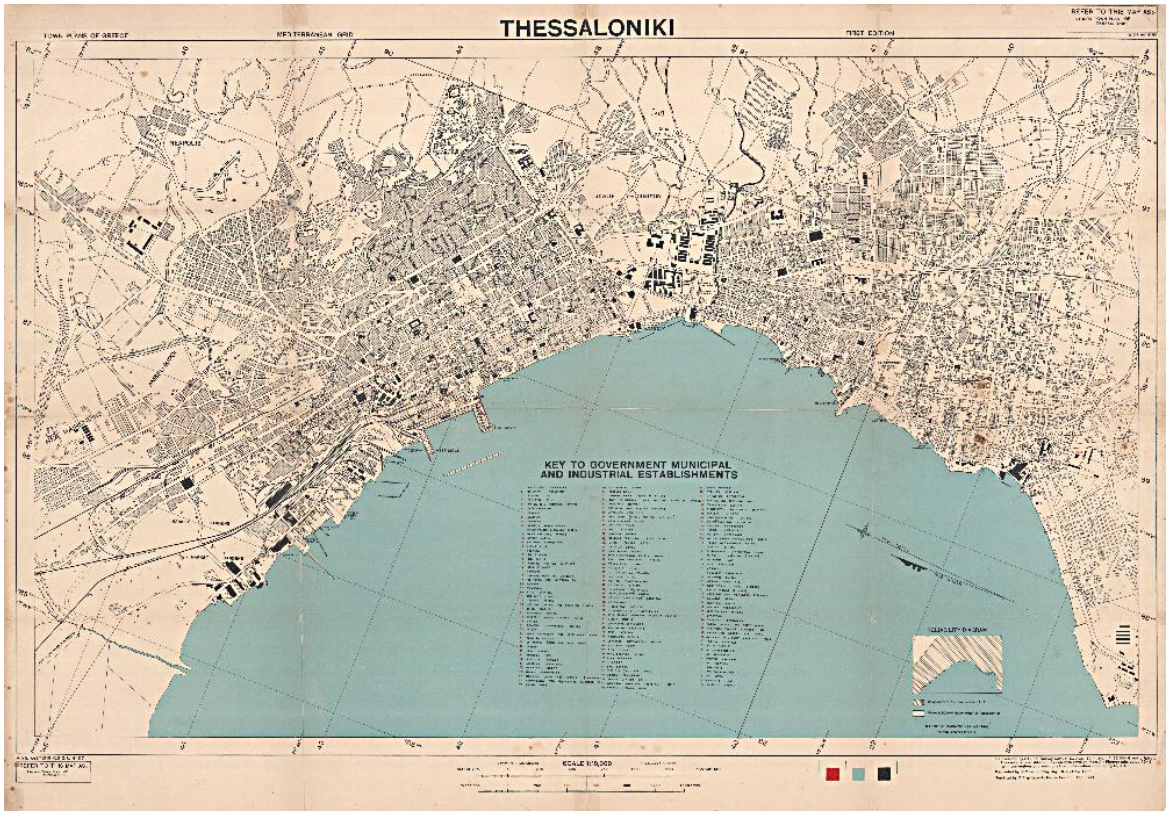


Figure 1. Original scanned map

This digitization method usually is the most challenging part as it can often lead to poor results, depending in the map quality, conversion types and methodologies applied to the initiative raster image.

In this paper, a new methodology is proposed based on free and open source GIS platforms in order to convert scanned (raster images) historical maps to manageable geospatial data (vector format). The basic stages of this procedure are the scanning and the correct georeference of the map, the pre-processing and the removal of pixel clumps in the raster image so that building footprints are extracted more efficiently. The second part of our methodology focuses on correcting the geometry of the converted vector data. Finally, a model containing all the processing steps and algorithms is developed in an open source GIS platform. This methodology has been applied on a single historic map, although it could be effective in any map containing information of a settlement, where buildings are represented as polygons.

Chiang et al., (2013) sought to automatically extract the road network from a USGS historical topographic map. They used the Color Image Segmentation (CIS) technique, which belongs to the shape recognition family, where the technique separates homogeneous color areas of the map into groups. Godfrey & Eveleth (2015) used Image enhancement techniques in a GIS environment to make regions of similar color more consistent, and then performed an unsupervised classification to group (pixels) at a similar value based on their color. Kim et al., (2014) extracted parcels from old cadastral maps by segmentation. First, they removed the grid reference lines to dilute the image, then the labels considering the morphological and geometric features of the diluted image, and then rebuilt the boundaries of the land while the terrestrial areas of interest of a user were shaped by their polygonal approaches. Finally, Iosifescu et al., (2016) extracted information from various features of the Topographic Map of the Canton of Zurich. Summarizing their proposed methodology: scanning and geo-referencing the map, use of morphology of shapes to identify lines and polygons, vector creation and elimination of outliers and spikes.

Data

In our project, we used an already scanned map provided by The Cartographic Heritage Archives in order to apply our proposed methodology for the extraction of building footprints from a historical map. It is a town plan of Thessaloniki in scale 1:10.000, reproduced and reprinted by Great Britain Army Royal Engineers Fd. Survey Coy., 512 in 1944, created with revision from air photographs dated 1943. This is a multicolor map focusing on government municipal and industrial establishments of Thessaloniki, represented as single color filled polygons, while building blocks are represented as pattern-filled polygons. Each building is tagged with a number corresponding to its description in the map legend.

Methodology

Overview

Automatic vectorisation of raster images is not a simple task, considering the quality of the image and the way features such as buildings are depicted on it. In our case the raster image is already scanned in 300 dots per inch, satisfying enough to proceed with our proposed methodology. Our methodology consists of four fundamental steps (Figure2). Each step is carefully chosen in order to deliver the desired results. Constantly validating the intermediate results from each step and fine-tuning the parameters of each toolbox implemented, is fundamental in order to create a final model able to extract vector data from a raster image. In QGIS model builder, a variety of tools from GRASS and SAGA can be imported in the same model, allowing us to run all of the processing stages as one single process. These stages are:

1. Georeference of the scanned map; 2. Raster image processing; 3. Conversion from raster to vector format; 4. Vector data processing

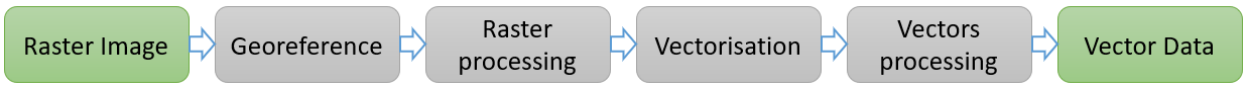


Figure 2. Methodology overview

Georeference

Integrating historical maps in GIS to analyze the spatial information they contain, or to incorporate them with other spatial data, requires georeferencing. That is, selected control points on the scanned image must be aligned with their actual geographical location, either by assigning geographical coordinates to each point, or by linking each point to its equivalent on a digital map. Once the control points are in place, mathematical algorithms are applied to warp the original raster image to fit the chosen map projection as nearly as possible. Further adjustments can be done manually to try to find the best fit for all parts of the original map.

In our case a total of 11 control points were selected distributed equally throughout the scanned image. Corresponding coordinates were selected manually using a basemap developed from Hellenic Cadaster with a pixel size of approximately 50 cm, and Greek Geodetic Reference System 1987 (GGRS87 / Greek Grid) as our reference system. Map units in this system are measured in meters. Choosing carefully places that exist both in the basemap and the historical map and moving control points to their corresponding coordinates resulting in mean error 0,894282 map units or 1,05941 pixels, in order to apply second-order polynomial transformation and nearest neighbor sampling method in QGIS Georeferencer toolbox. The output from this stage is a georeferenced tagged image file format (TIFF) raster image with no compression.

Table 1. Control points coordinates and residuals

ID	Enabled	Pixel X	Pixel Y	Map X	Map Y	Res X (m)	Res Y (m)	Res Total (m)
0	yes	5728	-3032	410905.088	4497529.053	0.16047	-0.0616369	0.171901
1	yes	10312	-6466	409839.129	4492794.061	0.151569	0.497035	0.519631
2	yes	4541	-1192	411916.473	4499101.103	0.331448	0.324183	0.463629
3	yes	4207	-774	412123.748	4499508.069	0.200376	-0.0958679	0.222128
4	yes	4841	-1555	411739.698	4498741.246	-0.0259153	-0.10182	0.105066
5	yes	3788	-2558	410598.486	4499176.636	-0.399351	-0.900828	0.985379
6	yes	7151	-2020	412182.433	4496815.487	-0.559665	0.345404	0.657669
7	yes	7184	-2564	411772.514	4496593.600	0.0757101	-0.380722	0.388177
8	yes	3161	-3366	409755.892	4499357.962	-0.864928	0.429498	0.965696
9	yes	2170	-2417	410144.813	4500455.658	0.514763	1.17594	1.28367
10	yes	4533	-3969	409764.555	4498097.663	0.415524	-1.23118	1.29941

Raster image processing

The next step of the image processing in order to apply our vectorisation method is the reclassifi-cation of the raster image and the conversion to binary image so as to extract building footprints from the original map. This is a significant first step towards extracting valuable information from our historic map as it separates the pixels containing information from the background of the map.

Reclassification of the raster image is performed applying rules, setting up a threshold initially in each band (red, green, blue) targeting to classify the image into two classes, one containing the buildings and one for every other value. Inquiring some pixels within color (black) filled polygons, in our map depicting buildings, we can identify that the maximum value a pixel can have is 80 out of the maximum of 255 in RGB scale. Since this value is the same at every band in black color there is no need to process the image separating each band or using any other color image segmentation technique. This is why we use the r.reclass toolbox from GRASS GIS, setting the following reclass rules: 0-80=1 , 81-255=0.

The reclassified raster image can then be negated using r.null toolbox from GRASS GIS in order to convert the class containing pixels with value equal to zero, as no-data. This step is greatly decreasing the size of the image, while ensuring our further applied processing will be applied in parts of the image containing pixels with information and not background values.

The binary image resulted from the previous steps needs some cleaning before it is ready for the vectorisation process. Although many filters can be applied, "remove small pixel clumps to no-data" toolbox from GRASS GIS seems to deliver the best results in our case. Removing groups of scattered pixels while not erasing a single pixel representing a building or generally part of a polygon. Thresholds here could be set from 50 to 350 regarding how many pixels represent a building. All groups of pixels with a total count of pixels below this threshold are set to no-data, removing many unwanted labels, letters in the map or even pixels scattered outside polygons. In this case we set the threshold at 200 considering optimal results without erasing a single building throughout the map.

Vector data processing

In this step a simplification procedure was performed in order to create more smooth lines for the features. The most commonly used algorithm in cartography is the Douglas-Peucker (DP) (Shi & Cheung, 2006). The algorithm splits the line data recursively and controls the compression quality by the threshold, and it is widely used in simplifying the trajectory of

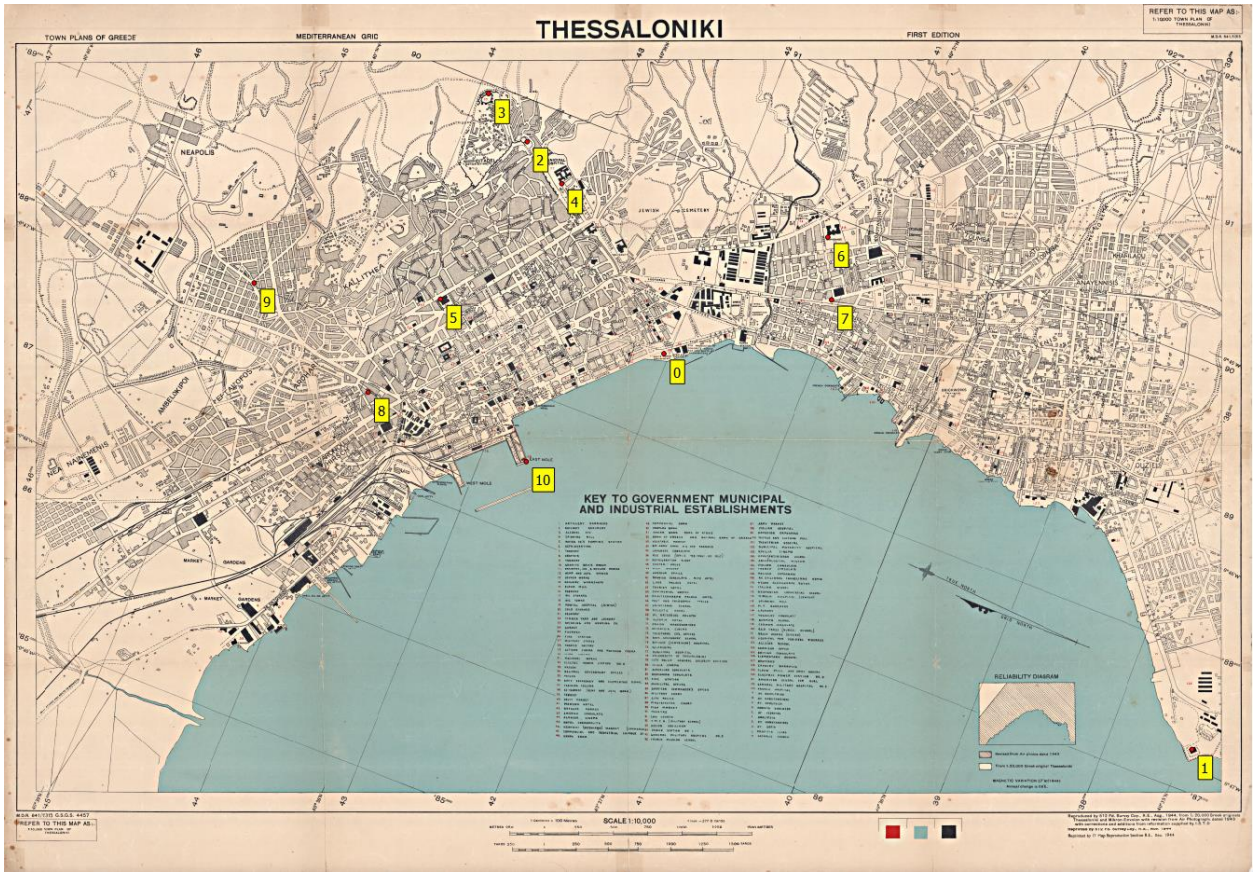


Figure 3. Control points used in Georeference

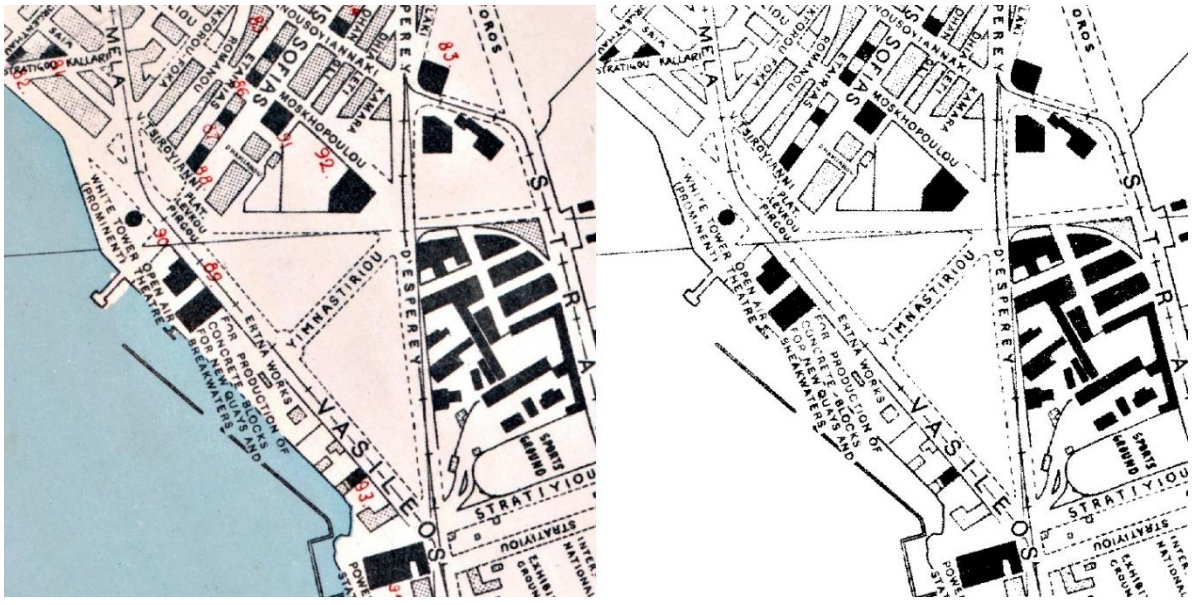


Figure 4. Left – scanned RGB image, Right – binary image

moving point objects due to its speed and accuracy. The DP algorithm constitutes of the following steps:

1. The first and the last point of the polyline are considered fixed and are linked by a straight line. The distances between the remaining points and the straight line is computed.
2. A tolerance distance is defined by the user and the distance of the furthest point is examined; if this distance is greater than a predefined tolerance, the point is added as a new vertex of the output polyline.
3. The iteration is now complete and the algorithm starts from the step 1, except that the new points created from the step 2 are now considered fixed. If no other points exceed the tolerance distance the algorithm concludes.

The last step was the filling of the holes (i.e. building block) less than a predefined size.

Figure 5. Left – binary image, Center – clumps marked as red, Right – binary cleaned image

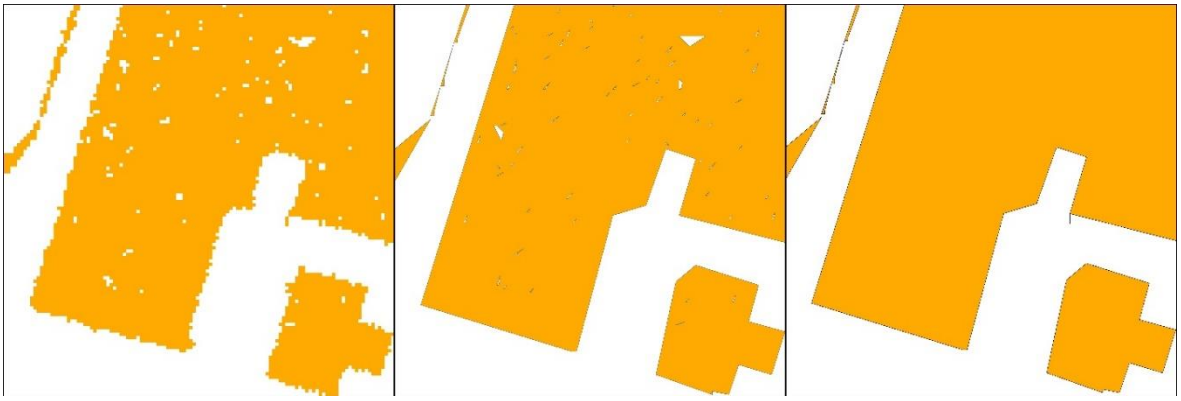


Figure 6. Left – Vectors extracted from cleaned binary image, Center – Simplified polygon lines, Right – Final results after the filling of holes

Model in QGIS

Final step of our proposed methodology is to implement all the processing described above into one model in QGIS. After running each step separately in order to fine-tune parameters and evaluate intermediate results, now we can use graphical modeler in QGIS to incorporate a series of tools to further automatize the vectorisation process meeting our needs for building footprints extraction from a scanned historic map.

In this case we insert all the tools from QGIS, GRASS GIS and SAGA GIS we used in the previous steps with the parameters described in each step. The only difference is the input of one more r.reclass and r.null tools in order to convert the filtered raster after the removal from clumps to no-data into a binary image.

Testing our model in a personal computer with eight-core processor, 16 gigabytes of RAM, and SSD disk, the results are more than promising. The georeferenced raster image of 700 megabytes, used as input, after a processing time of 2 minutes and 48 seconds, extracted vector data depicting building footprints quite accurately.

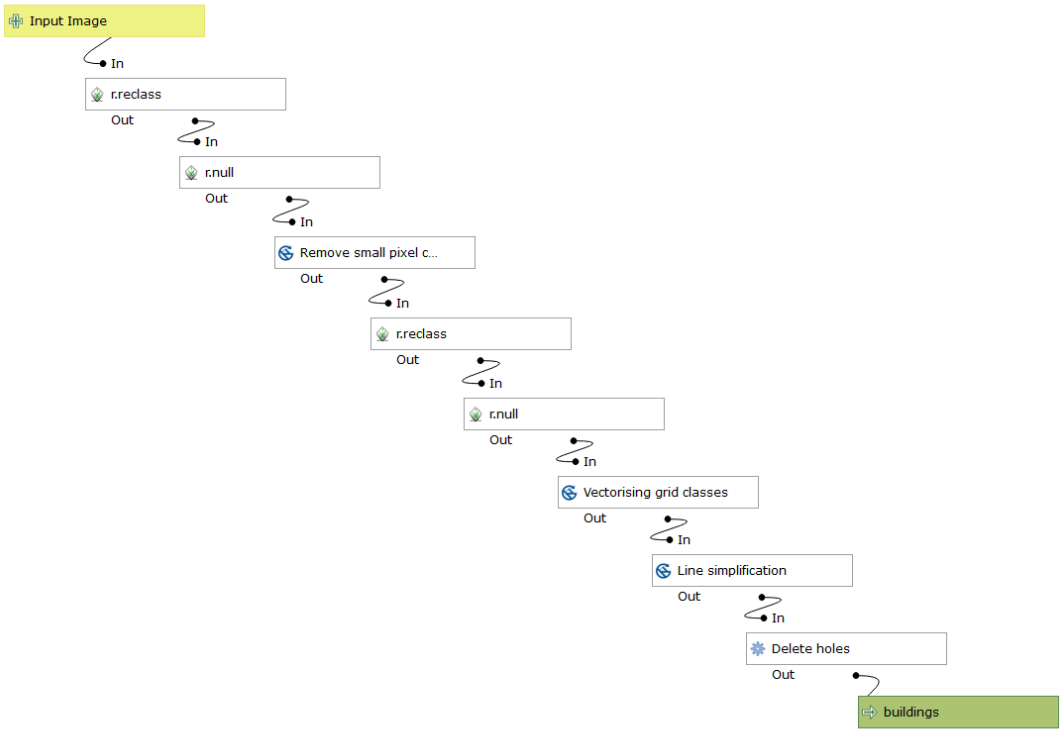


Figure 7. Model in QGIS Graphical Modeler

Figure 8. Left – Scanned image, Center – Clumps removed in red color, Right – Vectors extracted

Conclusions

In this paper we have presented an applied, semiautomated, practical workflow for generating polygon vector features from raster images implementing tools and algorithms from GRASS GIS and SAGA GIS into graphical modeler in QGIS. This general workflow can be used by researchers in generating vector data sets from thematic map raster images in hopes of unlocking some of the information contained in historical maps. Users need to modify parameters of the individual geoprocessing steps within the model to achieve the desirable result on a different map based on RGB values of the scanned image, but the general approach should prove valuable. QGIS is an open source GIS platform, thus allowing a wide variety of users to perform geoprocessing functions in several applications. Therefore, the threshold to modify this workflow, albeit adjusting the parameters to fit specific values, should be attainable for those interested in extracting valuable information from a scanned historical map within minimum amount of time. Based on the individual needs for automatic vectorisation and the available data, further tools could be added in the proposed model.

In some parts of the map, gridlines from the scanned image, text connected with polygons or strikethrough text presenting toponyms and other labels could not be removed with our methodology. In order to completely clear the image of any artifact not presenting a building polygon, manual selection of these polygons could help remove them. Text can not be automatically removed, so further processing is needed in order to totally remove any letter in the scanned image.

Acknowledgment

Authors would like to thank The Cartographic Heritage Archives, for providing an already scanned historical map ready to apply our proposed methodology

References

Ramer, U. (1972): An iterative procedure for the polygonal approximation of plane curves. Computer Graphics and Image Processing, 1(3), 244-256

Douglas, D., Peucker, T. (1973): Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. The Canadian Cartographer 10(2), 112-122

Yao-Yi, C., Leyk, S., and Knoblock, C.A., "A Survey of Digital Map Processing Techniques." *ACM Computing Surveys* 47, no. 1 (May 1, 2014): 1–44. https://doi.org/10.1145/2557423.

Kim, Nam Wook, Jeongjin Lee, Hyungmin Lee, and Jinwook Seo. “Accurate Segmentation of Land Regions in Historical Cadastral Maps.” *Journal of Visual Communication and Image Representation* 25, no. 5 (July 2014): 1262–74. https://doi.org/10.1016/j.jvcir.2014.01.001.

Godfrey, B., and Eveleth, H., “An Adaptable Approach for Generating Vector Features from Scanned Historical Thematic Maps Using Image Enhancement and Remote Sensing Techniques in a Geographic Information System.” *Journal of Map & Geography Libraries* 11, no. 1 (January 2, 2015): 18–36. https://doi.org/10.1080/15420353.2014.1001107.

Conrad, O., B. Bechtel, M. Bock, H. Dietrich, E. Fischer, L. Gerlitz, J. Wehberg, V. Wichmann, and J. Böhner. “System for Automated Geoscientific Analyses (SAGA) v. 2.1.4.” *Geoscientific Model Development* 8, no. 7 (July 7, 2015): 1991–2007. https://doi.org/10.5194/gmd-8-1991-2015.

Iosifescu, I., Tsorlini, A., and Hurni, L., “Towards a Comprehensive Methodology for Automatic Vectorization of Raster Historical Maps.” *EPerimetron* 11 (2016): 20.

Pallero, J.L.G. “Robust Line Simplification on the Plane.” *Computers & Geosciences* 61 (December 2013): 152–59. https://doi.org/10.1016/j.cageo.2013.08.011.

Shi, W., and Cheung, C.K., “Performance Evaluation of Line Simplification Algorithms for Vector Generalization.” *The Cartographic Journal* 43, no. 1 (March 2006): 27–44. https://doi.org/10.1179/000870406X93490.

Marguerite le Riche[1]

Identifying Building Footprints in Historic Map Data using OpenCV and PostGIS

Keywords: feature extraction, computer vision, historic maps, building footprints, geographical information science (GIS).

Summary: This paper describes a combined GIS and computer vision approach to the extraction of high quality vector building footprints from a series of scanned historic maps (Ordnance Survey County Series maps). The monochrome input maps predominantly use a speckled texture within a closed polygon to indicate the presence of a building. Image processing, segmentation and a manual data cleanup steps are performed, after which the raster data is vectorised and transformed into geo-referenced PostGIS geometries (via contours generated using the python OpenCV module). The identification of buildings, joining of sheets and post-processing is then undertaken using Structured Query Language (SQL) queries passed to a PostGIS enabled PostGreSQL database. To date, it has not been possible to fully automate the process, due to original cartographic decisions like printing text over buildings, damage to the original paper maps and technical limitations. However, the semi-automated process described here has yielded high quality building footprints for two map editions covering the study area.

Introduction

Background

The House Age Project (HAP) is a long term project which originated within the Registers of Scotland (RoS) and is currently running as a pilot in collaboration with Historic Environment Scotland (HES). Data for the study area, the historic Scottish County of Edinburghshire (Midlothian) [4] is provided by the National Library of Scotland (NLS) Map Library. The long-term ambition of the project is to assign an age (interval) estimate to each building which currently exists in Scotland (to the extent that it is possible to do so) by combining GIS data extracted from scanned historic maps with present-day GIS datasets.

These project aims encompass a number of challenging research problems, which are being investigated by the project participants using GIS, computer vision and entry-level machine learning approaches. One of the challenges is the extraction of building footprints from scanned historic maps to produce derived GIS vector datasets. The extracted building footprints will be tested as input for further analysis aimed at tracking individual buildings across datasets.

An assumption is made that tracking buildings across datasets will require shape-based comparisons. For this reason every effort has been made to develop a methodology which extract vectorised building shapes as completely and accurately as possible. Due to the large amount of available data, an automated method for extracting the data is highly desirable.

[1] GIS Team Leader, Registers of Scotland, Edinburgh [marguerite.leriche@ros.gov.uk]

Tools

The analysis is conducted using open source tools. The main python libraries used include OpenCV (computer vision), the python bindings for GDAL (Geospatial Data Abstraction Library), rasterio, scikit-image, numpy, re, pickle, glob and psycopg2 (PostgreSQL database adapter for python). The data is further manipulated in a PostgreSQL database with the PostGIS spatial extension enabled. Data is visualized using QGIS and a manual cleanup step is performed using the GIMP graphics editor.

Data

The source data are manually georeferenced (either by NLS or in a few instances, by the author), 3-channel, ~12000 x 16000 pixel scans of the Ordnance Survey's 25-inch-to-the-mile (1:2500) County Series map sheets [1,2], provided by the NLS in the GeoTIFF (Geographic Tagged Image File) format. Images are reproduced with permission of the National Library of Scotland under a Creative Commons Attribution-Non-Commercial-ShareAlike 4.0 International license.

There are four editions/revisions of the County Series Maps available, mapped between 1855 and 1949, covering urban areas across Scotland. The study area, comprises ~35 map sheets per edition, covering approximately 137 km^2. It includes a range of residential and commercial buildings in different configurations – from densely packed urban tenements to sparsely distributed, rural buildings and a range of suburban build patterns in between.

This paper describes processing of two editions of the 2nd revision, dated 1905-06 and 1912-13 respectively, wherein buildings are indicated using a speckled texture fill. Each map sheet has been assigned a unique identification number by the NLS. During processing, this reference number is retrieved from the filename and permanently associated with the data extracted from the map sheet. The NLS also provides index shapefiles containing polygons which correspond to the clipped boundaries of each map tile for each edition [3]. The attribute table of the index shapefile contains additional metadata such as survey and publication dates associated with the original maps.

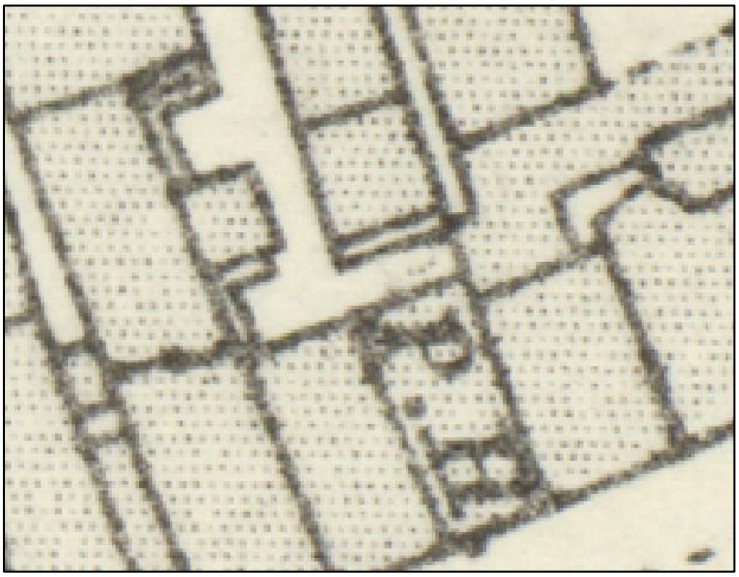


Figure 1. Buildings indicated with a speckled fill (OS County Series)

Methodology

Preprocessing

The GeoTIFF image metadata standard [6] expands the .TIFF image format to allow for geo-referencing information to be stored in the image header. This allows spatially enabled software (e.g. QGIS) to display the image correctly using a geographic or projected map coordinate system.

The OpenCV Library uses an image coordinate system and does not copy geo-referencing information from the header when writing a new .TIFF file. Therefore, as a first step, the geo-referencing information attached to each map sheet is retrieved using the rasterio transform function, and the resulting python object is saved for later re-use using the python pickle module.

As the input data is monochromatic, the initial approach falls into the category that can be described as edge-detection [9]. The input images have a bimodal distribution of pixel values, with the two peaks corresponding to the white background paper and printed black ink. Otsu's binarisation algorithm, which binarises the input image at an automatically calculated threshold (between the two modes) was found to yield the best result for this data.

The threshold value is calculated for each individual sheet, which accommodates variations in the amount of ink laid down on different map sheets as well as variations in scanning. The calculated value is shifted by a small amount before it is applied, so that light gray pixels are sliced to the same binary category as dark gray/black pixels. This is to retain as much information as possible about the original application of ink to paper, and to avoid creating small gaps in lines.

Printed pixels with the same value as the darker noise in the pixels which represent the background paper, are lost during the thresholding step. A number of options for image smoothing/noise removal were tested, but it was found that noise removal degrades the speckle texture.

The thresholded raster image contains a large number of unwanted map features, symbol and labels which are detected as contours by the OpenCV findContours() function. These include trees, boundary lines and a variety of other map symbols. Most problematic are symbols (e.g. archways) and letters (especially hollow letters like 'o', 'p', 'B' etc.) which are printed over, fused to the inside, or bisect the contours which represent buildings. This phenomenon is very well described in the literature [8, 9]. As any building identifiers are assigned post-extraction, it is not possible to automatically identify which polygons constitute the separate parts of bisected or fragmented buildings.

Some attempts have been made to remove the unwanted features in a targeted manner (e.g. K Nearest Neighbour algorithms, Haar patterns, approaching others with more computing power for assistance) but their removal remains resistant to automation.

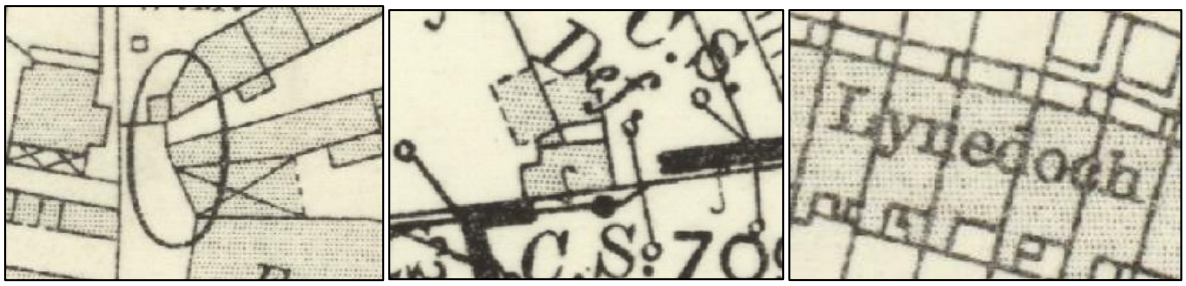


Figure 2. Examples of unwanted overprinted and fused features

For example, no attempt has been made to automatically reconstruct the 'walls' behind fused symbols after they have been removed. An acceptable compromise between speed and accuracy is to inspect contours for size and roughness (perimeter/area) and to mark those which are unlikely to be speckles or buildings for removal. Fused features are resolved in the manual step.

Due to paper warping, the map neat line is clipped in some places, leaving 'opened' buildings in the georeferenced raster images. To prevent buildings from being lost at the sheet edges, a new neat line is constructed by obtaining the rotated bounding box of all the data within the map sheet.

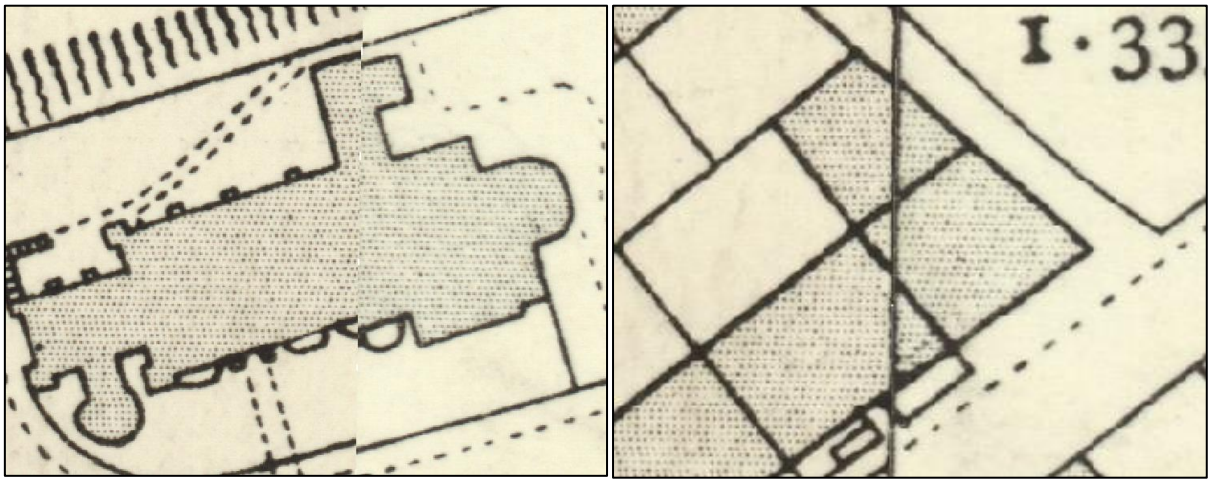


Figure 3. Lost (left) and retained (right) neatlines on sheet edges

The data contains a number of temporary glasshouses and canopied structures whose walls are indicated with dashed lines. Glasshouses are not extracted at this time, but buildings indicated with dashed lines are reconstructed and extracted. The first step in reconstructing dashed walls is to isolate dash-like contours in close proximity to speckles. The centroid of each of these contours is then connected to the nearest centroid within a threshold distance with a line. The shape of dashes in the dataset is not always well defined enough to reliably capture their orientation, making it difficult to draw closed shapes or extend the reconstructed lines to join up with neighbouring structures. Any unclosed shapes and unwanted artifacts created by the dashed line reconstruction are resolved in the manual step.

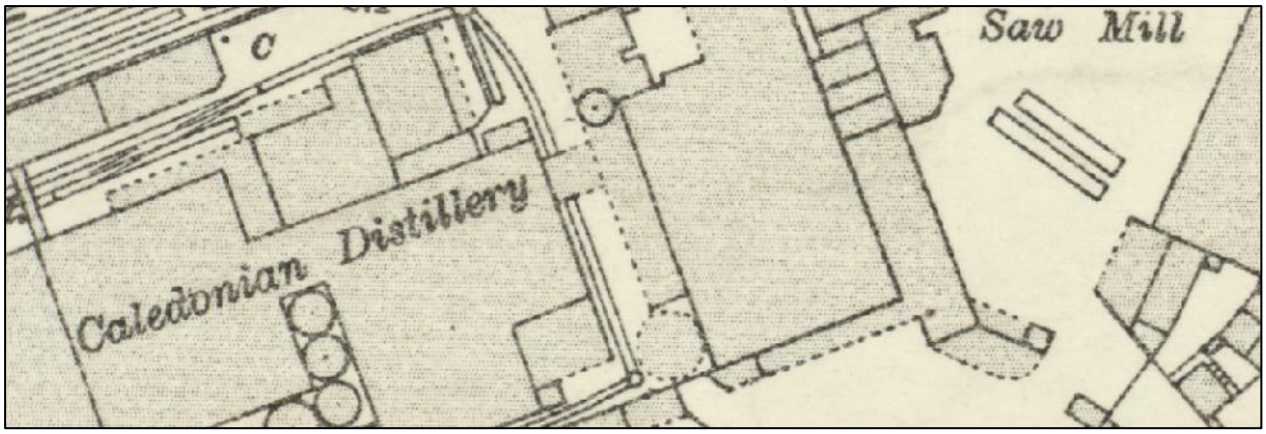


Figure 4. Structures indicated with dashed lines

The outputs from the steps above are then used to create a segmented image where speckles, potential buildings, potentially unwanted features and partially reconstructed dashed walls are differentiated by colour.

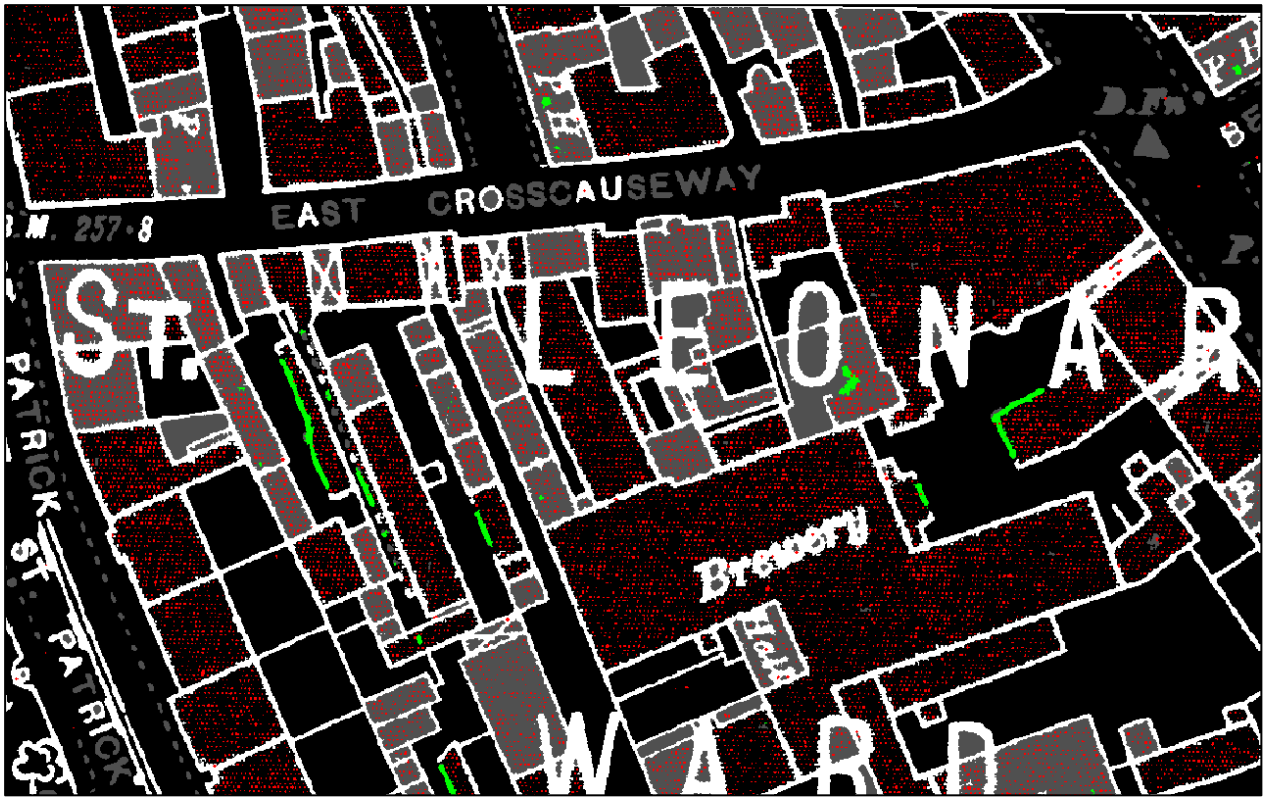


Figure 5. Segmented image with speckles (red), areas for removal (gray), probably walls (white) and reconstructed dashes (green)

Manual Step

Each coloured, segmented map sheet is inspected and edited manually using the GIMP software editor. This means that the input data is altered from this point onwards. However, the alterations can be made in different colours (e.g. yellow for changes which need to be retained and dark blue for areas which need to be discarded) to make the manual step transparent and auditable.

Features that have been classified as unwanted (dark gray) are checked. Any misclassified features are filled in a light colour, using the bucket tool. This issue may affect very small freestanding buildings or freestanding buildings with gaps in the outline.

Large gaps in lines are also closed. The OpenCV distance transform and watershed algorithms are highly effective at closing small gaps. However, some of the narrowest buildings are narrower than some of the largest gaps in this dataset. It is not possible for the watershed algorithm to distinguish between these cases, which results in spurious lines being drawn across narrow buildings. The algorithm also causes a noticeable rounding of corners beyond that which is present in the input data. In this instance, as we wanted to capture the shape of buildings as faithfully as possible, it was decided to close gaps in the manual step.

Hollow fused text, and features which completely bisect buildings are scored through to break up the closed shapes. Some non-hollow text and the scored-through remnants can be left as these and other artifacts it will be reduced to GIS geometry errors once the data has been sent to the PostGIS database, where it can be automatically repaired. However, they will in some cases leave a trace in the form of a small dimple in the final building outline.

Some buildings are so small that their speckles are all fused to the lines and the clone stamp can be used to resolve this.

The bright green reconstructed dashed lines are checked for open segments, which are closed with the crayon tool. Some areas with heavily printed, joined-up speckles may contain artifacts created by the dash reconstruction and these are removed.

There are rare locations where so many features are overprinted that it is not possible to make a (subjective) decision about where the building outline is. These are left as-is, for extraction to fail.

The time taken per sheet varies from 10 minutes for a more rural sheet to 2 hours for a dense city center sheet. This requires a reasonable investment of time and sustained concentration and may not be feasible for a large number of maps of dense urban areas. However, it only needs to be done once per dataset and in this instance did not take as long as developing a training dataset for testing machine learning approaches.

Extraction and Vectorisation

The manually corrected image is segmented again using colour thresholds. This yields two new binary arrays, one consisting of the speckles and the other of the larger line features which need to be retained. Contours which are not potential buildings or speckles are discarded.

The OpenCV contours for the speckle array is inspected to capture the first node of each contour. The contour node is transformed using the pickled affine transformation and wrapped in a text string which resembles the Well-Known-text (WKT) spatial data format. The WKT string is then passed to the PostGIS database using the psycopg2 python module, within a command to generate PostGIS geometry from text (ST_PointFromText()). There can be many speckles in a map tile (hundreds of thousands) and as the work is being conducted on a standard developer laptop, disk speed was a factor in the processing time.

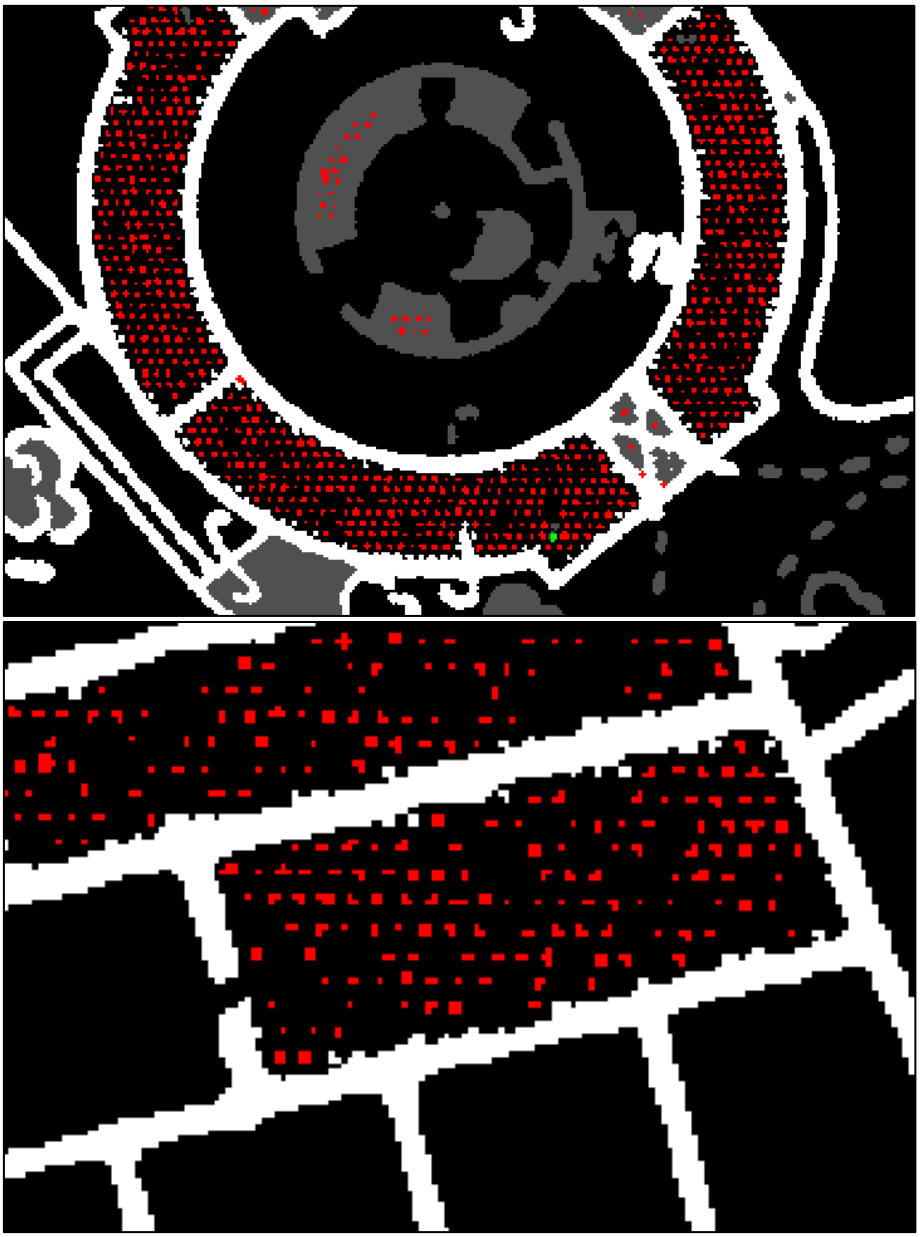


Figure 6. Manual corrections, building incorrectly classified as unwanted (gray), overprinted archway (white 'X'), artifact from dash-reconstruction (green) and gap in line (right image)

The line features array undergoes further processing. The GDAL sieve algorithm is used to remove pixel 'burrs' with a single 8-connection to the lines. These burrs are caused by speckles that have been fused/partially printed over the lines. They may cause the final building edges to be slightly offset from the center of the line in the input image.

The array containing the lines is skeletonized using a straightforward application of the scikit-image skeletonisation algorithm. This yields a skeleton with a large number of unwanted spurs which are difficult to remove using a raster/array approach. However, once the contours are captured and sent to the PostGIS database, these spurs are reduced to GIS geometry defects which can be easily repaired using standard functions like ST_MakeValid().

The skeletonized contours are retrieved and a basic validity check is performed, to confirm that it has at least three nodes. They are then transformed, formatted as WKT and posted to the PostGIS database. From this points onwards the analysis is conducted using SQL and a GIS approach.

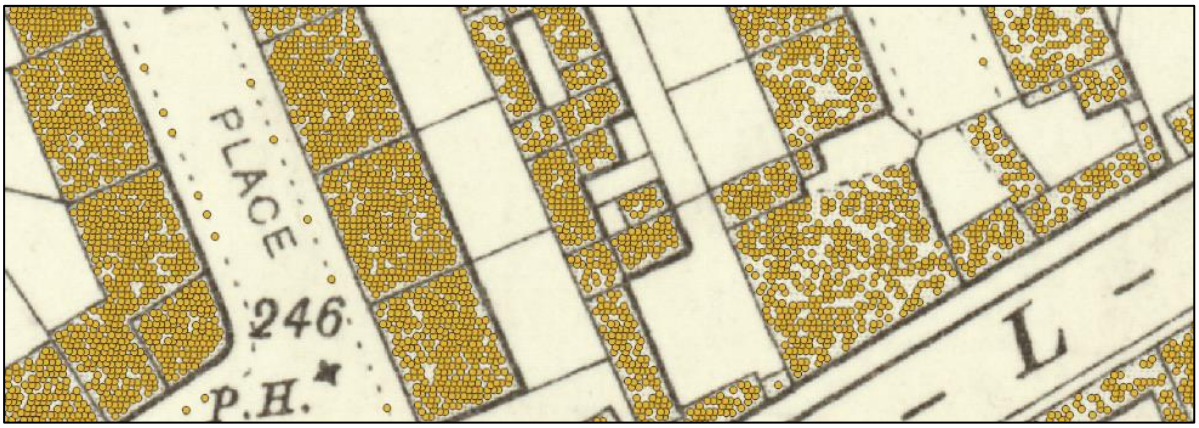


Figure 7. Speckles as vector point geometries after vectorisation

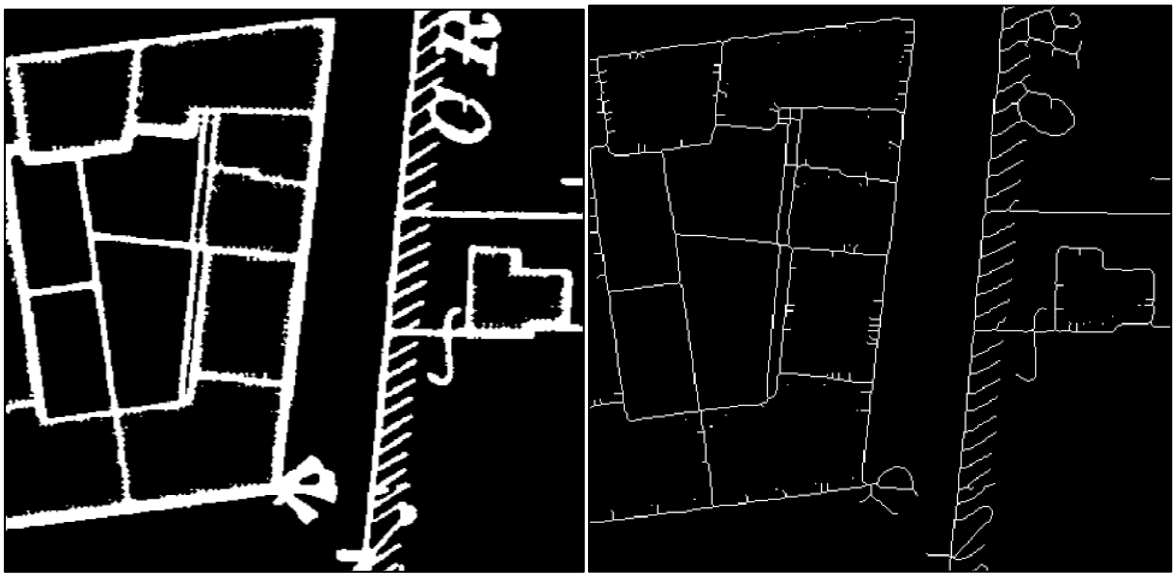


Figure 8. Lines before and after skeletonisation

Building Identification

At this point the PostGIS geometries contain invalid geometries, as defined by the Simple Feature Standard [7], which is not implemented in OpenCV contours. It is important to note that these geometry defects may, in some instances, represent useful information which will be lost once the geometry is repaired.

The newly created PostGIS geometries are reduced to their smallest constituents using ST_Dump() and then repaired using ST_Makevalid(). A tiny buffer followed by a tiny negative buffer resolves any remaining spurious geometries caused by skeletonisation and small unwanted features that were not removed in the manual step. New fields are added, the area of each polygon is calculated and those that are considered too large or too small to be buildings are discarded. A standard GIS point-in-polygon operation is performed to count the number of vectorised speckles inside each polygon.

The ratio of point to area is used to determine whether a given polygon is likely to be a building or not. An arbitrary threshold was chosen based on examination of the data and applied to all the map sheets, but equally it could be possible to calculate an optimal threshold for each sheet.

Figure 9. PostGIS vector geometry before and after repair and filtering

Once all the map sheets have been processed and copied to a master table, a primary key is added. At this stage the geometries are all simple, single part polygons consisting of a single ring. The table is inspected using QGIS. Some extremely large or oddly shaped buildings, e.g. buildings with a very large (speckle-free) central courtyard, may have been dropped. These can be retrieved from a back-up of the unfiltered set.

Post-processing

OpenCV creates two contours for each feature – one child inside the white edge and one parent outside. For freestanding buildings, this results in two overlapping polygons with very similar area and ratio properties. It is possible to remove one of them using SQL queries, and a choice was made to retain the inner one. However, making an automated choice between two slightly different shapes will have implications for any future shape-based analysis. Some cases may need to be resolved manually.

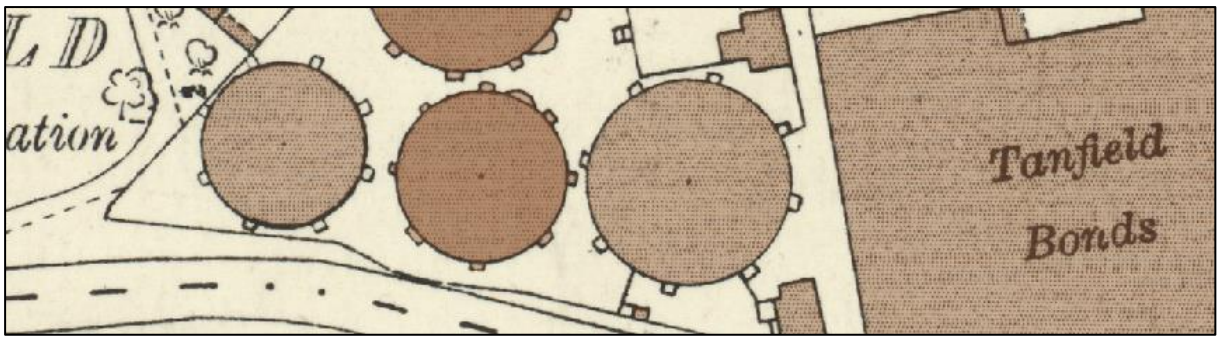


Figure 10. Overlapping polygons (darker) caused by parent/child contours

Sheet edges are joined by identifying all building polygons within a search distance of the neat line/index polygon edge. The geometry of buildings with different tile numbers are then snapped

to each other and their union calculated (with ST_Union()). A slightly more complex query is required to calculate the union for buildings which fall on the meet point of four corners. The results are varied and the resulting shapes may not be suitable for further shape analysis. In some locations, the union is excellent, but ST_Snap tends to yield unexpected results where the gap between the sheets are large or the buildings are not perfectly aligned. Where the gap is very large due to severe paper warping or imperfect geo-referencing, they may fail to join. In the dataset tested there were 37 good unions (although they did not necessarily yield a good shape), 58 correct unions with defective geometries, 6 unions were missed and 28 false unions were made. Further work will be required.

Figure 11. Joining sheet edges

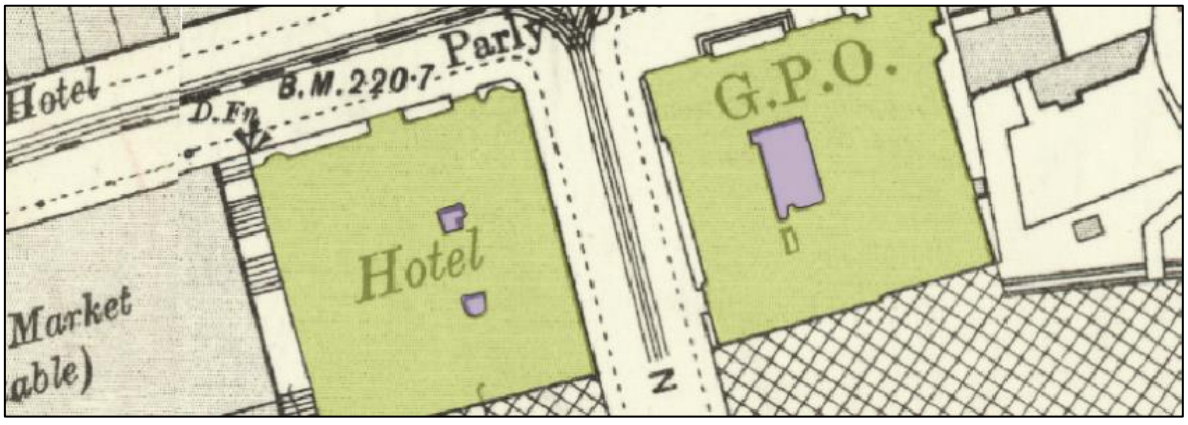


Figure 12. Adding interior rings for courtyards

The final step is to detect small, low ratio courtyard polygons using a polygon-in-polygon test. They are then cut out of building polygons as an inner ring. If hollow text is not removed before vectorization, they may cause spurious inner rings to be created in this step. Metadata is copied from the index layer to each building polygon extracted from the sheet.

Results

A confusion matrix was created using a sample of four (1st edition of the second revision) map tiles covering Edinburgh City Centre and Arthur's Seat. The results were inspected at a viewing scale of 1:750. 73087 OpenCV contours were sent to PostGIS as potential buildings. 18321 polygons were eventually classified as buildings and 54766 discarded as not buildings. There were 369 false negatives and 435 false positives. The majority of the false negatives are very small buildings which fell below the selected size threshold. The majority of false positives are similarly small polygons just over the size threshold, which form part of complex symbology like beach shingle or embankments where some 'speckles' may be present.

Table 1. Confusion matrix

<table>
<tr><th>Actual class (across) / Predicated class (down)</th><th>Building</th><th>Non-building</th><td rowspan="3">435 False positives:
32 sheet edge artifacts
386 tiny artifacts
17 yard/courtyard polygons merged with building polygons</td><td rowspan="3">369 False negatives:
21 dashed buildings lost
4 buildings fragmented
246 tiny buildings lost
59 buildings lost to gaps
2 slivers of buildings lost at sheet edges.
37 buildings lost as they were incorrectly misclassified as unwanted in preprocessing</td></tr>
<tr><td>Building</td><td>17886</td><td>435</td></tr>
<tr><td>Non-building</td><td>369</td><td>54397</td></tr>
<tr><td colspan="3">Accuracy = (17886 + 54397) / 73087 = 98.9%
Misclassification = (435 + 369) / 73087 = 1.1%
False positive rate = 435 / 18321 = 2.4%
False negative rate = 369 / 54397 = 0.7%</td></tr>
</table>

It is possible to extract the majority of buildings without a manual step, but buildings are lost in particular where overprinted features and gaps in the lines are present. An earlier version of the methodology (without the manual step, but including a watershed algorithm) identified 19066 features as buildings in the four sample map sheets, of which 1916 were false negatives. 425 dashed buildings were lost and 504 buildings were fragmented into two or more polygons.

Buildings affected by overprinted text are not randomly distributed – they tend to occur in more densely built up areas and a disproportionate number of public houses (pubs) are affected, as they tend to be small, oddly-shaped street corner buildings and have the text 'P.H.' overprinted in relatively large capital letters [5].

The manual step brings the overall extraction rate to a satisfactory 98.9%. However, the extraction rate is not the only factor to consider when judging the quality of the polygons and their suitability as input for shape analysis.

The boundaries of the extracted polygons are well-contained within the lines of the original scanned map image and generally falls visually on the centerline. The boundaries of adjacent buildings are coincident. Corners are slightly rounded and the geometries are highly complex, with a very large number of vertices, as the extracted lines follow the jagged pixel edges of the input raster data. Due to the complexity of the geometry, the output file size is large and subsequent processing slower than desired. Further work on simplification/generalization is being undertaken and it is not clear yet whether the outputs are suitable for further shape-based analysis.

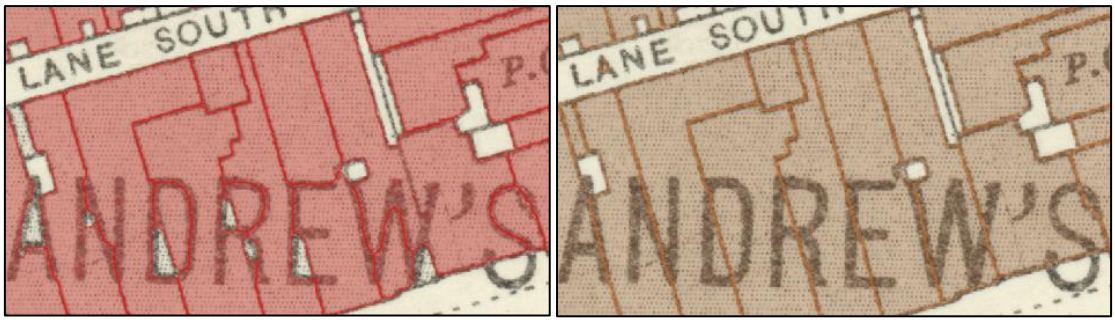


Figure 13: Effect of overprinted text without (left) and with (right) the manual step

Discussion

A number of practical factors are reflected in the methodological choices described above. The focus of the work is to produce datasets, rather than to solve all issues around automation. These unresolved issues include effects caused by some of the original cartographic choices, e.g. text printed across buildings and buildings indicated by dashed lines; as well as issues caused by the age of the original paper map sheets, e.g. damage and warping at sheet edges. For this reason, a semi-automated solution is being considered as part of the House Age Project. A manual step resolves the issues that are resistant to automation with a high degree of success. However, it does involve making (auditable) changes to the input data, which introduces a subjective element into the analysis.

The hardware available to the project has been limited to high-specification business laptops. The resulting lack of processing power has closed off a lot of potentially fruitful avenues of inquiry, from simple options like searching images with a sliding window to more advanced machine learning algorithms. However, it has been possible to implement a combined computer vision and GIS approach which solves the problem of building identification through simple spatial queries rather than processing intensive and complex feature detection methods.

The Well-Known-Text data format offers an easy and flexible way of transforming OpenCV contours to GIS geometries. It has an advantage over the shapefile format in that it allows OpenCV contours to be imported into a GIS system as-is, which may include mixed geometry types and invalid geometries (i.e. which do not confirm to the Simple Features Standard). Geometry defects are automatically removed during shapefile creation, so using WKT as an intermediary format allows for much greater control over the data during the transformation of OpenCV contours into valid GIS geometries.

PostgreSQL/PostGIS database tables and Structured Query Language (SQL) offers flexible methods for establishing spatial relationships between selected geometries in the same or adjoining map sheets. The geometries created using this approach are highly complex (they contain a lot of nodes) and this is already causing unexpected results to occur in steps relating to joining parts of buildings across sheet edges. Work on generalization is being undertaken.

Conclusion

To date, it has not been possible to develop a fully automated process for extracting building footprints from the OS County Series Maps used in this pilot study. However, a semi-automated process which makes use of a spatial approach to building identification has yielded high quality building footprints, with very little processing power and relatively easy-to-understand tools.

Acknowledgements

I would like to thank James Crone (EDINA), the Scottish Government's Data Science Accelerator Programme, Mike Middleton (Historic Environment Scotland) and Chris Fleet (National Library of Scotland) for their support.

References

National Library of Scotland (2020). Ordnance Survey Maps – 25 inch 1st edition, Scotland, 1855-1882. In digital form *https://maps.nls.uk/os/25inch/index.html*

National Library of Scotland (2020). Ordnance Survey Maps – 25 inch 2nd and later editions, Scotland, 1892 – 1949. In digital form *https://maps.nls.uk/os/25inch-2nd-and-later/index.html*

National Library of Scotland (2020). Ordnance Survey map – Survey and revision dates for county series mapping, 1843-1943. In digital form *https://maps.nls.uk/os/county-series/dates.html*

National Library of Scotland (2020). Counties of Scotland, 1580s-1940s.
In digital form *https://maps.nls.uk/counties/#edinburghshire*

National Library of Scotland (2020). Ordnance Survey Abbreviations.
In digital form *https://maps.nls.uk/os/abbrev/p.html*

Open Geospatial Consortium (2020). OGC GeoTIFF Standard.
In digital form *https://www.opengeospatial.org/standards/geotiff*

Open Geospatial Consortium (2020). Simple feature Access – Part 1: Common Architecture. In digital form: *https://www.opengeospatial.org/standards/sfa*

K. Tombre, S. Tabbone, L. Pelissier, B. Lamiroy, and P. Dosch. (2002). Text/graphics separation revisited. *Proceedings of the 5th International Workshop on Document Analysis Systems V (DAS'02)*, D. P. Lopresti, J. Hu, and R. S. Kashi, Eds., Springer-Verlag, London, UK, 200–211.

Yao-Yi Chiang, Stefan Leyk, and Craig A. Knoblock. (2014). A survey of digital map processing techniques. *ACM Computing Survey* 47, 1, Article 1.

Yao-Yi Chiang and Craig A. Knoblock. (2013). A general approach for extracting road vector data from raster maps. *International Journal on Document Analysis and Recognition (IJDAR)* 16, 55-81.

Chenjing Jiao[1], Magnus Heitzler[1], Lorenz Hurni[1]

Extracting Wetlands from Swiss Historical Maps with Convolutional Neural Networks

Keywords: Siegfried map, map feature extraction, Fully Convolutional Neural Networks, wetland reconstruction

Summary: Historical maps can serve as valuable resources for various kinds of researches such as ecology, land reclamation (Ngo et al, 2015), toponymy, history, etc. (Chiang, 2016). Specifically, extracting wetlands from historical maps facilitates researchers to investigate the spatio-temporal dynamics of the hydrological and ecological situation. Deep learning methods, especially Fully Convolutional Neural Networks (FCNN), provide an efficient and effective way to extract features from raster maps. To extract wetland areas from the Swiss Siegfried maps, we trained a U-Net based architecture and applied the learnt model to 573 map sheets across Switzerland. The pixel-wise prediction results were converted to polygons through a vectorization and generalization step. The vector wetland layers will be used for studies on land-cover change.

Introduction

The accessibility of the large numbers of historical maps indicates the high demand for successful feature extraction methods (Leyk & Boesch, 2009), because a wealth of cartographic information is still locked in the existing historical map series, such as building footprints, hydrography, map labels, etc. It is difficult to assess and compare wetland's historical and the current coverage and condition. Thus, historical maps are valuable sources for wetland change analysis as they retain the most reliable spatial data. Wetland areas in historical maps are often shown as composite elements that consist of clustered small symbols whose spatial distribution and patterns can define higher-level spatial objects, such as forests, grasslands, etc. (Leyk & Boesch, 2009). In *Figure 1*, the wetlands are denoted with composite horizontal strokes.

Conventional extraction approaches such as color image segmentation (CIS) have demonstrated their effectiveness in extracting map features, like roads, buildings, text, etc. Nevertheless, the conventional approaches are not perfectly suitable for historical maps, because the maps often suffer from poor graphical quality due to bleaching of the original paper maps and archiving practices (Chiang, et al., 2014). Based on the color feature of image pixels, CIS assumes that homogeneous colors in the image correspond to the same classes. The homogeneous regions that can be classified by CIS must show spatial contiguity. Thus, CIS may fail in extracting a composite object as a whole because of the trivial symbols and their complicated cluster. As we can see from *Figure 1*, one single wetland area is clustered by some trivial strokes, but the symbols are not connected with each other, although they form a contiguous texture.

[1] Institute of Cartography and Geoinformation, ETH Zürich [cjiao@ethz.ch]

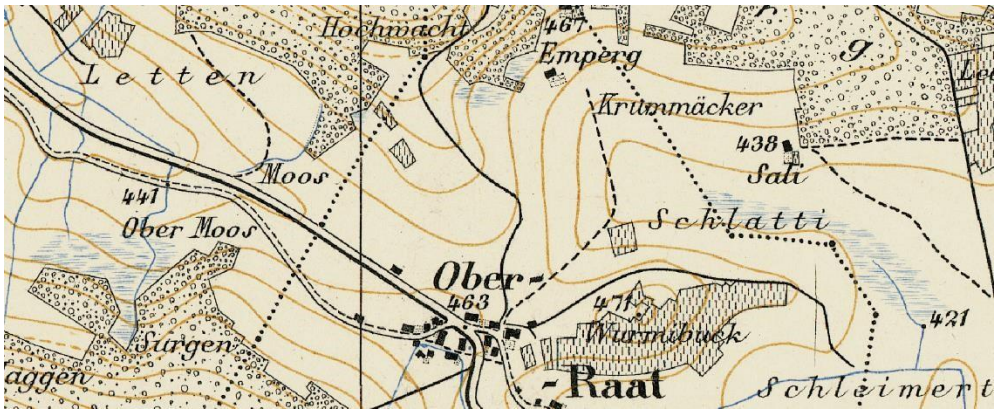


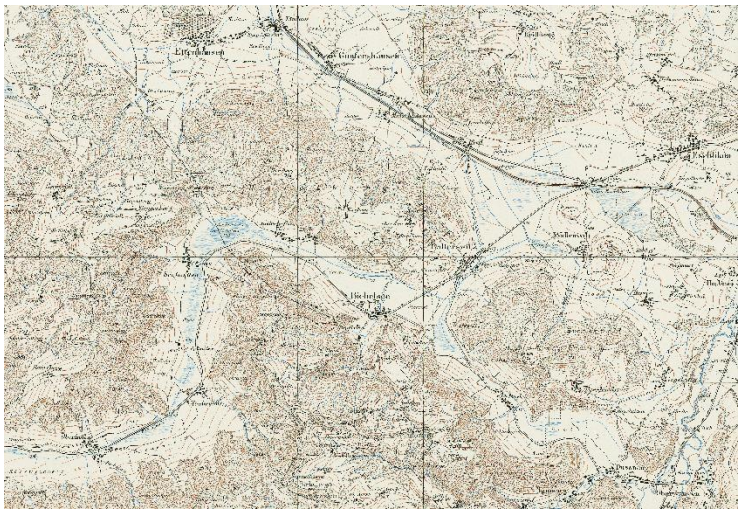
Figure 1. Wetland areas shown as composite graphic elements consisting of clustered horizontal strokes. Geodata © Swisstopo

In recent years, a group of supervised machine learning algorithms, known as Fully Convolutional Neural Networks (FCNN), has been exerting their superiority in automated feature extraction from images, especially historical maps (Saeedimoghaddam and Stepinski, 2019). An FCNN architecture, consisting of several convolutional, ReLU, and pooling layers, gradually extracts complex features from an input image, and then predicts the possibility of a feature correlating to a specific class. Yet, one of the challenges of FCNNs is that they inevitably require a large amount of training data to perform sufficiently well. However, producing training data manually might not be an option due to a lack of resources, which motivates the search for alternative sources of training data.

In this paper, we used the data from another study on wetlands in Switzerland as training data and trained an FCNN model to detect wetland pixels, which is followed by a vectorization and generalization step that generates wetland geometries.

Data and Methods

The Siegfried map is a comprehensive Swiss national map series that was published between 1872 and 1949. In a recent real-world use case, wetlands for whole Switzerland for the year 1880 are to be extracted for a Non-governmental Organization (NGO) from the Siegfried Map Series to carry out a study on land-cover change. For training an FCNN model, we need the input map sheet and the corresponding labeled wetland data. However, due to the unavailability of the labeled data for the year 1880, data of the year 1900 from the study on the historical development of wetlands in Switzerland has been used for training (Stuber & Bürgi, 2018). One Siegfried map sheet is shown in *Figure 2(a)*, with its corresponding labeled training data presented in *Figure 2(b)*, whose pixel holds a one if it belongs to a wetland, and zero otherwise. Naturally, these two datasets for 1880 and 1900 come from somewhat different distributions. The symbolization may have changed slightly between the two periods, and the map sheets of the training dataset have been georeferenced in a different way from the sheets of 1880.



(a)

(b)

Figure 2. Data example: (a) One map sheet; (b) the labeled wetland data corresponding to (a). Geodata © Swisstopo

We developed a segmentation model that is based on the U-Net architecture (Ronneberger & Brox, 2015), which was adapted to the problem at hand. According to Heitzler and Hurni (2020), the U-Net is comprised of two diametrical network paths, the contracting path and the upsampling path, which is presented in Figure 3. The contracting path consisting of convolution layers and max-pooling layers generates increasingly higher abstract representations of the input image. The max-pooling layers used in conjunction with the convolution layers allow for reducing the extent of the input matrix. The up-sampling path infers the class of a pixel based on its position and its surroundings by gradually combining these features on different granularities. Deconvolution layers are used in the up-sampling path to quadruple the extents of input layers. Each layer is followed by a corresponding elu activation layer except for the max-pooling layers and the final convolution layer, which has a sigmoid activation layer. These activation layers delimit the output probabilities to the range [0, 1]. In the used implementation, the convolution layers have 32 filter kernels.

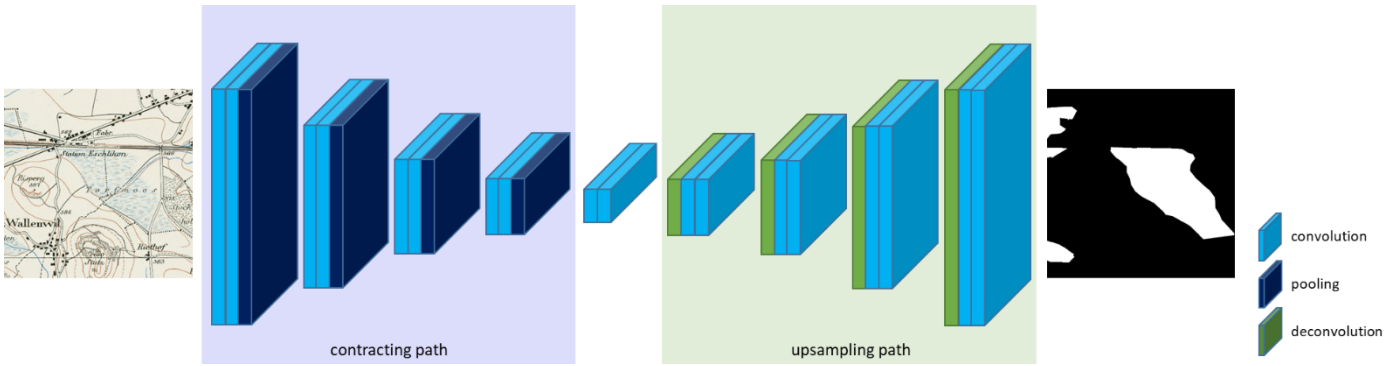


Figure 3. The FCNN architecture that is adapted based on the U-Net by Ronneberger et al. (2015)
Geodata © Swisstopo

Each map sheet sized 7,000×4,800 pixels is subdivided into a grid of tiles with the size of 200x200 pixels to be processable by the U-Net model. Furthermore, to avoid misclassifying the border pixels, the extent of the input tile is expanded by 60 pixels on each side. A well-learnt model is obtained after training with 36 map sheets, namely 36×35×24 tiles. When a map sheet is fed into the trained model, it generates a pixel wise prediction result, holding the probability of each pixel belonging to a wetland. The pixels with a possibility greater than 0.5 are segmented as wetland areas.

Once the wetland segmentation results are generated, vectorization is performed using functions from the Geospatial Data Abstraction Library (GDAL) to convert the raster binary results into vector geometries. Next, generalization is implemented using the Douglas–Peucker algorithm, which iteratively traces the line segments of the original polyline and only keeps the start point, end point, and the farthest point of the current line segment (Douglas & Peucker, 1973). If required, the vectorized and generalized geometries can still be manually corrected to yield high-quality wetland vector layers.

Result

An example for a wetland reconstruction result is shown in Figure 4, where Figure 4(a) presents the map sheet. Fig 4(b) illustrates the pixel-wise prediction result of a subsection of the area, in which the luminance of the red color corresponds to the possibility that the pixel belongs to a wetland. In other words, the brighter red pixels indicate a higher probability of them belonging to a wetland. The final result of this subsection is presented in Figure 4(c), which overlays the wetland geometries on the map sheet. We can see that the wetland boundaries are accurately extracted. Besides, a wetland area denoted with composite elements can be extracted as a whole, even though it is intersected with contour lines or other composite elements. The results verify the effectiveness as well as the robustness of the reconstruction approach. Wetland areas have been extracted and vectorized from 573 Siegfried map sheets across Switzerland with this method.

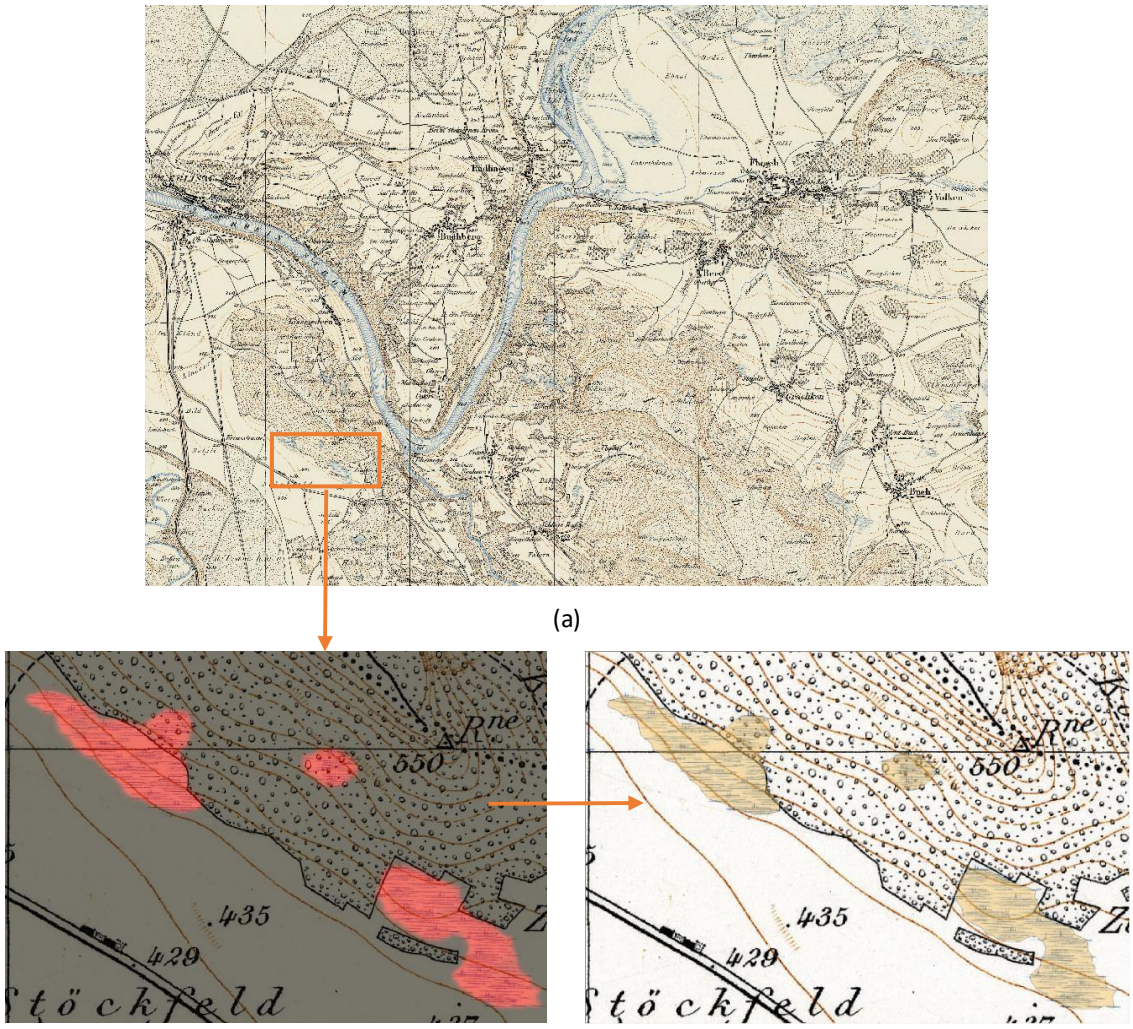


Figure 4. The wetland extraction result: (a) the map sheet; (b) the pixel-wise prediction result of a fractional area; (c) the vector geometry after vectorization and generalization. Geodata © Swisstopo

Conclusion

This paper addresses the wetland reconstruction issue by developing and training a U-Net based architecture, and applying the well-learnt model to the target map sheets. It is followed by a vectorization and generalization step, which outputs vector wetland layers that can be analyzed by Geographic Information Systems (GIS). The wetland reconstruction method performs well for recognizing and extracting the composite wetland elements. Future work will not only focus on the wetland extraction, but also on other hydrological features on the historical maps.

References

Chiang, Y., Leyk, S., Knoblock, C.A., 2014. A survey of digital map processing techniques. *ACM Computing Surveys, 47*, 1:1-1:44.

Chiang. Y., 2016. Unlocking textual content from historical maps - potentials and applications, trends, and outlooks. *International Conference on Recent Trends in Image Processing and Pattern Recognition*, 111-124, Bidar, India.

Douglas, D., Peucker, T., 1973. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *The Canadian Cartographer* 10: 112-122,

Heitzler, M., Hurni, L., 2020. Cartographic reconstruction of building footprints from historical maps: a study on the Swiss Siegfried map. *Transactions in GIS*.

Ngo, V., Swift, J., Chiang, Y., 2015. Visualizing land reclamation in Hong Kong: a web application. In: *International Cartographic Conference*, Rio de Janeiro, Brazil.

Ronneberger, O., Fischer, P., Brox, T., 2015. U-Net: convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computerassisted intervention*, 234-241, Munich, Germany.

Saeedimoghaddam, M., Stepinski, T.F., 2019. Automatic extraction of road intersection points from USGS historical map series using deep convolutional neural networks. *International Journal of Geographical Information Science*, 1-22.

Leyk, S., Boesch R., 2009. Extracting composite cartographic area features in low-quality maps, *Cartography and Geographic Information Science*, 36(1): 71-79.

Stuber, M., Bürgi, M., 2018. Vom "eroberten Land" zum Renaturierungsprojekt. Geschichte der Feuchtgebiete in der Schweiz seit 1700. Bristol-Schriftenreihe: Vol. 59. Bern: Haupt Verlag.

Mátyás Gede, Valentin Árvai, Gergely Vassányi, Zsófia Supka, Enikő Szabó,
Anna Bordács, Csaba Gergely Varga, Krisztina Irás[1]

Automatic vectorisation of old maps using QGIS – tools, possibilities and challenges

Keywords: map vectorisation, QGIS, Python

Summary: The authors experimented with the open source GIS software QGIS to reveal its possibilities in automatic vectorisation. The target object was a 1 : 200 000 map sheet of the third Austro-Hungarian military survey. Although QGIS provides tools for the main steps of the vectorisation process – such as colour segmentation of the raster image, skeletoning, vector extraction and vector post-processing – using them on an old map raises several challenges. This paper discusses the results of these experiments, and introduces a few simple python scripts that may help users in the vectorisation process.

Introduction

Historical maps are an important source of topographic, statistical and other types of information of past ages. Although it is possible to manually extract the required data from these maps, it is a tedious process especially in the case of sheets of old surveys that have very high information density. The goal is usually to transform specific content of these maps into vector-based feature data with attributes that can be further processed, analysed by GIS software. There are various attempts to fasten this process by automatizing certain steps or, if possible, the whole task.

Brown (2002) uses the commercial R2V software for vectorising geological maps. Arteaga (2013) developed the R tool Map Vectorizer, optimized for building outline extraction from cadastral/insurance maps. Iosifescu et al. (2016) experimented on 19th and early 20th century Swiss topographic maps also focusing mostly on building footprints but dealing with hydrography as well. They used various open source software: GDAL, ImageMagick and QGIS with GRASS. Peller (2018) provides a comprehensive review of several projects on this field, utilizing commercial or open source solutions.

Commercial GIS software ArcGIS provides the extension ArcScan to help vectorisation process. It is able to trace lines, recognise basic geometric shapes on raster images (ESRI, 2016). Open source QGIS also has modules for the task but these are not (yet) organized into a single user-friendly tool. Automatic vectorisation algorithms are often used on satellite imagery for extracting linear features (e.g. rivers) or landcover polygons (e.g. in Schwenk et al, 2018). All vectorisation methods follow the same chain of steps (although usually not implementing all of them):

- Scanning,
- Georeferencing,
- Image pre-processing (colour enhancement, noise removal, etc.),

[1] Department of Cartography and Geoinformatics, ELTE Eötvös Loránd University, Budapest [saman@map.elte.hu]

- Colour segmentation,
- Raster editing (line thinning, removing small objects etc.),
- Raster-vector conversion,
- Vector post-processing.

While works mentioned above mainly deal with large scale (mostly cadastral) maps, this paper summarizes the results of vectorisation experiments on late 19th and early 20th century topographic maps, mostly in QGIS environment.

Target object

The target object (*Figure 1*) of the present study is a 1 : 200 000 (Generalkarte) map sheet of the third Austro-Hungarian military survey depicting the confluence of river Olt into the Danube. The coverage of this map series stretches far beyond the actual territory of Austria-Hungary. This sheet was chosen because it is in a very good state comparing to other sheets physically available at our department and it depicts several various landscape types which makes it perfect for testing vectorisation methods.

Figure 1. "42°44° Slatina" sheet, the target object of present study.
(Courtesy of the Map Collection of ELTE Eötvös Loránd University)

Processing steps

Scanning

Although the first step, scanning the maps seems to be a quite straightforward thing, choice of resolution, colour depth, and image format may affect the success of every following step. If the resolution is too low, important details can be lost and nearby objects can be blurred. On the other hand, using too high resolution, the material errors and print errors of the paper (for example smudged ink) might also appear after vectorisation. The sample map sheet was scanned in 600 dpi, as a 24 bit colour TIFF image with lossless compression. This latter is important – using a JPEG compression on the image causes loss in the original pixel information.

Georeferencing

The Generalkarte sheets of the 3rd Military Survey were composed in the Lichtenstern polyhedric projection (Timár et al, 2009). As this projection is not supported by any known GIS software, a projection with similar properties is required. The Lichtenstern projection maps the selected 1° by 1° geographic quadrangle into a trapezium in which the central meridian as well as the bounding latitudes are true scale, so applying the Sinusoidal projection on it with proper datum and central meridian settings will result infinitesimal distortion as this projection also has true scale central meridian and parallels, and the bend of the edge meridians of the 1° by 1° quadrangle is negligible at this latitude. Although the georeferencing process also can be carried out in QGIS, the authors used Global Mapper for this task, using the corners and the edge midpoints as ground control points (GCPs).

Image preprocessing

Depending on the type of features to be extracted, there might be different image preprocessing solutions needed. The current map depicts water bodies and larger rivers with a horizontal blue hatch fill. This pattern usually needs to be blurred into a uniform colour before vectorising, which can be realized using a convolutional filter with cross-shaped kernel matrix. If the goal is the enhancement of these horizontal lines, another kernel with a horizontal line can be used (*Figure 2* and *3*). This preprocessing can be done for example in GIMP (to stay on the open source field).

0	0	0	0	0
0	0	0	0	0
1	1	1	1	1
0	0	0	0	0
0	0	0	0	0

1	0	0	0	1
0	1	0	1	0
0	0	1	0	0
0	1	0	1	0
1	0	0	0	1

Figure 2. Kernel matrices for enhancing (left) or blurring (right) horizontal hatching

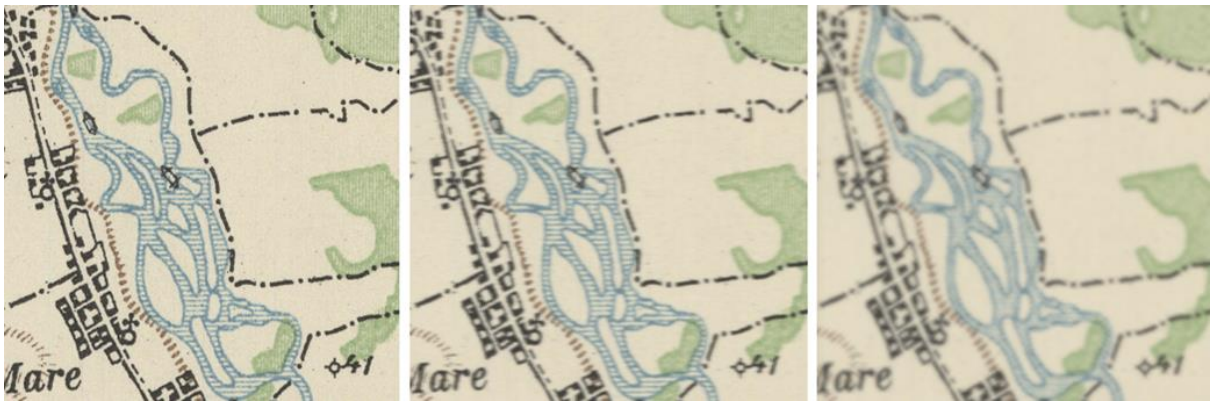


Figure 3. The original image with enhanced (left) and blurred (right) hatch

Colour segmentation

Before vectorising, we need to select those pixels of the raster that the features are shown by. This is called colour segmentation and depending on the method used may be resulted in a binary raster (where pixel values are set to 1 or 0 based on their colour belongs to that given feature type or not) or to a palette image in which there are a handful of possible colour values corresponding to the original printing colours of the map.

The *dzetsaka* plugin of QGIS does the latter. The user needs to create a few polygons on a vector layer, marking examples of the various printing colours (classes), and the plugin uses this information to classify pixels (Karasiak, 2016). This plugin proved to be successful for extracting black features (roads, buildings, labels). For rivers, however, another solution was used, taking advantage that blue was overprinted on other colours, therefore, by converting the image into HSV (hue/saturation/value) colour model and using hue filtering the blue features can be extracted even when they are printed upon other colours.

Raster editing

Due to the outworn paper of archive maps, the faded colours on it and eventual scanning errors, colour segmentation usually results in a raster that needs further processing before vectorisation: removal of small holes, standalone pixels, which is usually followed by skeletoning (thinning) lines. In QGIS, there are many ways for that. SAGA *Close One Cell Gaps* module fills every 1-pixel hole and makes the objects more homogeneous. With *Remove small pixel clumps* function the values of lone pixels, small pixel clumps can be changed to *"no data"* and after that these pixels are not in use. GDAL *Translate (convert format)* function can convert the pixels with 0 value (which means unwanted background colours) to *"no data"*.

To help colour segmentation and raster editing task, a simple Python application was developed. (see scripts created for the project: *https://github.com/samanbey/auto_vector*.) The program is based on OpenCV and the *rivgraph* module (Schwenk, 2020).

The interface of HSV filter program (*Figure 4*) consists of a set of sliders. The first six slider of interface can be used to define the limit of the HSV values of highlighted colour. There are more six sliders to eliminate the inaccuracy derived from errors of paper or scanning by setting properties of the erosion filter kernel, the gradient filter and the closing filters. The erosion filter eliminates remove small pixel clumps and makes lines thinner, the gradient filter can be used to extract outlines while the closing filter fills small gaps.

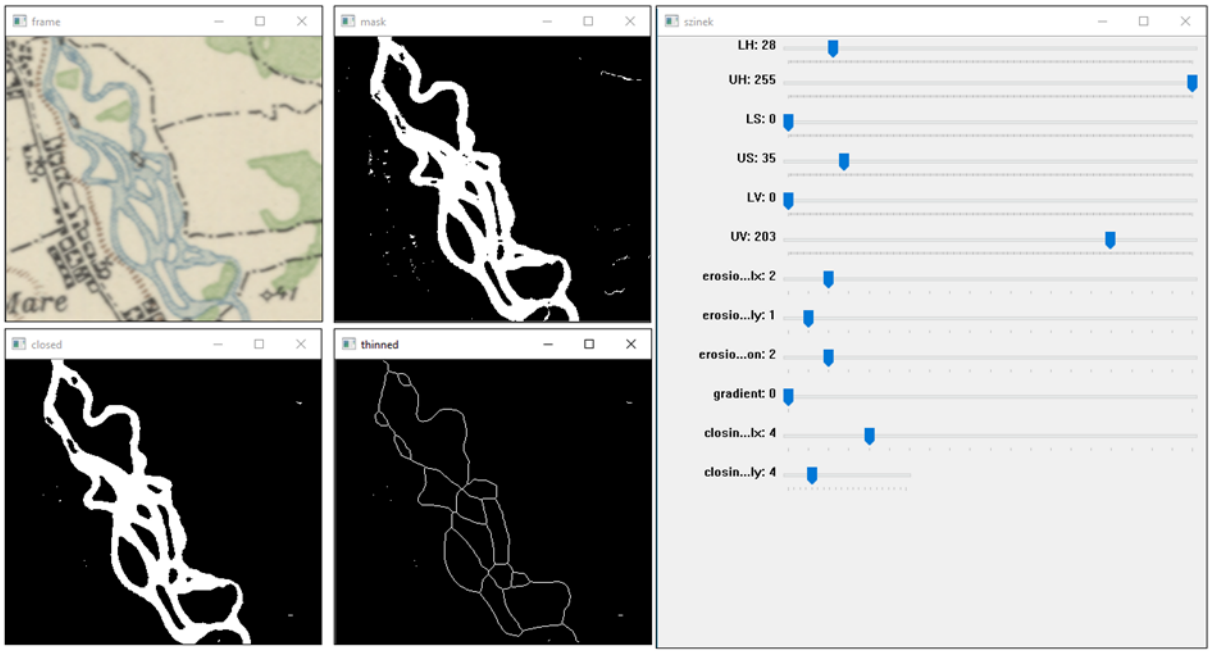


Figure 4. User interface and results of the colour filtering/skeletoning program

Another approach to thinning lines is using the "*r.thin"* function in the GRASS module of QGIS. Thinning is a crucial step before line vectorisation as the conversion only works with one-pixel wide lines.

Vectorising lines and vector post-processing

The thinned raster can be converted to vector using the *"r.to.vect"* function from GRASS module. The resulting set of polylines are still far from good: most lines are disjointed or branched into spiky short sections. The first step to eliminate these is running *"v.clean"* function with *rmdangle* command in GRASS module to remove most of the spikes. Some small branches still remain around the lines, so in SAGA module with "line properties" function is used to insert line segment lengths into the attribute table after which lines shorter than a certain limit are deleted from the table (The limit should be slightly larger than the actual pixel size of the raster). Finally, a geometric reduction is performed using the "Simplify" function in *Vector/Geometry Tools* menu.

Line vectorisation can also be carried out without thinning. In this case *"r.to.vect"* is used to create polygons from the raster image. Lines will be represented as polygon stripes. In the next step the holes inside the polygons are removed with *"delete holes"* function and a geometric reduction is performed with the "Simplify" function from *Vector/Geometry Tools* menu. Then the centerlines of polygons are created by the "Skeleton/Medial axis" function in *HCMGIS* plugin in Geometry Processing menu. To remove the remaining spikes and branches, the *"v.clean"* function is used.

The only remaining task is joining the still separated line segments. A simple python script can do the trick in QGIS. The algorithm iterates over possible polyline pairs and checks the distance of their endpoints. If it is below a given threshold and the orientation of the corresponding line segments is also similar, the lines are joined. The algorithm continues until no more line joins are performed. See script at: *https://github.com/samanbey/auto_vector*

Results

We got the best results so far when extracted rivers depicted as surfaces (with horizontal hatch fill) either as vector surfaces or generalized into polylines (*Figure 5*). Results with single-line hydrography elements (creeks, smaller rivers) are also promising, but auto-connecting fragments are not always perfect. Another problem with this category that colour filtering settings depend on how other map elements interfere with hydrography.

The toughest case is in mountainous areas where forest patches and relief hatching both may intersect water, not mentioning roads in narrow valleys running very close to creeks, and crossing them several times. *Figure 6* shows results in such a sample area. Although most of the hydrographic network is well extracted, the resulting polylines are fragmented. Automatic connection of these lines did not bring the expected results yet; depending on the parameter settings many gaps remained unconnected or the script also connected line ending that should not be connected. Road and railroad network is also troublesome, mainly because the same black colour is used for several different element groups: Buildings, fences, roads, tracks, railways and place names.

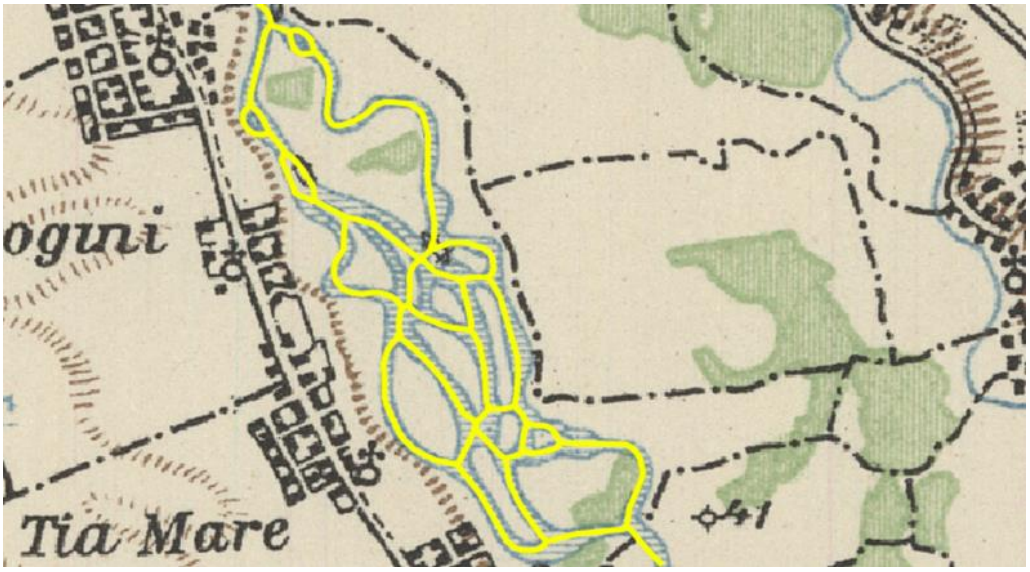


Figure 5. Rivers extracted as polylines

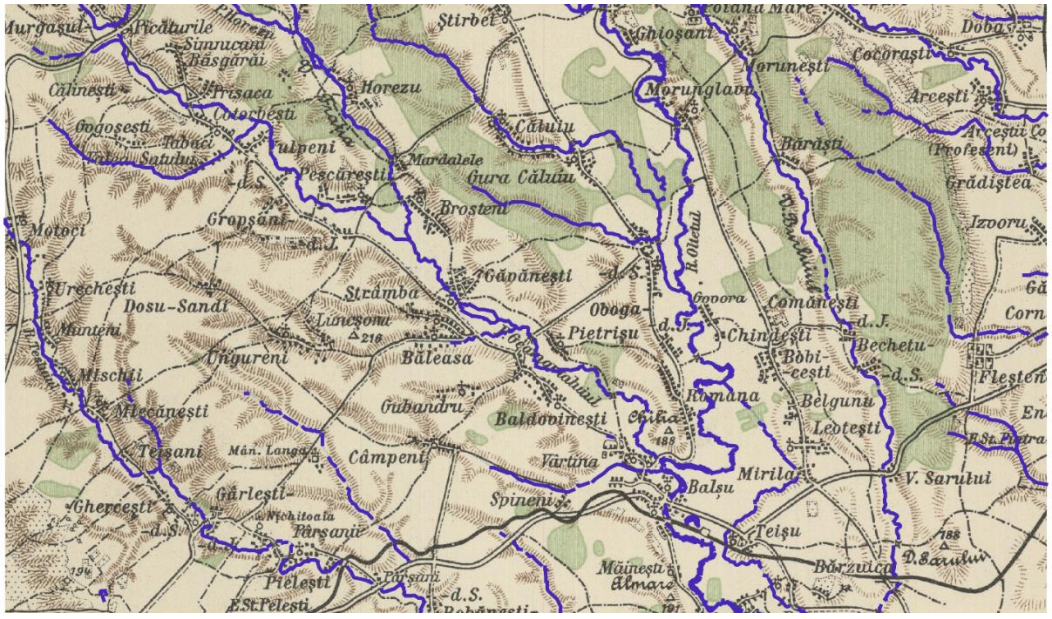


Figure 6. Recognition of single-line rivers in mountainous environment

Conclusion, further plans

Results show that it is possible to automatically extract vector features from scanned old maps using QGIS, but there is no uniform solution for all feature types. The steps of extraction depend on printing colours, the interference of various map elements and the content density of the map. The results are still far from prefect, and further improvements are needed.

Future plans include the improvement of vector post processing – especially auto-connecting fragmented lines, recognising linear elements with double or dashed lines. Another field of interest is recognising and extracting map symbols and labels – this may also help in distinguishing different element categories printed in the same colour, as the already recognised elements can be removed from the raster image before further processing.

Acknowledgement

EFOP-3.6.3-VEKOP-16-2017-00001: Talent Management in Autonomous Vehicle Control Technologies. The Project is supported by the Hungarian Government and co-financed by the European Social Fund.

References

Arteaga, M. (2013). Historical Map Polygon and Feature Extractor. Proceedings of the 1st ACM SIGSPATIAL International Workshop on MapInteraction. New York, NY: ACM, pp. 66–71. http://dx.doi.org/10.1145/2534931.2534932

Brown, K. (2002). Raster to Vector Conversion of Geologic Maps: Using R2V from Able Software Corporation. USGS Open-File Report 02-370. https://pubs.usgs.gov/of/2002/of02-370/brown.html

ESRI. (2016). Getting Started with ArcScan: ArcMap 10.4. http://desktop.arcgis.com/en/arcmap/10.4/extensions/arcscan/what-is-arcscan-.htm

Iosifescu, I., A. Tsorlini, L. Hurni (2016). Towards a comprehensive methodology for automatic vectorization of raster historical maps. e-Perimetron, Vol. 11, No.2, 2016 [57-76]

Karasiak, N. (2016) Dzetsaka: classification tool. https://github.com/nkarasiak/dzetsaka

Peller, P. (2018) From Paper Map to Geospatial Vector Layer: Demystifying the Process, IASSIST Quarterly 42 (3), pp. 1-22. https://doi.org/ 10.29173/iq914

Schwenk, J., A. Tejedor, A Piliouras, E Foufoula-Georgiou, J Rowland, (2018). Automatic Extraction of Channel Network Topology (AGU 2018)

Schwenk, J. (2020). RivGraph. https://github.com/jonschwenk/RivGraph

Timár, G., G. Molnár, V. Crăciunescu, (2009). Georeference of the 1:200,000 ’degree maps’ of Central Europe (about 1910). Geophysical Research Abstracts, Vol. 11, EGU2009-2574, 2009

Anna Piechl[1]

(Semi-)automatic vector extraction of administrative borders from historical raster maps

Keywords: image enhancing; Historical boundary data, Raster-to-Vector Conversion

Summary: Historical raster maps can be a useful source of information and allow us to travel backwards in time. The HistoGIS Project wants among other things also the creation of geodata from historical political/administrative units to allow queries with this data. Therefore, already existing vector data is used but also scanned historical raster maps from the 19th and early 20th century. When it comes to the creation of new vector data the use of historical maps is inevitable. As the creation and extraction of vector data from raster maps can be very challenging especially when it is done manually by heads-up digitizing. Since this can be a very time-consuming and tedious task, the aim was to find an easy semi-automatic solution for extracting and vectorizing administrative borders from historical maps using open source tools. Therefore, different software tools and processes were tested. The main data source of this work builds the "Historischer Atlas der österreichischen Alpenländer".

Introduction

Historical border changes can help to answer a variety on open research questions. Spatial references become more and more important, not least of the "Spatial Turn" which places space or geographical space as a cultural entity[2]. Therefore the HistoGIS project from the Austrian Center for Digital Humanities and Cultural Heritage which is part of the Austrian Academy of Sciences wants to contribute this development. HistoGIS should support research projects with its historical geodata as an analytical tool, enrichment service, publishing platform as well as a workbench and stable repository (oeaw.ac.at). The project involves not only scanning, georeferencing and the extraction of administrative boundaries, but also collecting already existing shapefiles and adapting them to the projects data schema. Still the focus of the project is to generate data sets from the Habsburg Monarchy as granular as possible between the timespan of approximately 1780 until 1918.

When it comes to the investigation of the already existing shape files, we realized how inaccurate some of the geometrical data was. And on the other hand, there are still a lot of political/ administrative entities missing.

At this point, where we need to compare the existing data with other maps on the one hand and where we need to generate new data from old historical maps on the other hand. To create new data from old maps is more complex and time intense. So the aim of this paper was to find and afterwards provide an easy solution for vectorizing historical administrative borders from old maps.

The methodology of the workflow that was used for creating and extracting vector boundaries from old maps, here preferably from maps of the "Historischer Atlas der österreich-ischen Alpenländer" is described below.

[1] Austrian Academy of Sciences, Vienna [anna.piechl@oeaw.ac.at]

[2] Döring, J.; Thielmann, T. (Hrsg.) (2008): Spatial Turn. Das Raumparadigma in den Kultur- und Sozialwissen-schaften. Transcript Verlag, Bielefeld.

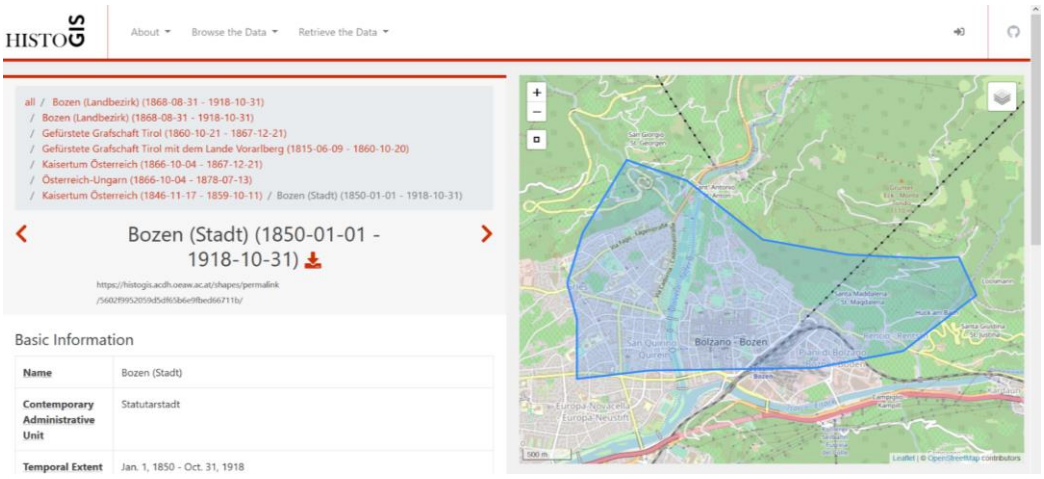


Figure 9. Example of a polygon with poor spatial accuracy (Screenshot HistoGIS App)

Preprocessing

After choosing an already scanned map for the region of interest it continues further to the next step. To be successful on (semi-) automated vectorization not only the step where it gets to the raster-to-vector conversion is important, but also to do some preprocessing, such as working on the image quality as well as enhancing and editing pixels. This is not only applicable for open source solutions but also for commercial ones[3]. Before on heads further to the GIS (Geographical Information System) part some image enhancement for the scanned map will take place. Here GIMP a pixel based open source image processing software was used. First it is important to emphasize the objects of interest of the map. Image mode at the beginning must be RGB (red, green, blue) which allows using Brightness and Contrast tool. In case of the map from the "Österreichischer Atlas der historischen Alpenländer". Brightness and Contrast were both increased for the value of 70.

After colour brightness and contrast are adjusted, I play with the threshold of the canals of the map and increase the green value to get clearer borders. Whereby the threshold changes the raster into a black and white image. Black pixels represent pixels with a value out of the threshold range, white ones show pixels in the threshold range[4].

As soon as the Image Enhancement is done the Fuzzy Select Tool was used to select the borders. After selecting, the selection gets copied in a new image and will be exported in TIFF-Format.

[3] https://www.iassistquarterly.com/index.php/iassist/article/view/914
[4] http://search-ebscohost-com.uaccess.univie.ac.at/login.aspx?direct=true&db=nlebk&AN=923882&site=ehost-live

Figure 11. Image before pre-processing

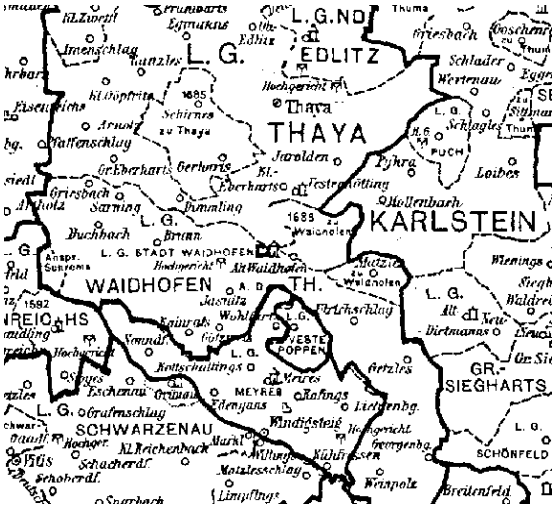


Figure 10. Image after pre-processing

Georeferencing the selected borders

To assign the raster map to geographic coordinates and a coordinate system it is necessary to georeference the raster image with the emphasized borders. For this step the Georeferencer Plugin in QGIS open source software was used. As a base map and therefore geospatial reference OpenStreetMap was used. Because a coordinate grid is not available in this map it is necessary to pick Ground Control Points (GCPs) manually. Beside the Ground Control Points, another very important factors are the Transformation Settings. There are three very common algorithms in use. A first-order polynomial transformation will scale, translate and rotate, a second-order polynomial transformation is able to handle some curvature, hence a third-order polynomial transformation allows even higher rates of distortion. We can find also projective option that offers rotation and translation of coordinates and a linear one on the other hand, which is used to create world files, but does not really transform the raster. Another option it to use the Thin Plate Spline algorithm which was determined for our map. As a resampling method Cubic Spline was defined. The reason for using these options was because they are useful for historical data because it is able to introduce local deformations in data and it is suggested for raster images with low quality[5].

Raster to Vector Conversion

A semi-automatic vectorisation approach can help to reduce working time for a project. Especially when it comes to the vectorization of a larger number of maps it is worth to think about a somehow automatized option. A semi-automatic way working with non-commercial tools makes sense when someone deals with maps that have the same or similar colours or style and come from one source it is a good opportunity to use a prepared automatized workflow.

Once the preselected borders are georeferenced we want to head to the vectorization part. Therefore, the first step is to use the GDAL Tool *Contour* from the Raster extraction toolbox. It is supposed to be used for creating Digital Elevation Models (DEM). So, this is kind of a remote sensing techniques come into play. Here the default settings were used, except the

5 https://gracilis.carleton.ca/CUOSGwiki/index.php/Georeferencing_Raster_Imagery_in_QGIS_using_Vector_Data# Adding_Control_Points

Interval number between the contour lines got changed. For the maps of the “Historischer Atlas der österreichischen Alpenländer” the value had to be increased to at least 150 (*Figure 4*) to get a usable result. Testing shew that values below 150 would produce to many line objects which would make further working with the data even more difficult.

As a result, we get a line vector layer which shows the borders of the raster selection. With the settings done we received to lines. See in *Figure 5*. To receive a single border line, it is needed to find the centreline of the two outline border lines. After testing and not getting any good result a buffer came into play. Therefore, the next step was to create a buffer around the vectorized borders. To reach a “closing” result the Buffer Distance had to be set up to at least 180 meters (*Figure 6*). The tests showed that any lower value would not be enough to reach the wished results.

As a buffer result (*Figure 7*) we receive a polygon vector layer. The resulting buffer objects must be merged then and saved as a Polygon shapefile to extract the center lines afterwards.

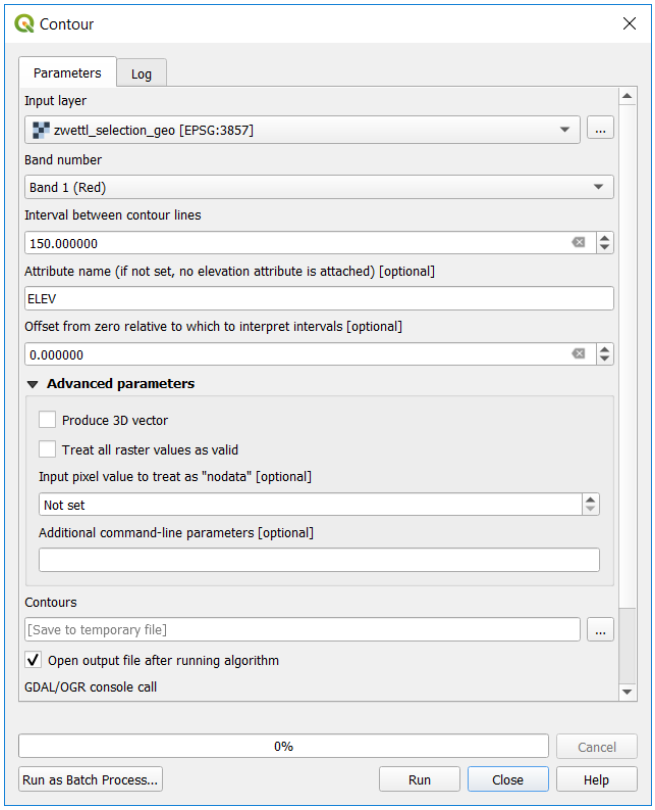


Figure 12. Screenshot Contour Tool Preferences

Figure 13. Results after running the Contour tool

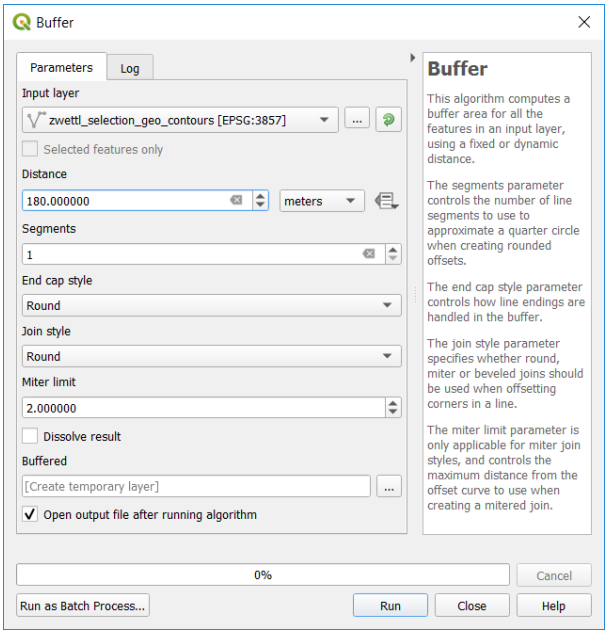


Figure 14. Buffer preferences

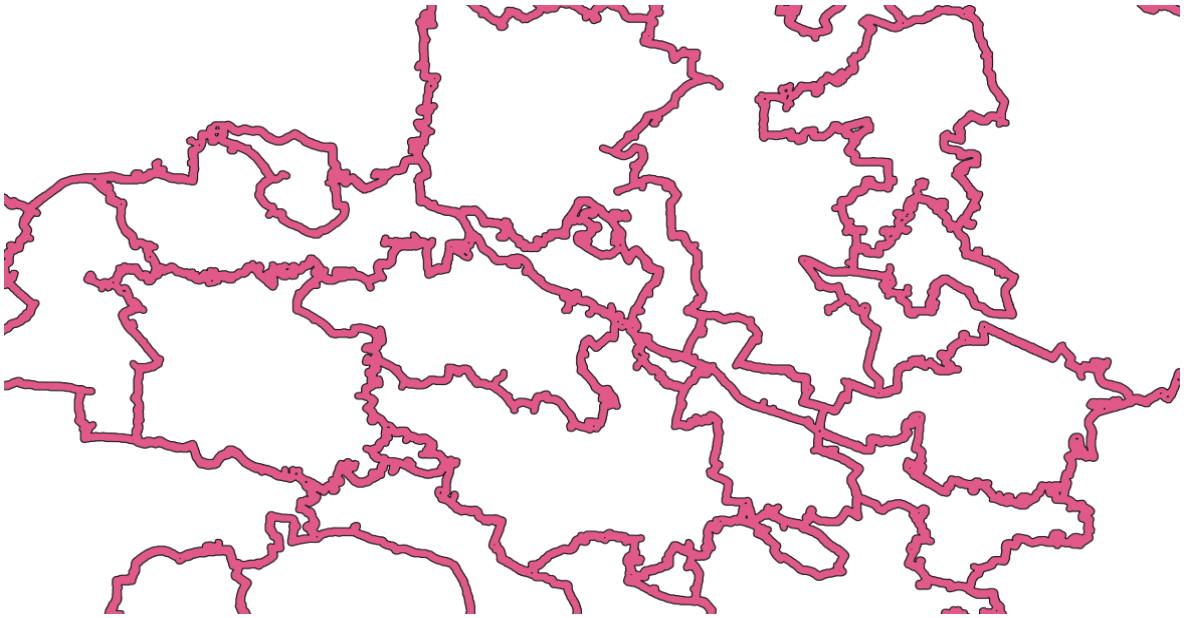


Figure 15. Results Buffer

Using Open Jump for Skeletonizing

After testing several tools and methods in QGIS that did not show satisfying resuls for a skeletonize or center line extraction process the solution was to try a different QGIS tool. The open source software OpenJUMP provides a Skeletonizer Plugin to be tested. The GIS-Software and tools are freely downloadable. This tool creates skeletons of polygon objects. To get the centerline of our buffer polygon the buffer shape file, that was created beforehand in QGIS must be loaded in the OpenJUMP Software. See preferences in *Figure 8*). As a result, we get a connected line shapefile (*Figure 9*). After the process we export the data as a shape file.

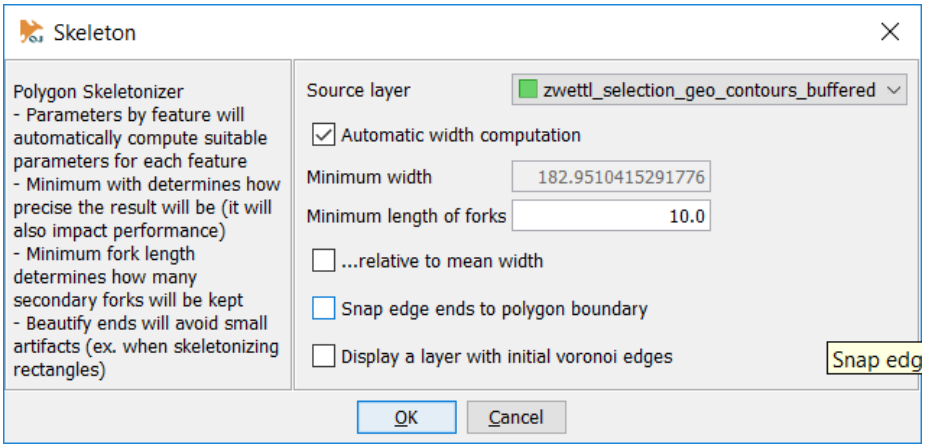


Figure 16. Skeletonizing buffered border selection - preferences

Figure 17. Result Skeletonizing

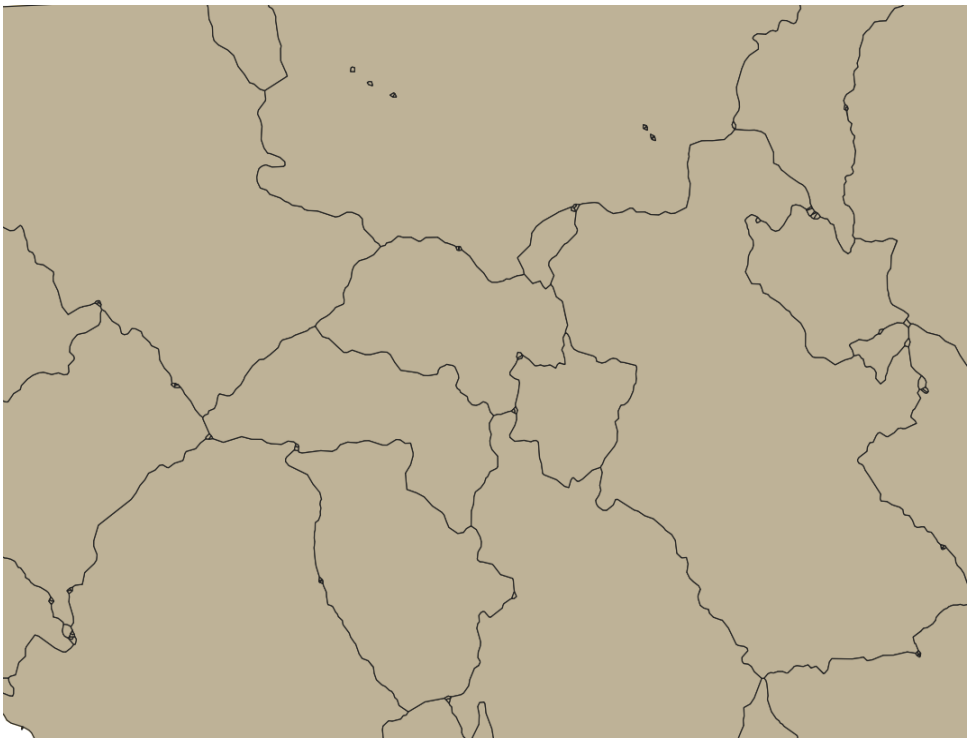


Figure 18. Polygonized Boundaries

Creating final Boundary Data set

To get the final boundary data set we load the skeletonized results back into QGIS. As we wish to have polygons for our political entites we transform our lines into polygons. To reach this we must use Polygonize tool. As shown in *Figure 11*, some of the boundaries do not close, therefore the end of the lines will be closed, by doing this the Polygonize tool will transform the line features into polygons, see in *Figure 12*. The result is okay but not perfect. Within the polygons there are still small areas that must probably be removed manually or with a cleaning tool. This leads back to the poor image segmentation.

Conclusion

The research in this paper presents one easy solution for extracting vector data from historical maps. Although this paper shows only one way in an easy manner for doing this process but still it lets us know that there are several options when using non-commercial software such as GIMP, QGIS or OpenJUMP. The tools show usable results, even if they are still in need of improvement. The testing shows that it is not possible to be successful without doing any pre-processing. And still it is no easy to find an open source solution that is easy to deal with for people who have low programming knowledge.

References

Döring, J.; Thielmann, T. (ed.) (2008): Spatial Turn. Das Raumparadigma in den Kultur- und Sozialwissenschaften. Bielefeld: Transcript Publishing.

Graser, A. (2014) *Learning QGIS - Second Edition*. Birmingham: Packt Publishing (Community Experience Distilled). Available at: http://search-ebscohost-com.uaccess.univie.ac.at/login.aspx?direct=true&db=nlebk&AN=923882&site=ehost-live (Accessed: 13 February 2020).

Peller, Peter. (2018). From Paper Map to Geospatial Vector Layer. IASSIST Quarterly. 42. 1-24. 10.29173/iq914.

Előd Biszak[1], Gábor Timár[2]

First vector components in the MAPIRE: 3D building modelling and Virtual Reality view

Summary: MAPIRE is an originally raster-based portal, presenting scanned and geo-referred historical topographic, cadastral and town maps, as overlays on modern cartographic datasets, such as OpenStreetMaps or HereMaps. Its first 3D option was adopted via the SRTM elevation dataset, still on raster base, showing 3D panoramas from the viewpoint, using the historical map layer. Here we introduce the first vector component of the MAPIRE, developed in Cesium software environment. To improve the city and town maps, we applied a new 3D building modelling utility (MapBox Vector Tiles). The details of the historical buildings – varying in different time layers – are reconstructed from archive plans as well as photographic heritage provided by the Arcanum (the host of MAPIRE) and historical postcards. Unrevealed building parts are defined according to the 'usual outline' of the given age. Using this technology, we are able to provide virtual 3D tours for the users of MAPIRE, first in the frame of the 'Budapest Time Machine' project. Besides, the Virtual Reality (VR) tour is available also in the service, still in raster basis: as a pilot project, we can follow pre-recorded paths on the historical map using VR glasses.

[1] Arcanum Database Ltd., Budapest, Hungary
[2] Deptartment of Geophysics and Space Science, ELTE Eötvös Loránd University, Budapest, Hungary [timar@caesar.elte.hu]

Daniel Laumer[1], Hasret Gümgümcü[2], Magnus Heitzler[2], Lorenz Hurni[2]

A Semi-automatic Label Digitization Workflow for the Siegfried Map

Keywords: historical maps, vectorization, deep learning, convolutional neuronal network, label extraction

Summary: Digitizing historical maps automatically offers a multitude of challenges. This is particularly true for the case of label extraction since labels vary strongly in shape, size, orientation and type. In addition, characters may overlap with other features such as roads or hachures, which makes extraction even harder. To tackle this issue, we propose a novel semi-automatic workflow consisting of a sequence of deep learning and conventional text processing steps in conjunction with tailor-made correction software. To prove its efficiency, the workflow is being applied to the Siegfried Map Series (1870-1949) covering entire Switzerland with scales 1:25.000 and 1:50.000. The workflow consists of the following steps. First, we decide for each pixel if the content is text or background. For this purpose, we use a convolutional neuronal network with the U-Net architecture which was developed for biomedical image segmentation (Ronneberger, 2015). The weights are calculated with four manually annotated map sheets as ground truth. The trained model can then be used to predict the segmentation on any other map sheet. The results are clustered with DBSCAN (Ester, Kriegel, Sander, & Xu, 1996) to aggregate the individual pixels to letters and words. This way, each label can be localized and extracted without background. But since this is still a non-vectorized representation of the labels, we use the Google Vision API to interpret the text of each label and also search for matching entries in the Swiss Names database by Swisstopo for verification. As for most label extraction workflows, the last step consists of manually checking all labels and correcting possible mistakes. For this purpose, we modified the VGG Image Annotator to simplify the selection of the correct entry. Our framework reduces the time consumption of digitizing labels drastically by a factor of around 5. The fully automatic part (segmentation, interpretation, matching) takes around 5-10 min per sheet and the manual processing part around 1.5-2h. Compared to a fully manual digitizing process, time efficiency is not the only benefit. Also the chance of missing labels decreases strongly. A human cannot detect labels with the same accuracy as a computer algorithm. Most problems leading to more manual work occur during clustering and text recognition with the Google Vision API. Since the model is trained for maps in a flat part of German-speaking Switzerland, the algorithm performs poorer for other parts. In Alpine regions, the rock hachures are often misinterpreted as labels, leading to many false positives. French labels are often composed of several words, which are not clustered into one label by DBSCAN. Possible further work could include retraining with more diverse ground truth or extending the U-Net model so that it can also recognize and learn textual information.

[1] Institute of Cartography and Geoinformation, ETH Zurich, Switzerland [laumerd@student.ethz.ch]
[2] Institute of Cartography and Geoinformation, ETH Zurich, Switzerland

Introduction

The Siegfried Map was the official map series from Switzerland from 1870 to 1949. It is the successor of the Dufour Map which was the first official Swiss map. Some of the main differences is the change from hachures to contours and the introduction of colors. The production included highly tedious work from surveying in alpine terrain and engraving the map contents on different copper plates. The cartographers took much attention to detail and wanted the map not only to be a mere means to an end but also a work of art. Especially the meticulously drawn rock hachures helped to shape the reputation of the famous Swiss cartography style (Bleuer, 1968). The map was produced in two different scales, 1:50'000 in the mountainous regions and 1:25'000 for the lowland. It consists of around 500 different sheets, each spanning around 200 km^2 for the 1:50'000 regions or 50 km^2 for the 1:25'000 regions.

In the efforts of preserving the map and making it available for future generations, each sheet has already been scanned and georeferenced. But to use the full potential of analysis of historic data, current projects focus on the vectorization of the different map elements like buildings (Heitzler & Hurni, 2020), contours or hydrological features. The objective of this project is part of this bigger goal and namely deals with the extraction of the labels together with their respective position. Those digitized labels will help for historical analysis and show how the toponyms changed over time due to spatial development.

Since there are around 500 sheets with multiple versions, amounting to around 4000 sheets in total, a pure manual label annotation process is considered to be unfeasible. Therefore, the main part of the process was designed to be automatic, however a final manual correction step could not be avoided.

Related Work

The interest in digital map processing started in the early 80's. One of the main reasons was the nearly exponential growth of scanned maps in numerous archives. Many disciplines in the natural and social sciences became increasingly interested in using historical maps as a primary resource (Chiang, Unlocking Textual Content from Historical Maps - Potentials and Applications, Trends, and Outlooks, 2017). Also to deeply understand the impact of environmental change, analyzing historical maps can be essential. They can provide a new dimension to the processing of geographic data by expanding from spatial to spatio-temporal analysis (Chiang, Leyk, & Knoblock, 2014).

Label extraction of historical maps is a very difficult task. Labels can vary strongly in shape, form, size, and type. Especially characters overlapping with roads or shading make this even harder. While other features on historical maps like roads or buildings often have a more consistent structure and are easier to extract, labels can vary greatly in appearance. They can be written with different angles, bending or even different spacing.

These issues have been addressed with different approaches. Deseilligny et al. (1995) developed a rotation-invariant character recognition method to overcome the issue that pure Optical Character Recognition (OCR) likely mixed up similar characters. Frischknecht (1996) introduced a software package which extracts symbols and labels on scanned maps with a so-called knowledge-based matching method. To solve the very difficult separation process of text and background graphics, Velázquez and Levachkine (2004) developed a method called V-lines which can also handle curved

labels. An OCR-based recognition algorithm using artificial neural networks is applied to define the coordinates, size, and orientation of the character strings. An increasing trend in image processing research is the use of deep learning algorithms. For example Wick and Puppe (2018) proposed a fully convolutional neural network for historical document segmentation of text and pictures.

Methodology

This project also follows the aforementioned trend of using deep learning algorithms. The workflow is described in *Figure 1*. First we use a previously trained Convolutional Neuronal Network (CNN) to predict a pixel wise segmentation of the map into label and non-label areas (1). To determine which pixels belong to the same word, the areas are clustered using DBSCAN (2). With this information, for each word the corresponding bounding box can be extracted from the map and the background can be removed, so that the word can be read more clearly (3+4). This is then fed into the Google Vision API, which is designed to read text from images (5). This is sometimes still erroneous. To refine the result, we use an existing database of the labels in the current map work. The vicinity is checked for a similar label and recorded if there is a match (6). To determine the final result, the last step is to manually review the labels and correct potential errors.

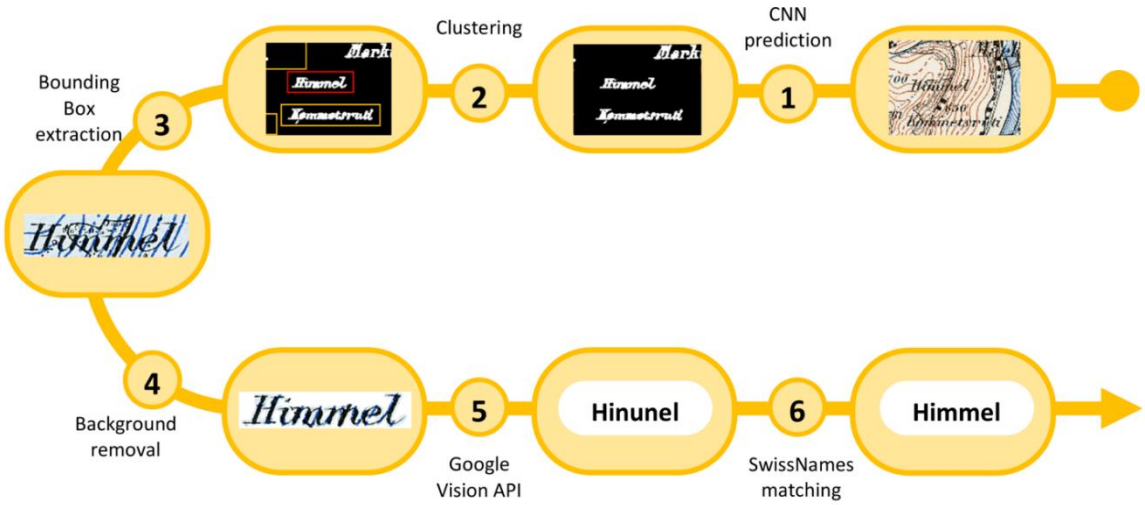


Figure 1. Automatic workflow schematics

Training the Deep Learning Network

A CNN is a deep learning algorithm which can learn to find and classify patterns on images. Several different classes of patterns can be defined for the deep learning model to look for. It takes as input the individual pixels and numerous parameters and outputs to which class each pixel most probably belongs to. The parameters are also called weights and have to be trained beforehand. This training needs ground truth data, meaning map sheets where each pixel is already labelled as text or background. With the help of this, the algorithm can adapt the weights according to the specific problem.

In this project, the ground truth was produced with cumbersome manual annotation of four sheets. Polygons were drawn over each word, following the exact shapes of the letters. This took around one workday per sheet. Of the 4800x7000 pixels, only around 2% represent text. Therefore we have a strongly unbalanced training data (see *Figure 2*).

For the CNN we use the U-Net model which was proposed by Ronneberger et al. (2015) for biomedical image segmentation. As an input it takes images of 256x256 pixels. Therefore, we crop out input sheets into batches of 200x200 pixels and then up-sample them to the required 256x256 pixels. The training consists of iteratively adjusting the weights to better fit the ground truth dataset using gradient descent. The loss function is a measure of how well the model behaves in comparison to the desired values. When the loss function converges to a small enough value, the training is completed and the weights are saved for prediction on other sheets.

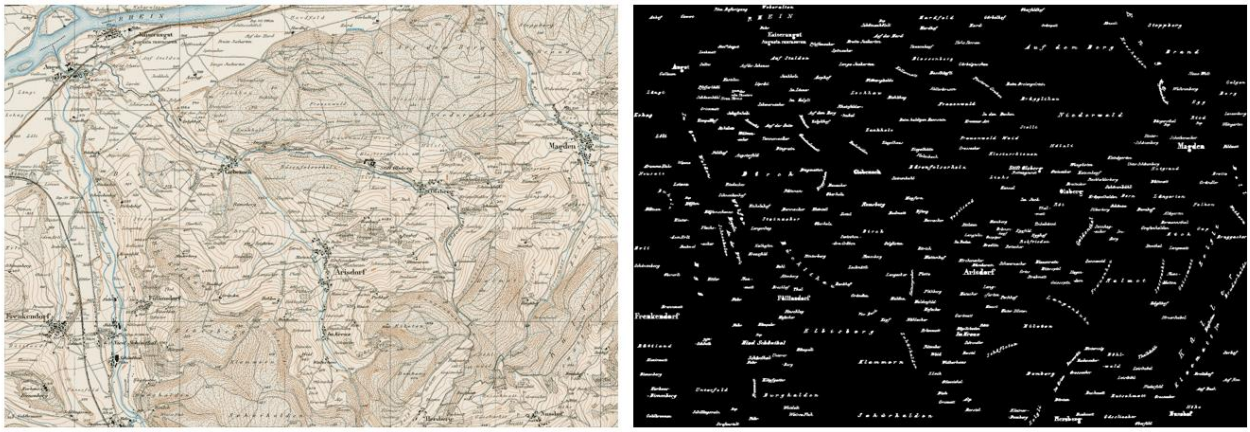


Figure 2. Training data sheet (left) and ground truth (right)

CNN Prediction

When the model is trained, it can be used to predict the segmentation of any other sheet. Before feeding the sheet into the algorithm, it is also first cropped into smaller batches in order to speed up the process. When the network finishes the pixel-wise prediction of the class, the output is a value between 0 and 1. This indicated the likelihood of the pixel to belong to the foreground class. We defined a threshold of 0.5 to define the class for each pixel. All values above are classified as text. The last step is to recombine the small batches to the size of the input sheet.

Clustering Results

To determine which segments belong to a common label, the results of the prediction are grouped using a density based clustering algorithm called DBSCAN. It has the advantage that arbitrary shaped clusters can be found and it is also robust to outliers. This is especially helpful for maps since multiple labels of different form and size can often be found close to each other. The parameters for this clustering method were found empirically and might have to be adjusted for different types of maps. When all clusters are found, a bounding box for each cluster is defined.

Google Vision and Swiss Names Matching

After the location and segmentation of each labels is found, the next step is to understand the content of the text. For this we need a tool provided by Google Vision API (Google, 2020). It allows uploading images containing text and outputs the digitalized text. There are several pre-trained networks which work on many different text fonts and styles.

To prepare the images to be sent to Google Vision, each label found in the clustering step is cropped using the corresponding bounding box and saved as a separate image. In order to reduce errors caused by confusing background patterns, we use the segmentation information to delete the background parts on the cropped images. This makes the text stand out and more clearly visible.

The Google Vision API does not always work correctly. Sometimes there are some wrong or extra letters. To address this issue, the last step of the automatic workflow is to compare it with an existing database. For the up-to-date maps of Switzerland there is an open-data database in vector form available, called SwissNames (Swisstopo, 2013). For each result of the API, the existing database is searched for labels in an area of 1km^2 around the result. Then the label which is most similar is chosen as the best match. As a measure of similarity, the Levenshtein distance was used (Levenshtein, 1966). Over the years the label changed significantly, but for some it can still improve the result produced by the Google Vision API. To decide which label is correct and in which cases to take the matched result from SwissNames, some manual post-processing is needed.

Manual Annotation Process

To make the manual process as efficient as possible, a user interface was created to swiftly click through the images and decide which label is correct. The VGG Image Annotater by Dutta et al. (2016) was used as a base and customized to this specific case. It is a browser-based, simple and lightweight annotation tool which lets the user define and save labels to corresponding pictures in the CSV format. Each image is displayed and the user has to decide if the Google Vision API result, the SwissNames match or neither of them is correct (see *Figure 3*).

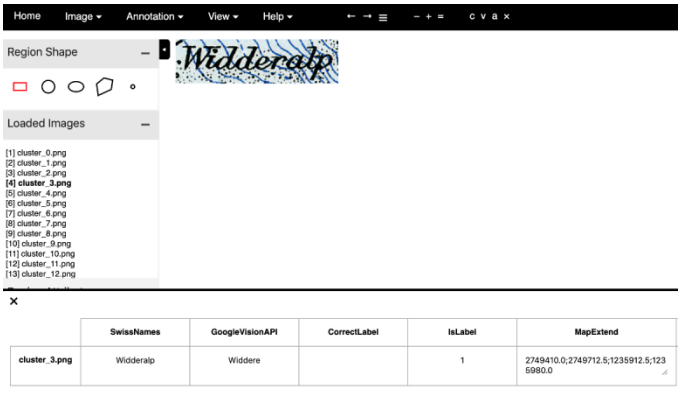


Figure 3. Interface of the adapted VGG Image Annotator

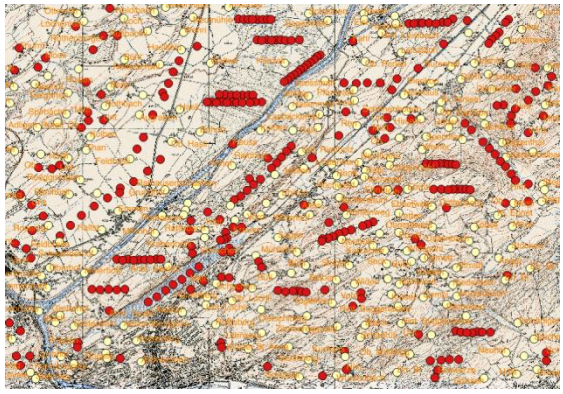


Figure 4. Manual processing in QGIS. Red dots indicate labels which need dome check or correction

Sometimes the clustering algorithm only groups part of the label or several labels together in one cluster. Those cases can be flagged as false and will be corrected in the last step. After the annotation, the results are converted to a shapefile (ESRI, 1998) and loaded into QGIS. There, the incomplete labels are highlighted and can be corrected by placing new labels and filling in the text (see *Figure 4*). Since the sheets are georeferenced, also the location of the labels can be transformed from the local position to a national coordinate reference system. Once all point features are corrected, the shapefile is saved and the labels of this sheet are successfully digitized.

Results & Discussion

Performance

The fully automatic part takes around 5-10 min per sheet. It depends on the number of extracted labels, the GPU power and also the internet connection. The usage of the graphics processing unit (GPU) offers the possibility of a high degree of parallelization. If the many simple calculations needed to train and use a neuronal network would have to be done with a CPU, the time would increase drastically.

The manual processing part containing the annotation and post-processing in QGIS takes around 1.5 to 2 hours. The time strongly depends on the quality of the detections as well on the number of labels in the specific area.

Our framework reduces the time consumption of digitizing labels by around a factor of 5. Compared to a fully manual digitizing process, time efficiency is not the only benefit. Also, the chance of missing labels decreases strongly. We argue that the combination of an automatic procedure followed by a human checking process increases the reliability as well as the accuracy of the detection and classification of labels.

Challenges

There were several problems we had to cope with during the development of this workflow. The first is the scarcity of training data. The manual segmentation of the labels takes up a huge amount of time and was therefore kept to a minimum. Deep learning models however are famously known for needing a lot of training data. This is probably also the reason why the algorithm struggles with the sheets in alpine regions. The rock hachures look similar to label structures and there are many false positives. When a label is missed in the first step, it can only be found again in the last manual processing step in QGIS.

A second issue shows up in the clustering step. In the French speaking parts of Switzerland almost each label consists of several words. Since the clustering algorithm was customized for German speaking regions, where the labels are often one single word, each word is identified as a single label. This increases the manual processing time since those single words have to be merged to multi-word labels. Ideally, the DBSCAN parameters should be adapted according to the different regions.

When matching with the SwissNames database, we encountered some language based problems. In the Siegfried maps the labels were all written in standard German. Nowadays, all text is written in the local dialect, which can greatly differ to original German spelling. This complicates the

matching process. The French spelling also changed in the last 100 years, which always makes another manual intervention necessary to correct the differences.

Outlook

There are several different ideas on how to go on with the project. Firstly, a straight forward approach would be to generate additional and more diverse training data. This would improve the prediction of segmentation and also make it more reliable. Since by now already 300 of the 500 sheets have been fully processed with the workflow presented here, this data could be used as new ground truth. One idea is to tailor the Google Vision step to this specific data by either retraining it or using another text recognition algorithm which is less of a black box and provides more insights and control.

Also the clustering algorithm could be extended by adding additional rules or methods to boost performance and capture the reality more accurately.

Of the several versions of each sheet, for now only the latest sheet has been chosen for processing. To find the labels on sheets of other epochs, a comparison with the digitized epoch could help to find the correct label more efficiently.

References

Bleuer, A. (1968). Zum 100-Jahr-Jubiläum der Siegfriedkarte der Schweiz. *Geographica Helvetica, 23*, 153–156.

Chiang, Y.-Y. (2017). Unlocking Textual Content from Historical Maps - Potentials and Applications, Trends, and Outlooks. In K. Santosh, M. Hangarge, V. Bevilacqua, & A. Negi, *Recent Trends in Image Processing and Pattern Recognition* (Vol. 709, pp. 111–124). Singapore: Springer.

Chiang, Y.-Y., Leyk, S., & Knoblock, C. A. (2014). A Survey of Digital Map Processing Techniques. *ACM Computing Surveys, 47(1)*.

Deseilligny, M. P., Men, H. L., & Stamon, G. (1995). Character string recognition on maps, a rotation-invariant recognition method. *Pattern Recognition Letters, 16*(12), 1297–1310.

Dutta, A., Gupta, A., & Zissermann, A. (2016). *VGG Image Annotator (VIA)*, 1.0.4. Retrieved Feb 10, 2020, from http://www.robots.ox.ac.uk/~vgg/software/via/

ESRI (1998). ESRI Shapefile Technical Description. *ESRI White Paper*.

Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise. *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD'96)*(AAAI Press), 226–231.

Frischknecht, S. (1996). Extraktion beliebig orientierter Kartensymbole und Schriftelemente aus gescannten topographischen Karten durch rotationsinvariante Filterfunktionen und Template Matching. In L. Hurni, *Kartographie im Umbruch - Neue Herausforderungen, neue Technologien* (Vol. 96, pp. 180-186). Schweizerische Gesellschaft für Kartographie.

Google (2020). *Vision - Derive Vision Insights via ML*. Retrieved Feb 1, 2020, from Cloud Vision API: https://cloud.google.com/vision

Heitzler, M., & Hurni, L. (2020). Cartographic reconstruction of building footprints from historical maps: A study on the Swiss Siegfried map. *Transactions in GIS*.

Levenshtein, V. I. (1966, Feb). Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady, 10*, 707.

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. In N. a. Navab (Ed.), *Medical Image Computing and Computer-Assisted Intervention –MICCAI 2015* (pp. 234-241). Cham: Springer International Publishing.

Swisstopo (2013, Aug 15). swissNAMES3D. *Bundesamt für Landestopografie (Art. 30 GeoIV)*. Bern.

Velázquez, A., & Levachkine, S. (2004). Text/Graphics Separation and Recognition in Raster-Scanned Color Cartographic Maps. In J. Lladós, & Y. Kwon, *Graphics Recognition. Recent Advances and Perspectives. GREC 2003* (Vol. 3088). Berlin, Heidelberg: Springer.

Wick, C., & Puppe, F. (2018). Fully Convolutional Neural Networks for Page Segmentation of Historical Document Images. *13th IAPR International Workshop on Document Analysis Systems (DAS)*, 287-292.

Kenzo Milleville[1], Steven Verstockt[1], Nico van de Weghe[2]

Improving Toponym Recognition Accuracy of Historical Topographic Maps

Keywords: Automatic Data Extraction, Computer Vision, Text Recognition, Topographic Maps

Summary: Scanned historical topographic maps contain valuable geographical information. Often, these maps are the only reliable source of data for a given period. Many scientific institutions have large collections of digitized historical maps, typically only annotated with a title (general location), a date, and a small description. This allows researchers to search for maps of some locations, but it gives almost no information about what is depicted on the map itself. To extract useful information from these maps, they still need to be analyzed manually, which can be very tedious. Current commercial and open-source text recognition tools underperform when applied to maps, especially on densely annotated regions. They require additional processing to provide accurate results. Therefore, this work presents an automatic map processing approach focusing mainly on detecting the mentioned toponyms and georeferencing the map. Commercial and open-source tools were used as building blocks, to provide scalability and accessibility. As lower-quality scans generally decrease the performance of text recognition tools, the impact of both the scan and compression quality was studied. Moreover, because most maps were too large to effectively process as a whole with state-of-the-art commercial recognition tools, a tiling approach was used. The tile size affects recognition performance, therefore a study was conducted to determine the optimal parameters. First, the map boundaries were detected with computer vision techniques. Afterward, the coordinates surrounding the map were extracted using a commercial OCR system. After projecting the coordinates to the WGS84 coordinate system, the maps were georeferenced. Next, the map was split into overlapping tiles, and text recognition was performed. A small region of interest was determined for each detected text label, based on its relative position. This region limited the potential toponym matches given by publicly available gazetteers. Multiple gazetteers were combined to find additional candidates for each text label. Optimal toponym matches were selected with string similarity metrics. Furthermore, the relative positions of the detected text and the actual locations of the matched toponyms were used to filter out additional false positives. Finally, the approach was validated on a selection of 1:25000 topographic maps of Belgium from 1975-1992. By automatically georeferencing the map and recognizing the mentioned place names, the content and location of each map can now be queried.

Introduction

As more and more historical data is being digitized, the need for automated processing techniques grows across all fields of research. Having an easily-accessible, machine-readable data collection allows researchers to query and analyze this collection much more efficiently. It provides a window into the past and opens new research opportunities. Manually annotating vast amounts of digitized data is a very tedious and time-consuming process. Many institutions have a (large) collection of

1 IDLab, Ghent University – Imec, Belgium [kenzo.milleville@ugent.be]
2 CartoGIS, Ghent University, Belgium

digitized historical maps, which are often only annotated with a title, general location, a date, and a small description. This allows researchers to search for maps of some locations, but it gives little to no information on what is depicted on the map itself. To extract useful information from these maps, they still need to be analyzed manually. By automatically processing and extracting the place names and georeferencing the map, the content can be queried as well. The extracted place names can be used to situate the map more accurately and improve query results. Map processing techniques aim to extract the text labels and geographic features from a raster map, to enable subsequent manipulations in a geographic information system (GIS) (Chiang et al. 2014). These techniques make the map more accessible and allow for a large-scale analysis of entire collections.

In this work, we propose an automated map processing approach, capable of extracting place names and georeferencing topographic maps. The goal is to apply the methods described in this work on Atlas[3], the online digitized cartographic library of Ghent University. Currently, the maps are annotated with their name, type, country, and scale, but no information is given on what is depicted on each map. We apply and validate our approach on a selection of topographic maps from Belgium. This work uses open-source and commercial text recognition tools and gazetteers as building blocks, making the approach more scalable and accessible. Furthermore, a study is made on the impact of the compression quality of the maps and the tile size used to analyze these maps.

Text recognition on maps is not a trivial task as it comes with additional challenges. Text labels are mostly black and aligned horizontally, but they can appear in multiple colors, orientations, sizes, and fonts (Deseilligny et al. 1995). In historical maps, the text labels are often handwritten, making them sometimes hard to transcribe, even for a human. A non-uniform background and the overlap of other map features with the text labels further reduces recognition accuracy. This is especially true for topographic maps, which contain a multitude of geographical features (roads, contour lines, waterways, vegetation, etc.) and text labels (place names, elevation data, street names, etc.) (Pezeshk and Tutwiler 2011). An example of the variety in text labels is given in *Figure 1*. Topographic maps are designed to give a ‘good’, general view of the landscape using multiple colors, symbols, and text labels (Kent 2008). These maps are relatively easy to read and interpret for most people but are much harder to process automatically. Gazetteers can be used to match recognized text with real-world place names, improving the annotation quality. When the map is georeferenced, a region of interest can be specified based on the relative text location, to limit the potential candidates to that region. The recognized text is then matched with a toponym candidate by using string similarity metrics.

Quality Analysis

This section details a small study on the impact of the scan and compression quality on the text recognition results. The section starts with a description of the used dataset. Afterward, it details the used approach for the quality analysis study and the chapter ends with a discussion about the results.

[3] https://www.atlas.ugent.be/

Figure 1. Example of some text labels from one of the topographic maps. These text labels can come in different colors, orientations, and sizes, complicating automated processing techniques.

Dataset

The dataset consists of a collection of 1:25000 topographic maps of Belgium from 1975-1992 (series M834, second edition) (De Maeyer et al. 2004). Because these maps are from the same series, their uniformity improves this automated process, but most of the proposed techniques can be applied to other maps as well. The toponyms present on the map were manually labeled with the corresponding transcription and a non-rotated bounding box. Each scanned map contains a legend, which details the used abbreviations, symbols, coordinates, and much more. The map itself is surrounded by a black rectangle, on which coordinate information is given. Three of the maps were labeled in order to validate the developed approach. Each map took around 2-4 hours to label. One of the topographic maps from the dataset is shown in *Figure 2*, it details the region Gent-Melle.

Preprocessing and Georeferencing

To perform the quality analysis, the map itself still needs to be extracted from the original scan. Because the maps are from a uniform series, the position of the map stays constant for each scan. However, the scans are not aligned perfectly, therefore the location of the map does vary slightly with each image. By detecting the black rectangle surrounding the map with morphological operations (Weeks 1996), we were able to consistently extract the map and its surrounding coordinates.

First, the map was converted to grayscale, binarized with a threshold of 127, and inverted. Afterward, a vertical and a horizontal structuring element (kernel) were defined. Both were 100 pixels long and 1 pixel wide. Next, for each structuring element, an erosion was performed followed by a dilation operation (Weeks 1996). The two masked images were then merged with a bitwise OR operation. Because the lines were not perfectly horizontal or vertical, this resulted in multiple line segments per scan. These segments were then filtered; the longest and broadest one was taken. Finally, the segments were extended until their ends formed a rectangle. This resulted in the location of the surrounding black rectangle. If some of the maps were heavily rotated during the scanning process, a line detection method should be used instead to detect the bounding rectangle.

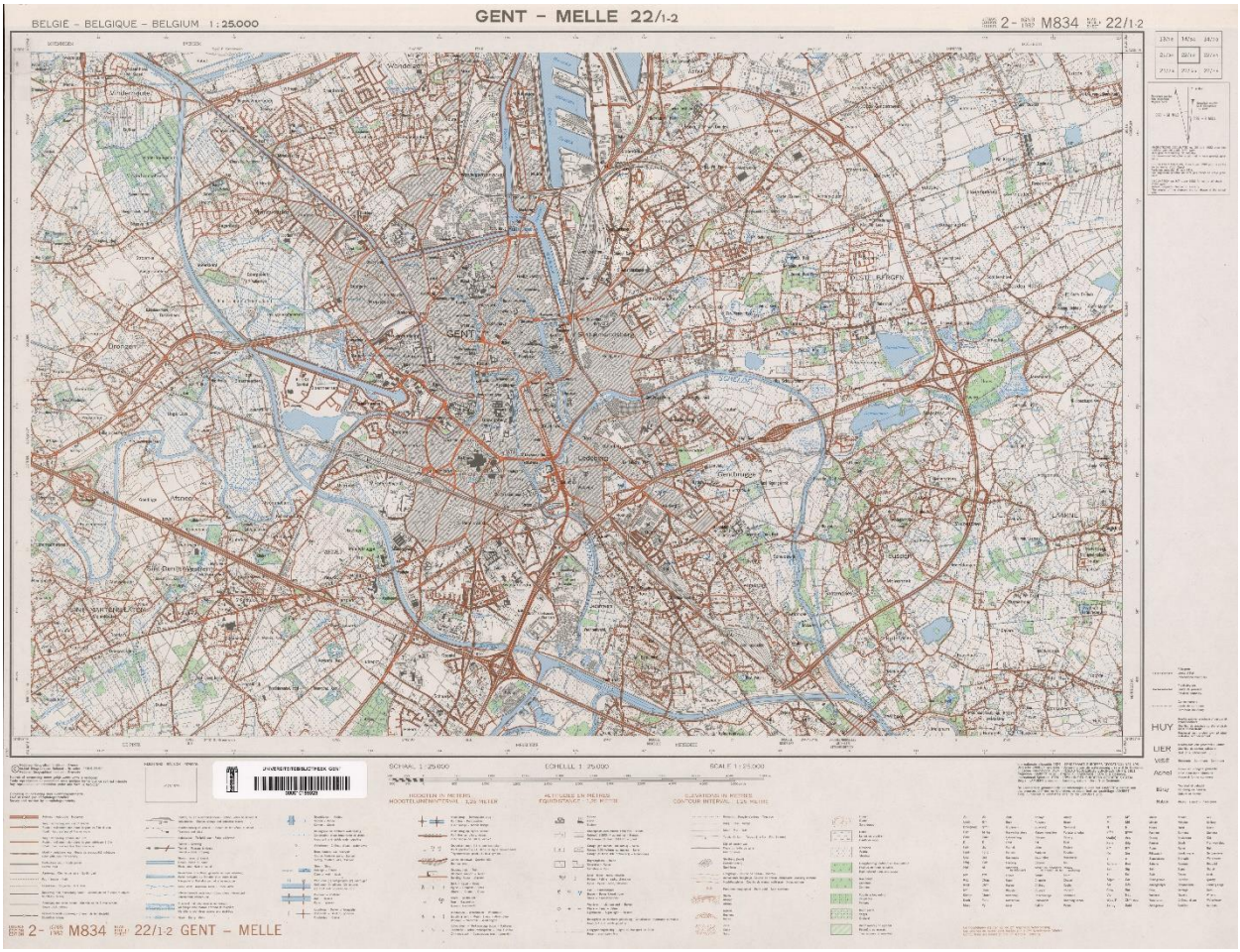


Figure 2. The topographic map for the region of Gent-Melle.

Inside the rectangle, there was still some blank space, where the coordinate information and neighboring cities are denoted. To detect the effective bounding box of the map, crops were extracted from all four sides of the rectangle. Each side was split into multiple tiles because they were too large to process effectively. Each tile was analyzed with a commercial text detection system (Azure Read API), from which the locations of the coordinates were extracted. Finally, the locations of the surrounding coordinates were used to define the map location. *Figure 3* demonstrates the output of the text detection system. The numbers at the edges of the map note the x and y location in the Belgian Lambert72 projection (Donnay and Lambot 2012).

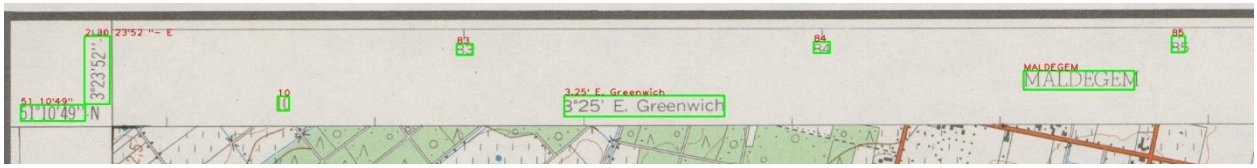


Figure 3. Result of performing text detection on the upper-left tile of the map coordinates. The upper coordinates (83, 84, and 85) represent the horizontal coordinates in the Belgian Lambert72 projection.

Both the horizontal and vertical Lambert72 coordinates surrounding the map were used to georeference the map. Because the text recognition is not perfect and there exists some inaccuracy in the location of each coordinate (the center of each coordinate text box is not the same as the location of its mark), a simple averaging method was made to reduce the error. The pixel and coordinate distance between all the succeeding coordinates were taken and averaged, to find a conversion factor between the pixel distance and the corresponding coordinate distance. With the determined conversion factor, each pixel's corresponding coordinate location can be found. Of course, the georeferencing will not be perfectly accurate, as that is near-impossible to accomplish without human intervention from a scanned map.

Compression and Tiling Parameters

Now that the location of the actual map is extracted, a study on the compression and tiling parameters can be performed. Four different quality settings were tested for each map: original (png, 350 DPI), half resolution (png, bicubic interpolation) and jpeg compression with and without compression of each tile to jpeg (denoted as jpeg_jpeg). The original and half resolution maps were converted into the lossless png format from the original tif format scans and a quality setting of 95 was used for all the jpeg compressions. The lossless png compression is often recommended for OCR, especially on segmented text images, to not introduce additional noise.

As stated before, the maps are too large to process as a whole with any commercial text detection system, therefore a tiling approach was used. By varying the size of each tile, a trade-off is made between cost and quality. Smaller tile sizes will generally be more accurate, whilst increasing the total amount of tiles to be processed by the system. Halving the tile size effectively multiplies the number of tiles by four. The following square tile sizes were considered in our study: 1000, 1500, 2000, and 2500 pixels. Tiles smaller than 1000x1000 pixels were not considered, as this size already resulted in more than 50 tiles for each map. To not introduce additional compression errors, each tile was saved in the lossless png format (except for jpeg_jpeg). To limit edge errors, the tiles did not overlap, and text labels at the edges of the tiles (distance of 20 pixels or less) were ignored.

Each map was also processed with a state-of-the-art, open-source text recognition system, for each of the compression and tiling settings. The model from Liu et al. (2018) was used, which has an open-source Tensorflow implementation and a pretrained model available on Github[4]. The model performs text detection & recognition and outputs text-oriented bounding boxes.

The Character Error Rate (CER) is defined by the Levenshtein distance (edit distance) between the predicted text and its corresponding label, divided by the length of the label (Bluche 2015). It is one of the most common and important metrics to determine the performance of an OCR system. To determine which label corresponds with each predicted text box, the intersection over union (IoU) (Everingham et al. 2015) was used. The IoU gives a general indication of how well the position of the predicted text matches the position of the text label. Labels that did not have a corresponding prediction were ignored when calculating the average CER. When two or more predictions intersect the same label, the prediction with the lowest CER was taken. *Tables 1 and 2* show the results of this study using the commercial and open-source recognition systems, respectively. The topographic maps contain a lot of small text denoting height contours or kilometer milestones, which were often

[4] https://github.com/Pay20Y/FOTS_TF/tree/dev

not detected correctly. The results indicated in the tables below with an asterisk (*) exclude these small text labels, leaving only toponyms and abbreviations which indicate local features (e.g. chapel, station, water tower, etc.). The three maps contain a total of 2968 text labels, of which 1818 denote a toponym or abbreviation.

Table 1. Performance of the commercial text recognition system (Azure Read API) with varying compression and tile sizes. Results indicated with an asterisk (*) excluded text labels denoting height contours or kilometer milestones.

Tile Size	Compression	IOU	CER	Found (%)	IOU*	CER*	Found (%)*
1000	Original	0.598	0.301	**54.8**	0.627	0.227	**78.4**
	jpeg	0.599	0.293	54.2	0.629	0.221	77.7
	jpeg_jpeg	**0.6**	0.294	54.4	**0.63**	0.217	77.8
	Half resolution	0.587	**0.239**	41.7	0.604	**0.198**	63.7
1500	Original	0.58	0.267	49.4	**0.605**	0.217	72.4
	jpeg	0.578	0.275	**49.9**	0.601	0.223	**73.2**
	jpeg_jpeg	0.58	0.271	49.6	**0.605**	0.215	72.8
	Half resolution	**0.595**	**0.216**	23.2	0.6	**0.201**	36.7
2000	Original	0.58	0.264	**44**	0.598	0.217	**66.2**
	jpeg	**0.582**	0.259	43.7	**0.601**	0.212	65.6
	jpeg_jpeg	**0.582**	0.27	**44**	0.599	0.212	65.8
	Half resolution	0.58	**0.247**	15.6	0.593	**0.209**	23.7
2500	Original	0.594	0.239	32.8	**0.609**	**0.204**	49.8
	jpeg	**0.595**	0.242	**33.4**	0.608	0.21	**51.1**
	jpeg_jpeg	**0.595**	0.241	33.2	0.608	0.209	50.7
	Half resolution	0.577	**0.218**	5.1	0.58	0.216	8

Table 2. Performance of the open-source text recognition system with varying compression and tile sizes. Results indicated with an asterisk (*) excluded text labels denoting height contours or kilometer milestones.

Tile Size	Compression	IOU	CER	Found (%)	IOU*	CER*	Found (%)*
1000	Original	0.516	0.351	52.9	0.518	0.3	77.4
	jpeg	0.519	0.347	**53.8**	0.522	0.294	78.8
	jpeg_jpeg	0.523	**0.346**	**53.8**	0.524	**0.292**	**79.2**
	Half resolution	**0.55**	0.395	21.3	**0.551**	0.363	33.6
1500	png	0.521	0.347	54.8	0.523	0.298	79
	jpeg	0.526	0.347	55.3	0.528	0.291	79.4

	jpeg_jpeg	0.527	**0.343**	**55.6**	0.529	**0.286**	**80.1**
	Half resolution	**0.551**	0.397	22.9	**0.551**	0.366	36
2000	Original	0.521	0.354	54.5	0.524	0.295	78.1
	jpeg	0.525	0.348	55.9	0.528	0.286	79.4
	jpeg_jpeg	0.526	**0.343**	**56.3**	0.53	**0.284**	**79.9**
	Half resolution	**0.548**	0.384	23.8	**0.55**	0.347	37
2500	Original	0.523	0.347	55.4	0.526	0.294	78.7
	jpeg	0.528	0.337	55.8	0.531	0.286	80
	jpeg_jpeg	0.53	**0.336**	**56**	0.533	**0.284**	**80.3**
	Half resolution	**0.557**	0.404	21.8	**0.559**	0.365	34

From these results, we can conclude that for both the open-source and commercial system, there is no significant difference in performance when using jpeg compression for the map and/or tiles. Therefore, it is advised to use jpeg compression when analyzing a large dataset, to significantly reduce storage space (average compression ratio of 4.4:1). It is also clear that the lower resolution scans underperform dramatically, which was expected, as most OCR systems recommend a minimum scan resolution of 300 DPI. The CER might be lower, but the amount of found toponyms is usually halved. If we were to include the undetected words in the calculation of the CER (each would have a CER of 1), the error rate would dramatically increase. For both systems, the performance increases when leaving out the small text labels denoting the height contours and kilometer milestones. This was expected, as these text labels are very easy to miss, even when manually annotating the maps. The IoU remains relatively constant across all tile sizes.

Increasing the tile size for the commercial system leads to a lower number of detected toponyms but decreases the transcription error rate slightly. Perhaps only the most clearly visible toponyms are recognized correctly. For the open-source system, the CER also decreases slightly, but the number of found text labels stays relatively constant. Moving forward, the open-source system is used with a tile size of 2500 pixels. The larger tile sizes give a marginally better performance and introduce fewer errors on the tile edges, as there are fewer tiles that will need to be merged.

Merging adjacent tiles

When processing the topographic maps in tiles, several toponyms will be split on the edges of each tile. To solve this problem, an overlap region of 500 pixels is considered in each 2500x2500 tile. Because the maps are processed from left-to-right, top-to-bottom, the overlap is taken at the right side and bottom side of each tile. Recognized text which starts inside this overlap region is ignored and should be detected in one of the next tiles. Text which started before the overlap region but ends in it will need to be merged with the detection result of the next tile. If the next tile contains similar text (partial string similarity) and intersects the other label, the longest of the two detected strings is taken, the other is discarded. If the suffix of the first string exactly corresponds with the affix of the

second, they were instead merged on their longest common substring (e.g. “Sint-Pie” and “ikt-Pieter” will be merged to “Sint-Pieter”). After merging, the results were saved for further processing.

Toponym matching

After detecting and merging the text labels of each map, the detected text on each map can now be queried. Because of the relatively high recognition error rate, there will be a lot of false positives and the query results will generally be of low quality, as similar text might be recognized on multiple maps at different locations. To remove many of these irrelevant detections, the text labels were first preprocessed. Short, non-text labels were filtered out. All the labels were put in lowercase and symbols (excluding spaces) were removed from the strings. Next, if a text label contained an abbreviation, it was replaced with its full form. This is necessary to find a correct match with certain places of interest. To further improve the query results on each map, they can be adjusted to only return text labels that can be linked with their corresponding toponym and whose relative position on a georeferenced map corresponds with the actual location of the toponym. These linked toponyms have the additional benefit that they can be queried in multiple languages, as most gazetteers provide multilingual support. Three different gazetteers were used to determine the correct matches for each text label, namely Geonames, Google Maps, and TomTom. All three have a public API and allow for limited free usage. The gazetteers were queried with a fuzzy search, allowing for approximate and partial string matches. Moreover, a rectangular region of interest was specified, limiting the results to that area. For each text label, this region of interest was calculated based on its relative position on the map and the already determined georeferencing parameters. The square bounding box of a circular region with a radius of 2km was chosen, with its center taken at the center of the text bounding box.

To limit the number of gazetteer requests, they were queried sequentially. When the first one did not return a good match, the second one was tried, and so forth. A match was classified to be good when the partial string similarity score (0 - 1) between the query and the results was higher than a certain threshold (0.75). If there were multiple good matches, the one with the highest score was taken. In the case of a tie, the shortest toponym match is taken, as it is usually the better match. When the similarity score falls below this threshold but was still relatively high (0.6 - 0.75), the results were classified as possible matches. Increasing these thresholds will reduce the false positive rate but will decrease the number of results. An additional filter based on the relative string lengths was made to limit incorrect results as shorter query strings would sometimes return very long possible matches.

A large portion of the toponyms on the maps consist of multiple words. These words are often recognized as separate words and therefore do not find a correct match. To solve this, we checked for each detected text label without a correct match for other nearby unmatched text to the right and bottom side, as this is the usual reading direction for topographic maps. If nearby text was found, both of the strings were joined and a new query was made with the gazetteers. In most cases, no correct match was found, mainly due to errors in the text recognition and unrelated nearby words. To limit the number of gazetteer requests, we only tried combinations of two words. There are multiple toponyms on the map that contain 3 or more words, which were therefore not correctly linked. Often multiple possible logical arrangements exist between the words, increasing the number of requests exponentially. These complex situations are difficult to solve without a brute-

force method. *Figure 4* shows an example of such a complex situation. Words (or pairs of them) that have found a potential match are marked in green, words without a match are marked in red. Here, the words in blue ("Oude Leieput") should be merged, and "Hbg Karper" should be merged, the others are merged correctly. If the text color can be accurately determined, such false positives can possibly be filtered out. The fact that some of the correct toponym matches and their arrangements are not found in publicly available gazetteers, makes this process even more challenging.

Finally, duplicate matches of toponyms are resolved. When two or more detected text labels match with the same toponym, the least likely match should be removed. If the difference in string similarity between each label and the toponym is large, the most similar one is taken. If the difference is small, the text label whose relative position on the map corresponds the best to the actual position of the toponym is taken as the correct match. On average, 26% of the detected text labels (excluding irrelevant detections) found a good or a possible toponym match.

Figure 4. Complex arrangement of toponyms. Words (or pairs of them) that have found a potential match are marked in green, words without a match are marked in red. Here, the words in blue should be merged ("Oude Leieput"), and "Hbg Karper" should be merged, the others are merged correctly.

Conclusion

By automatically recognizing the surrounding map coordinates, we were able to georeference the analyzed topographic maps. The results of the quality analysis study show that the difference in text recognition performance for jpeg and png compressed maps and their extracted tiles is insignificant. A lower resolution of the maps does negatively affect the character error rate and the number of detected text labels. When using a commercial text recognition system (Azure Read API), a larger tile size resulted in the detection of far fewer text labels, with a similar error rate. We suspect this is due to some assumptions made about the possible text size, given the image dimensions. When using the open-source text detection system, a larger tile size was found to be marginally better, while also reducing the total number of tiles and merge conflicts. At the smaller tile sizes, the commercial API outperforms the pretrained open-source model. Still, both models vastly underperformed when compared to other text detection domains, where error rates of less than 10% are commonly achieved. This highlights the need for more robust text recognition models, which can handle the complex background of topographic maps or additional preprocessing to extract the text from the background.

To improve the quality of the extracted text labels, gazetteers were used to match them with toponyms in the local area. Due to the complexity of these topographic maps and the relatively high character error rate, many detected text labels could not be automatically matched to a toponym. A lot of mentioned text labels do not even have a corresponding toponym in any of the used gazetteers, which makes it impossible to find a correct match. When analyzing older historic maps, or maps of very remote regions, we suspect that this will pose an even bigger issue. Nevertheless, by georeferencing, text recognition and toponym matching, the mentioned place names and location of the map are found. This enables the contents of the map to be queried in much greater detail.

In future work, we aim to perform a deeper quality analysis to determine if higher scan resolutions will result in a lower recognition error rate. Performing transfer learning on the used text recognition model on a large dataset of annotated maps might also drastically improve its performance. The linking and merging of related words and toponyms can still be improved by incorporating text features (colors, font, relative location, etc.) and by improving the quality of the used gazetteers. The biggest problem is most likely the fuzzy search capabilities, which often do not return the correct toponym if the first letters are wrongly recognized (e.g. “ent” does not return “Gent”). Manually comparing detected text against a database of toponyms might therefore give better results.

References

Chiang, Y. Y., Leyk, S., & Knoblock, C. A. (2014). A survey of digital map processing techniques. *ACM Computing Surveys (CSUR)*, *47*(1), 1-44.

De Maeyer, P., De Vliegher, B. M., & Brondeel, M. (2004). *Spiegel van de wereld*. Academia Press.

Deseilligny, M. P., Le Men, H., & Stamon, G. (1995). Character string recognition on maps, a rotation-invariant recognition method. *Pattern Recognition Letters*, *16*(12), 1297-1310.

Pezeshk, A., & Tutwiler, R. L. (2011). Automatic feature extraction and text recognition from scanned topographic maps. *IEEE Transactions on Geoscience and Remote Sensing*, *49*(12), 5047-5063.

Kent, A. J. (2008). Cartographic blandscapes and the new noise: Finding the good view in a topographical mashup. *The Bulletin of the Society of Cartographers*, *42*(1), 2.

Weeks, A. R. (1996). *Fundamentals of electronic image processing*. SPIE Optical Engineering Press.

Donnay, J. P., & Lambot, P. (2012). Geodetic and cartographical standards applied in Belgium. *A Concise Geography of Belgium*, 41-42.

Liu, X., Liang, D., Yan, S., Chen, D., Qiao, Y. and Yan, J., 2018. Fots: Fast oriented text spotting with a unified network. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5676-5685).

Bluche, T. (2015). *Deep neural networks for large vocabulary handwritten text recognition* (Doctoral dissertation, Paris 11).

Everingham, M., Eslami, S. A., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2015). The pascal visual object classes challenge: A retrospective. *International journal of computer vision*, *111*(1), 98-136.

Alexandre Nobajas[1]

Targeted crowdsourced vectorisation of historical cartography

Keywords: crowdsourcing, gamification, systematic vectorisation, training set creation

Summary: Many institutions have systematically digitised their cartographic documents, so there are now millions of maps which have been digitised and can be accessed on-line. The next step many map libraries have undertaken has been to georeference their digitised maps, so they can now be used within modern digital cartographic dataset. However, many of these applications still use the raster version of the maps, so they hamper the full potential working with historical cartography has, as it limits the analytical capacity and impedes performing spatial analysis. Even though there have been timid steps to overcome this limitation, the discipline seems to be struggling to advance into the next step, the systematic vectorisation of historical maps. Vectorisation is the process of converting pixel-based images –raster- into node or point based images –vector, which can then be queried or analysed by individual components. Vector based maps have a series of advantages when compared to raster maps, as they allow scale changes without a loss in detail, classifying map features by type or performing spatial queries just to mention a few. Therefore, by vectorising a historical map we are providing an unprecedented level of usability to it and allowing detailed inquiries to the information contained into it. Although there have already been some successful experiences in vectorising historical maps, they have been either expensive or not systematic, so there is scope to build upon these successful experiences and achieve a widespread method of vectorisation which can be applied to map collections across the world.

There are different approaches to overcome this technological and methodological hurdle, most of them linked to the use of automation techniques such as using remote sensing land classification methods or, more recently, the use of machine learning and artificial intelligence techniques. However, a large number of training sets are necessary in order to train the automation systems and to benchmark the outputs, so large datasets are still needed to be generated. The use of crowdsourcing methods has been proven as a successful means of achieving the same goal, as experiences such as the ones developed by the New York Public Library demonstrate. If given the possibility and the right tools, people will help to vectorise historic documents, although Nielsen's 90-9-1 rule will apply, meaning that other ways of increasing participation are necessary in order to speed the vectorisation up.

This paper proposes a framework in which by using a variety of gamification methods and by targeting two different groups of people, retirees and school children, historic maps can be massively vectorised. If fully deployed, this system would not only produce a wealth of vectorised maps, but it would also be useful in teaching school children important disciplines such as cartography, geography and history while at the same helping older generations keeping them mentally active. Finally, an intergenerational dialogue that would enrich both participant groups would be possible thanks to their common participation in the vectorisation of historical documents.

[1] School of Geography, Geology and the Environment, Keele University, United Kingdom [a.nobajas@keele.ac.uk]

Jonas Luft[1]

Automatic Georeferencing of Historical Maps by Geocoding

Keywords: georeferencing; geocoding; label extraction; place-names; historical maps

Summary: Libraries and researchers face a big challenge: sometimes thousands, millions of maps are being conserved in archives to retard deterioration, but this makes it hard to obtain specific knowledge of their content, sometimes even of their existence. In recent years, the efforts have increased to digitise historical documents and maps to preserve them digitally, make them accessible and allow researchers and the general public a less restricted access to their heritage. But maps are more than a cultural artifact: they are data. Data about the past that can very well be important for science and decision-making today. The problem is, the vast amount alone makes it hard to know what to look for. Maps are usually archived and catalogued with limited meta-information, sometimes obscuring their actual content. This makes it impossible to find specific information without expert knowledge on history and cartography. Extracting the content, i.e. data, of scanned historical maps is a necessity to make the content searchable and to use modern digital tools for automatic processing and analyses. To combine data from different maps and compare them with modern geospatial data, georeferencing is paramount. This is usually done by hand and needs a lot of time and specialised training. We explore if and how usable GCP can be found automatically in historical maps with computer vision and document analysis methods of today. In this work-in-progress report, we use OCR on map labels and geocoding of persisting geographical feature designation for successful first experiments. This shows text recognition and vectorisation to be a promising research direction for large-scale automated georeferencing of historical maps.

Introduction

Until only few years ago almost any motorist used a road atlas to navigate. They usually contained an index of cities, road names or other destinations along with their corresponding location in the atlas and the respective map sheet. This index, also called a *gazetteer* in other map types, enables a quick localisation of named places in the map's respective reference system. Recently, gazetteers in this form have become somewhat less common. Instead, we now have access to a plethora of geocoding web-services which can be queried with search terms to identify and localise almost any location all over the world.

Today, historical maps are catalogued according to their metadata and not their content. To make spatial queries (in the form of "all 19th century maps showing Hamburg") or spatial analyses possible, they have to be georeferenced (Schröder 2013). To georeference a scanned map, matches between a set of so called *ground control points* (GCP) in the image and actual geographical locations have to be found.

After automatically extracting and recognising place names from a map and querying their respective locations in aforementioned geocoding services to obtain geocoordinates, we are able to

[1] CityScienceLab, HafenCity University Hamburg [jonas.luft@hcu-hamburg.de]

automatically georeference scanned historical maps. The goal is to thus enable batch processing of large series of maps, such as late modern period land surveys. To make automatic handling feasible with minimal human input, homogeneous graphical quality and design is required. Therefore we also investigate which properties of maps are necessary to allow this approach to be used.

Related work

Automatic matching of similar features in different images is a classic task of computer vision and has been thoroughly researched for spatial data in photogrammetry and applied to historical data (Chen et al 2016). Of special interest is automatic registration of heritage aerial images with modern day data, which are hard to match, since they use different colour spaces (black and white vs. colour images), often have quite different quality (analog vs. digital images, old images scanned from photographic materials), but especially because a significant change in the environment and built structures have to be expected. To overcome these issues Cléry et al. (2014) use line feature matching to register old aerial images with current georeferenced orthoimages, to georeference the historical data.

Interesting research has been done by Chen et al. (2004) into registering orthoimages with vector data, by automatically extracting road intersections from the raster image and matching those to vector data of the same scene for very high precision results.

To generalise this approach to unknown scenes Briggs and Li (2006) employ topological point pattern matching to find road intersections extracted from a raster map in a much larger dataset of vector data.

Wolter et al. (2017) have proven that place names in maps are very descriptive and can be used for identification and localisation of settlements and river tributaries. By using spatial reasoning, they have developed a method to automatically identify a river network in a map.

So far there has been limited research into automatic methods for extraction of control points to be used in registration and georeferencing that take the special difficulties of early modern era maps that are hand-drawn or lithography-printed, into consideration. In one prime example (Heitzler et al. 2018), the map boundary and grid lines are automatically detected by template matching and Hough line detection and subsequently matched with coordinate labels to automatically rectify and georeference the Siegfried map.

It remains open, how to deal with maps that don’t show grid lines, are too cluttered to extract them successfully, are drawn with unknown projection and datum or are distorted so extremely that given coordinates can’t be trusted.

Problem Definition

For most computer vision tasks, high-contrast points in images are used to build robust feature descriptors, which have over the years been the basis of increasingly better performance on detection and classification tasks. Increasingly complex object representations enabled by machine learning accurately recognize objects in cluttered scenes. The special property of maps on the other hand, which distinguishes them from almost every other computer vision task, is that they have already been optimised for human use. In order to improve readability of maps in spite of a high

density of information, mapmakers have developed abstract and generalised coding, almost completely removing latent information from the image texture. The steep gradients on the variety of overlaid lines, symbols and region boundaries make a map a homogeneously high contrast, high clutter image with ambiguous repeating patterns prohibiting the use of state-of-the-art computer vision descriptors for uniquely defining objects.

The biggest challenge for the application computer vision methods on topographic maps is that the possible figure of interest (i.e. foreground) exhibits similar properties to the (back-)ground. In particular, when we are interested in text content, it is very hard to distinguish from the remaining content of the map, such as contour lines, roads, symbolic features and building outlines.

A first step for figure-ground-segmentations can be the use of colour information to separate multiple feature layers (e.g. bodies of water in blue, contour lines in brown, writing in black) of the map. But since maps are usually drawn with only a limited palette of colours this will still result in overlapping features. This is an expected problem especially with old maps produced in lithography. Since creation of multiple printing plates immensely increased the production cost without significantly increasing readability, it is quite common to see maps with only three to four colours.

The resulting ambiguous use of (most commonly) black colour for writing as well as grid lines, special symbols and building outlines can make a complete figure-ground-segmentation impossible and severely impairs the quality of *optical character recognition* (OCR) results.

Machine-reading in maps poses some unique problems as a result of the apparently arbitrary positioning of text labels. While document analysis methods can usually rely on homogeneous line and character orientation and spacing to resolve ambiguities in character recognition, the page segmentation on maps is much harder: we have to deal with multiple fonts in multiple sizes, sometimes widely spaced and overlapping each other; text labels can have unusual orientations and even be curved to follow e.g. the path of a river. All of these make it especially difficult to know which two (or more) labels belong together when a designation has been split over multiple “lines” and placed in seemingly unrelated locations.

Finally, there are some unique problems with historical maps: some features are not present anymore in modern data, sometimes the orthography has changed and historical toponyms might not be in use anymore, all preventing reliable matching of place names.

Method

Our processing pipeline works in five consecutive steps: firstly, image preprocessing to equalise different graphical quality, then detection of text labels with their bounding boxes. Afterwards OCR on said text labels results in actual text strings. In the fourth step, geocoding by querying for recognised text strings thus returns geo-coordinates. Finally transformation and rectification is accomplished with the centers of the bounding boxes and geocoded coordinates as GCP-pairs. These steps will be described in more detail in the following section.

Map preprocessing

As a first step, we apply automatic white balancing by histogram equalisation, to limit the discolouration by paper degradation and allow all further processing steps to work on the full range of colour values.

Maps are commonly printed with a small palette of three to four colours, designed to be easily told apart by the human eye. But when scanning these maps with high resolution and subsequent rasterising, sharp colour transitions get transformed into gradients, resulting in image files with a lot more colours. This makes colour quantisation a non-trivial effort, with the risk of losing too much information, especially on the edges of text, which need to be preserved for successful application of OCR. Three-dimensional k-means clustering on the RGB-space showed unconvincing results because of run-time complexity on large map images and large differences in cluster sizes (e.g. many black pixels, very few blue pixels in many images).

Instead, our approach uses selective gaussian blur to eliminate noise from image compression, smudge hachure lines into a uniform colour and merge the printing dots on very high resolution scans. After an unsharpening filter to revert blurring of edges, a simple but inaccurate colour separation can be obtained by thresholding.

The resulting figure-ground-segmentation remains incomplete, since by colour only some of the features can be removed, while many map elements sharing the same colour as the desired text remain. Examples include roads passing through their designations as well as grid lines and building outlines intersecting place names. Humans have little difficulty separating text from underlying clutter, since their visual perception can rely on Gestalt-principles (Koffka 1935), in particular the law of continuity and the law of past experience, which have yet to be successfully formalised in a computer system.

Some research in text-graphics-separation has been done to tell these conflicting figures, such as text characters and lines, apart on maps, but so far showed unconvincing results even on relatively “clean” datasets, such as computer generated maps from digital vector data (Chiang and Knoblock 2006).

Text Extraction

Notwithstanding the unsatisfactory segmentation, machine learning systems have successfully been trained to detect text in cluttered scenes (Zhou et al. 2017). There has been some research into the peculiarities of detecting text in maps, in particular multi-oriented text in a single document (Roy et al. 2012) and curved text lines (Seytre et al. 2019).

With Strabo, Chiang and Knoblock (2014) have proven the capabilities of their system to robustly and quite accurately deal with aforementioned combined issues effectively to automatically extract the text layer from certain maps and localise individual text labels within. We employ Strabo’s text label detection command-line tool in our experiments without any tuning of parameters or additional training.

The resulting oriented bounding boxes of found text labels in a map are then rotated to be in horizontal orientation and fed to a text-recognition system as individual pages.

We use tesseract[5] with its python wrapper[6], a well-established open source OCR engine, with a multitude of readily-available pre-trained language models. It is necessary to specify the language to be detected, because tesseracts training makes character detection context sensitive. We use the available pre-trained legacy models, because the newer long short-term memory model does not allow whitelisting of characters, which we deem necessary to avoid erroneous detection of diacritics and special characters in the clutter of overlapping noisy map features. Specifically, we used the pre-trained language model for German and English language without any further additional training and no dictionary for subsequent correction of OCR results.

It is expected to obtain OCR results with only limited quality on maps (Chiang et al. 2016) and it is hardly possible to get accurate transcription without intense manual intervention. But our goal is not to fully digitise the map content: we only require a small number of correctly identified text labels, with a character recognition precision high enough to successfully match these words to a dictionary of place names.

Geocoding

Myers et al. (1996) have already shown the possibility of using gazetteers as dictionaries to be able to find and verify text strings in noisy maps, which can be hard for OCR to correctly recognise with a mere bottom-up method. In place of a dictionary of all place names of potential regions our map might be located in, which can easily contain millions of entries (ca. 2.3 million domestic toponyms are listed by the United States Board on Geographic Names[7]), we employ the web request APIs of OpenStreetMap[8] (OSM) or GeoNames[9] for matching and, where possible, localisation of detected place names. Geographical coordinates of identified toponyms obtained by these geocoding services can later be used for automatic georeferencing.

Text labels of feature designations are usually placed somewhat arbitrarily in the vicinity of a point feature (such as a city on large scale maps, or special buildings on small scale maps), along a linear feature (streets or rivers) or within a region (district, geographical area such as landscapes or forest areas or land uses like parks). This makes text labels an inherently imprecise indicator of location. Averaging displacements and disturbances of these location hypotheses over multiple candidate points is required to obtain a good match for the general location of the map.

Generally speaking, different features can be utilised on different scales: small scale maps (1:25.000 and smaller) show street names or point features (churches, plazas), which can lead to very high accuracy, whereas for large scale maps we have to make do with more vague feature designations and rely on averaging out of label displacements and unclear centre points, e.g. of districts. In turn, geodatabases usually store the geographical extent of these larger features, which can serve as a measure of uncertainty for localisation.

There are some special challenges connected to the matching of historical places and toponyms to modern-day databases: some old villages may have been incorporated into cities and only remain as street names. Local place names of regions, like the common name of some valley may not exist

[5] https://github.com/tesseract-ocr
[6] https://github.com/sirfz/tesserocr
[7] https://usgs.gov/geonames/domestic-names
[8] https://nominatim.openstreetmap.org
[9] https://www.geonames.org/export/geonames-search.html

as OSM features but may be discovered in street names or an old inn. Sometimes, on the other hand, both the region and a more modern feature exist distinctly but quite close to each other in modern maps. Just as well, former region names might stem from old rivers or brooks, and are now found at multiple different positions along the remaining river run, sometimes many kilometers away. All of these cases lead to multiple hypotheses of locations generated for each detected toponym in a historical map, which will then have to be filtered for the best fitting solution.

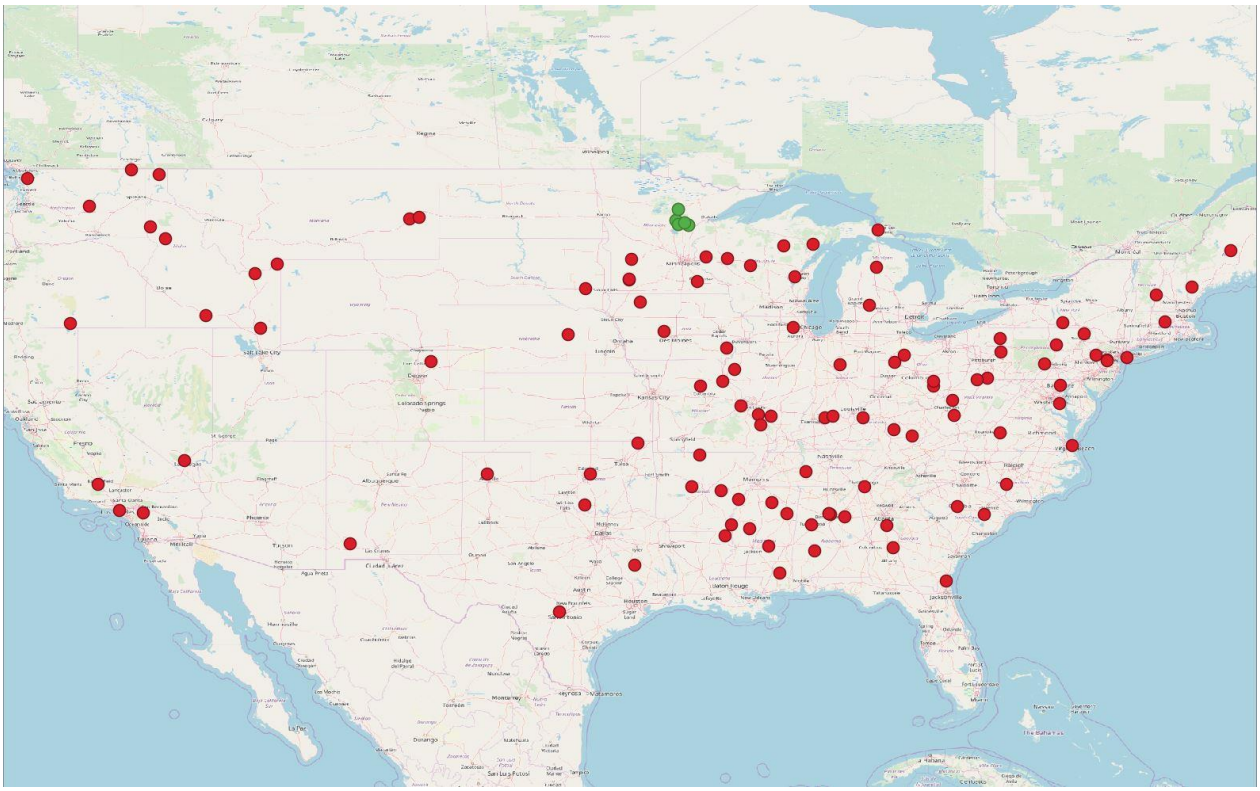


Figure 1. Red: all hypotheses for recognised place names. Green: successful density-based clustering. Background map: OpenStreetMap

Control Point Validation

Especially when recognised designations in a map are not complete because of OCR errors and designations are split over multiple text labels with a commonly used pre- or suffix (e.g. “new…”, “...county”), the place name queries become ambiguous and lead to localisations in far-apart places, even different countries. We can’t rely on the ordering of results, usually by size or importance, of geocoding web services either to help us in correctly identifying, for example, a city with a commonplace name. To illustrate, *Figure 1* shows in red all alternative hypotheses of place locations after geocoding toponyms recognized in a USGS map of Aitkin, Minnesota. In our test dataset of US maps this effect was stronger than in the German maps, possibly stemming from a stronger historical diversification of dialects and thereby toponyms in Europe.

Since even a small number of far off outliers can prohibit successful rectification of a map, we have to establish the correct subset of found hypotheses for further use. We can assume spatial

proximity of "correct" localisations (that is coordinates actually representing the specific place in the map), suggesting the use of clustering schemes. Considering the number of outliers can well be in the order of magnitude or even exceed the number of inliers, this leads to inadequate results from common methods like maximum likelihood estimation of Gaussian parameters or random sample consensus.

To overcome this problem, we employ a clustering scheme devised by Ester et al. (1996): density-based spatial clustering of applications with noise (DBSCAN). DBSCAN groups points by determining local neighbourhoods. All points which have a minimum amount of neighbouring points within a specified distance, will be put in a cluster with said neighbours. By the transitive property, a cluster can extend to neighbours of neighbours to include remote points on the border of the resulting spatial cluster. In our case, location hypotheses obtained by the previous geocoding step are clustered with a minimum distance set to a value smaller than the maximal extent of maps in the dataset, which can be inferred or estimated from the scale or sheet size, which are generally known a priori. As a result, we choose the most promising cluster by the greatest cardinality and use its constituents as GCPs for the following rectification step. The five remaining green dots in *Figure 1* show the effect of said clustering and a successful removal of all outliers.

Transformation

When the map to be transformed uses a known projection and is planimetrically correct, a first-order polynomial or affine transformation should be sufficient for rectification. For this we need at least three valid GCPs. With most maps from the 20th century we don't expect any major distortions within a map sheet and a low order polynomial transformation should give reasonable results. For maps with unusual projections or very large scale, a higher order transformation can help to warp the rectangular map to it's correct geographical extents, in turn requiring more GCPs.

Distorted maps need even more complex transformations (e.g. rubbersheeting) which in turn require a lot of accurately placed and evenly distributed GCPs otherwise errors can be amplified in sparsely covered areas.

For our uses, we want a method that can handle some single erroneously predicted locations without losing accuracy at correctly placed GCPs, which lead us to run our experiments with the *thin plate spline* (TPS) transformation.

This more flexible transformation methods is in fact not necessary for the datasets we use, because they are already more or less in the correct orientation and projection, but we use them to test whether our approach can work in general cases and its robustness to misplaced GCPs.

Evaluation

Dataset

We explored the suitability and performance of our approach on the following selection of survey maps: Firstly, German survey maps of Hamburg in 1:25.000 from 2002–2004 (TK25_20) in German language latin script including all-capitals, italics and left-leaning italics. They have a palette of three to four colours (black text, buildings, biome pattern symbols, roads, green woodland, blue bodies of water and corresponding name text, brown topography isolines and

corresponding height labels. The black text includes districts, hamlets, special land uses, named regions (swamps, state forests, etc.) and some major roads. Another set of maps of overlapping area, but from 1946 to 1959 and previously binarised into low-resolution black-and-white images (TK25_19). Two additional similar colour scanned maps in 1:50.000 from 1909 and 1915 were used which only use black colour (TK50).

For comparison we process a selection of USGS historical topographic maps from the years 1953–1985. They use latin script in the English language including all-capitals fonts and have a palette of five colours (red grid lines with corresponding designations, brown topography isolines, green vegetation, blue bodies of water and corresponding text, black administrative boundaries, text, symbols (i.e. mines), roads). The black text designates hamlets, county border markings, townships, land use (state forests, wildlife preserves, etc.), special point features (lookout towers, airstrips, gravel pits, etc.). Since USGS maps covering rural areas exhibit less clutter, we arbitrarily selected a handful of maps from the American north, mostly Minnesota. We found maps in 1:100.000 scale (USGS100_MN) to give the best results, since smaller scales don't contain many place names for many rural areas of the United States. Conversely, the bigger 1:250.000 (USGS250) scale engenders a lot of clutter and overlapping features with comparatively small font sizes.

Some maps use the margins of the map to indicate neighbouring places which lie just outside of the map area, often hinting at where a certain road leads. This writing on the margins, showing little clutter in the background, is happily recognised by the OCR engine, which can lead to significant distortion (see *Figure 2*). The same happens with other writing outside of the map area, such as the map title, issuing authority, or other meta-information. Since there is no easy remedy to this without knowing the exact map extents, we remove the map margins from the test images, in line with a future application of seamless stitching of georeferenced survey map collections.

Figure 2. Misplaced GCP because of a recognized place name on the map margin. Transformed Map: TK50 Langenhorn (1909) Background map: OpenStreetMap

Results

Table 1 shows the results for each processing step for the previously described maps. The second column lists the number of detected labels recovered from Strabo for each sheet in the map. The third column shows the number of labels which were recognised by OCR well enough to return results when querying OSM. Next we list the number of all alternative location hypotheses returned by OSM. In the fourth column the number of GCPs are listed which remain after removing outliers with DBSCAN. Finally, we show from which of these maps a georeferenced output image could be generated.

In total, about 38% of all maps were successfully georeferenced. For some maps enough correct GCPs were placed, but some incorrect ones led to so much distortion, a valid transformation solution could not be determined. Surprisingly, the oldest maps (TK50) delivered the best results (*Figure 5*). Even after 100 years of intense urban development, enough toponyms could be correctly identified to accomplish quite accurate georeferencing. Only a small inset map piece had to be cut out and could not be correctly localised itself.

Table 1. Results comparison for each dataset

Dataset (# of sheets)	**# labels**	**# readable**	**# after geocoding**	**# after clustering**	**successfully georeferenced**
TK25_19 (13)	77, 80, 104, 140, 122, 91, 181, 121, 156, 239, 107, 176, 136	4, 6, 5, 8, 3, 6, 19, 5, 12, 14, 5, 2, 5	19, 15, 17, 20, 16, 15, 97, 12, 46, 58, 14, 9, 23,	0, 0, 2, 5, 0, 0, 6, 0, 2, 5, 0, 0, 0	no, no, no, yes, no, no, yes, no, no, yes, no, no, no
TK25_20 (6)	189, 224, 225, 406, 265, 211	25, 13, 7, 19, 18, 24	79, 37, 23, 53, 54, 112	12, 4, 0, 8, 10, 2	yes, yes, no, yes, yes, no
TK50 (3)	83, 70, 117	7, 6, 9	19, 9, 29	3, 3, 2	yes, yes, no
USGS100_MN (4)	119, 82, 133, 46	10, 3, 23, 8	58, 21, 137, 44	0, 0, 5, 0	no, no, yes, no
USGS250 (3)	320, 177, 77	36, 13, 5	180, 83, 22	22, 4, 0	yes, no, no

A good fraction of TK25_20 maps were successfully localised as well. Because of split labels at more complex descriptions like “highway to Oldesloe” or “belonging to Rellingen” some wrongly attributed toponyms lead to distortions. With USGS maps the biggest issues was detecting township or county labels, whereas the smaller settlement labels, often smudged together because of low resolution and colour bleed, could not be recognised and thus geocoding returned wrong locations.

OCR

Experiments show that the label extraction regularly produces too small bounding boxes for the text labels, often cutting half of the first and/or last character of a word (*Figure 3*, left). This seems to happen predominantly with italic fonts. Conversely, sometimes we see too large, ill-fitting bounding boxes with multiple words from different labels, which Chiang et al. (2016) attribute to clutter from non-text symbols in the vicinity of correctly detected text (*Figure 3*, right).

A related, albeit expected, problem is that sometimes designations are split over multiple labels, because of wide spaces between words or a split over multiple lines, which makes it hard to combine them again to the correct place names. When words are split in this way their matches from gazetteers become ambiguous, but the spatial clustering proved to be very helpful in correctly verifying most of these cases.

Overall though, text label detection worked surprisingly well with most of the maps in spite of varying graphical quality. A much greater hindrance is the actual character recognition on the detected labels. This is a strong indicator for inability of the OCR system to deal with map fonts and overlapping features without at least further training. Common OCR engines seem to be trained exclusively on modern typeset fonts and struggle with the writing on maps. As a result we can determine out-of-the-box OCR software does not work satisfactorily on many "technical" stenciled fonts, such as in maps, and will probably be completely unusable on handwritten labels. This is evident from the evaluation of Strabo (Chiang and Knoblock 2014), with a reported 60% to 91% character recall on computer-generated map images from Yahoo and Google respectively, whereas we observe only limited results of far less than 50% recall in our test set of scanned historical maps.

But even with further training on the required fonts, OCR is very sensitive to clutter in figure-colour, especially when overlapping features are touching the characters. This is also the reason why we don't try to detect umlaute and other diacritics, as they are hard to distinguish from the granular scanning noise and some pattern symbols (*Figure 3*, middle). Not detecting diacritical letters is acceptable, since most geocoding services use a fuzzy string matching which works quite well when only supplied with the basic glyph without diacritics. Still, OCR is very sensitive to letter outline defects, so morphological operations to reduce noise and clutter extremely harm recognition precision and should be used with care. In summary, OCR proved to be the biggest bottleneck for successful exploitation of map labels because of difficulties with map-fonts as well as clutter leading to badly recognised characters and words.

Figure 3. Deficiently detected text labels. Taken from map: TK25 Wandsbek (2004)

Registration

While DBSCAN-filtering successfully removes outliers which stem from geocoded location matches of places with similar names, there are still other cases that, while being technically correct matches, still provide erroneous GCPs. Especially in USGS maps the label detection and OCR recognise the names of townships at their centre or of counties along their respective boundary. For the maps in 1:250.000 these boundaries can be quite long and thus have been labelled multiple times. When querying these names in geocoding services, they usually return the name of the capital city which gave the name to the respective township or county. This will lead to ground control points, which are more or less in the right area but can distort the output georeferenced map image. Compare *Figure 4*, where OCR recognised township names in a 1:100.000 USGS map. The spatial clustering alone can't help with this, instead some semantic handling to deal with these ambiguous designations will have to be investigated in the future.

We settled on mostly using thin plate spline transformation for rectifying map images, because polynomial transformations can shift the entire image to accommodate for minor displacements in the GCPs, which quickly leads to ill fits even at regions around correctly localised points. TPS

transformations on the other hand lead to distorted map areas around misplaced GCPs, which might make the map unusable for further applications but is better suitable for understanding the source of distortions. *Figure 4* shows the most problematic property of TPS: when an outlier is on the convex hull of the GCPs, it can quickly shift a big region of the image, even when all other points are correct. Here the top most point stems from the label of a township close to the edge of map, far distant from the actual township center, which is on the next map sheet.

The rectification performance is too bad for actual further use of the maps, but the georeferenced localisation of the map performs actually quite well considering this experimental approach under high uncertainty. To generally handle slightly imprecise GCPs, we just have too few successful localisations. Maybe a dictionary containing generally used terms, such as “marsh”, “lakes”, “forest” or “county”, could be used to remove some of the partially detected designations, that lead to imprecise localisations. Generally, we need better results from OCR to use contextual information or allow averaging of errors as well as further experiments with transformation methods to elaborate on each methods ability to deal with outliers.

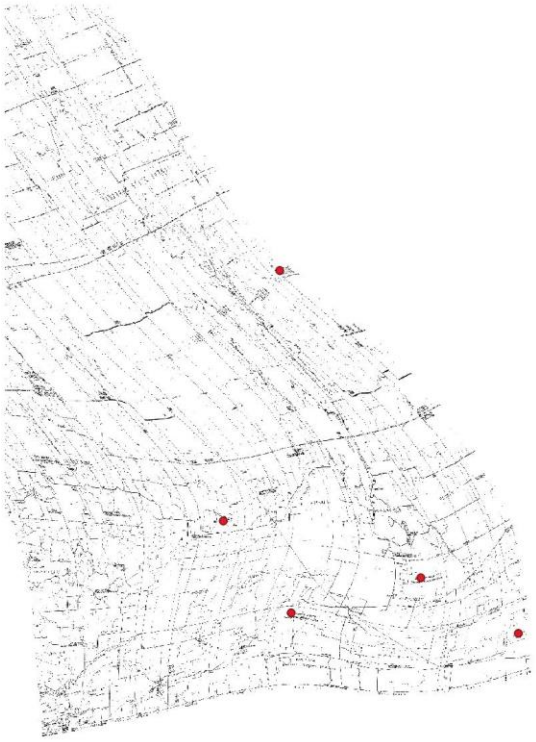


Figure 4. Misplaced GCPs because of geocoding equating township names and settlement names. Map: USGS Aitkin (1994).

Figure 5. Two successfully georeferenced map sheets with used GCPs (red) and ground-truth (blue) Maps: TK50 Langenhorn and Hamburg (1909, 1915)

Future work

The first step for both better recall in label detections as well as improved precision of OCR should be to enhance text-graphics separation, e.g. by symbol and line recognition (Chiang et al. 2006, Heitzler et al. 2018, Velázquez and Levachkine 2003).

Still, for the special use case of historical maps most OCR engines won't work out of the box. We will have to evaluate specialised OCR models or train our own, to obtain sufficient accuracy on stenciled or even handwritten map text. Further training of the Strabo text detector might improve label detection for scanned maps as well.

An additional processing step is called for, using information about differently sized text to infer context information and hierarchy of labels. For example, this could help to determine whether some place names refers to a settlement, county, region or other feature type. Through this better defined search queries can be used while geocoding, to quickly converge on more precise location hypotheses. The same goes for the recurring ambiguous partial place names, resulting from text labels being split over multiple lines or because of wide character spacing. Connecting split text labels (Lin et al. 2018) should greatly facilitate place name matching.

Another problem of geocoding is that the used web searches are very sensitive to single missing or incorrect letters in short queries, despite being able to correct some spelling mistakes. A better fuzzy string matching, that is more lenient to incorrect OCR results could help here, but might require a local copy of the place name database.

The positioning and size of a label alone can already create large geographical displacements. For large scale maps, where not many other features might be present that we can use to obtain more fine-grained alignment, a necessity to improve the accuracy will be to connect the labels to their respective positional markers, if present, as already explored by Budig et al. (2016).

When enough GCPs are left over from the geocoding step we are provided with an over defined solution for finding a transformation function. This can allow us to remove remaining outliers by minimising root-mean-square deviation residuals to increase accuracy from misplaced text labels and wrongly attributed toponyms.

Finally, the use of text labels alone for localisation of GCPS will probably not be precise enough for competing with manual georeferencing. Indeed we only intend it to be a first step of restricting the search space of candidate place matches and a coarse alignment and then further increasing the accuracy by using vectorised features for fine adjustments and stitching together multiple map sheets by other methods as proposed by Briggs and Li (2006) and Chen et al. (2004).

References

Briggs, R. and Li, Y. (2006). Automated Georeferencing Based on Topological Point Pattern Matching, Proceedings of AutoCarto.

Budig, B., van Dijk, T. C., Wolff, A. (2016). Matching Labels and Markers in Historical Maps: an Algorithm with Interactive Postprocessing. ACM Transactions on Spatial Algorithms and Systems (TSAS) 2(7).

Chen, I.-W., Chen, H.-R., Tseng, Y.-H. (2016). Automatic image matching and georeferencing of digitized historical aerial photographs. 37th Asian Conference on Remote Sensing (ACRS), 2: 1100–1108.

Chen, C.-C., Knoblock, C. A., Shahabi, C., Chiang, Y.-Y., Thakkar, S. (2004). Automatically and accurately conflating orthoimagery and street maps. Proceedings of the 12th Annual ACM International Workshop on Geographic Information Systems (ACM-GIS), 47-56.

Chiang, Y.-Y., Chiang, Y.-Y., Knoblock, C. A. (2006). Classification of Line and Character Pixels on Raster Maps Using Discrete Cosine Transformation Coefficients and Support Vector Machine. 18th International Conference on Pattern Recognition (ICPR), 2: 1034–1037.

Chiang, Y.-Y., Knoblock, C. A. (2014). Recognizing text in raster maps. GeoInformatica 19: 1–27.

Chiang, Y.-Y., Leyk, S., Nazari, N. H., Moghaddam, S., Tan, X. T. (2016). Assessing the impact of graphical quality on automatic text recognition in digital maps, Computers & Geosciences 93: 21–35.

Cléry, I., Pierrot-Deseilligny, M., Vallet, B. (2014). Automatic Georeferencing of a Heritage of old analog aerial Photographs. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences II–3: 33–40.

Ester, M., Kriegel, H.-P., Sander, J., Xu, X. (1996). A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD): 226–231.

Heitzler, M., Gkonos, C., Tsorlini, A., Hurni, L. (2018). A modular process to improve the georeferencing of the Siegfried map. e-Perimetron, 13(2): 85–100.

Koffka, K. (1935). Principles of gestalt psychology. London: Kegan Paul, Trench, Trubner And Co.

Landesamt für Vermessung und Geoinformation Schleswig-Holstein (2004). Topographische Karte 1:25.000 (TK25), Blatt Wandsbek, Hamburg.

Lin, H., Chiang, Y.-Y. (2018). Automatic extraction of phrase-level map labels from historical maps. SIGSPATIAL Student Research Competition (SRC), 9(3): 14–15.

Myers, G. K., Mulgaonkar, P. G., Chen, C.-H., DeCurtins, J. L., Chen, E. (1996). Verification-based approach for automated text and feature extraction from raster-scanned maps. Graphics Recognition Methods and Applications 1072: 190–203.

OpenStreetMap contributors (2020). OpenStreetMap. https://www.openstreetmap.org

Roy, P. P., Pal, U., Lladós, J. (2012). Text line extraction in graphical documents using background and foreground information. International Journal on Document Analysis and Recognition (IJDAR) 15(3): 227–241.

Schröder, J. (2013). Georeferenzierung und Entzerrung der brandenburgischen Kartenblätter des Schmettauschen Kartenwerks. Vermessung Brandenburg 2: 14–23.

Seytre, J., Wu, J., Achille, A. (2019). TextTubes for Detecting Curved Text in the Wild. ArXiv:1912.08990 [Cs].

United States Geological Survey (1994). 30x60 minute series (topographic), 1:100.000, sheet Aitkin, Minnesota.

Velázquez, A., & Levachkine, S. (2003). Text/Graphics Separation and Recognition in Raster-Scanned Color Cartographic Maps. Lecture Notes in Computer Science 3088: 63–74.

Vermessungsbüro der Bau-Deputation Hamburg (1909). Topographische Karte 1:50.000 (TK50), Blatt Langenhorn, Hamburg. Amended 1921.

Vermessungsbüro der Bau-Deputation Hamburg (1915). Topographische Karte 1:50.000 (TK50), Blatt Hamburg und Amt Ritzebüttel, Hamburg.

Wolter, D., Blank, D., & Henrich, A. (2017). Georeferencing River Networks Using Spatial Reasoning. *Proceedings of the 11th Workshop on Geographic Information Retrieval (GIR).*

Zhou, X., Yao, C., Wen, H., Wang, Y., Zhou, S., He, W., Liang, J. (2017). EAST: An Efficient and Accurate Scene Text Detector. *ArXiv:1704.03155 [Cs]*.

Geoff Groom[1], Gregor Levin[1], Stig Svenningsen[2], Mads Linnet Perner[2]

Historical Maps – Machine learning helps us over the map vectorisation crux

Keywords: Denmark, topographic maps, land category, OBIA, CNN, map symbols.

Summary: Modern geography is massively digital with respect to both map data production and map data analysis. When we consider historical maps, as a key resource for historical geography studies, the situation is different. There are many historical maps available as hardcopy, some of which are scanned to raster data. However, relatively few historical maps are truly digital, as machine-readable geo-data layers. The Danish "Høje Målebordsblade" (HMB) map set, comprising approximately 1100 sheets, national coverage (i.e. Denmark 1864-1920), and geometrically correct, topographic, 1:20,000, surveyed between 1842 and 1899, is a case in point. Having the HMB maps as vector geo-data has a high priority for Danish historical landscape, environmental and cultural studies. We present progress made, during 2019, in forming vector geo-data of key land categories (water bodies, wetland, forest, heath, sand dune) from scanned HMB printed map sheets. The focus here is on the role in that work of machine learning methods, specifically the deep learning tool convolutional neural networks (CNN) to map occurrences of specific map symbols associated with the target land categories. Demonstration is made of how machine learning is applied in conjunction with pixel and object based analyses, and not merely in isolation. Thereby, the strengths of machine learning are utilised, and the weaknesses of the applied machine learning are acknowledged and worked with. Symbols detected by machine learning serve as guidance for appropriate values to apply in pixel based image data thresholding. The resulting map products for two study areas (450 and 300 km^2) have overall false-positive and false-negative levels of around 10% for all target categories. The ability to utilise the cartographic symbols of the HMB maps enabled production of higher quality vector geo-data of the target land categories than would otherwise have been possible. That these methods are in this work developed and applied via a commercial software (Trimble eCognition©) recognizes the significance of a tried-and-tested and easy-to-use, graphical-user-interface and a fast, versatile processing architecture for development of new, complex digital solutions. The components of the resulting workflow are, in principle, alternatively usable via various free and open source software environments.

Introduction

Increasingly, the conversion of paper or image media historical maps to machine readable digital geo-data layers is moving from time and resource demanding visual interpretation of the maps and manual digitisation to less time-consuming automated production methods. In a Danish context, analyses of historical maps have most often been elaborated only for a limited number of map sheets and relatively small study areas (e.g. Fritzbøger and Caspersen 2002; Jensen and Jensen 1979; Kristensen et al. 2009; Svenningsen et al. 2015) due to a reliance on visual, manual methods. There are, however, exceptions, such as Dam (2005), who, for all of Denmark, manually produced geographical information from late 18th century maps.

[1] Aarhus University, Denmark [gbg@bios.au.dk]
[2] Royal Danish Library, Denmark

In Denmark there is a unique opportunity to develop and apply digital map processing methods for automated extraction of land categories from historical maps as all publicly produced topographical maps are made freely available as high-resolution scans by the national mapping agency (SDFE). An automated extraction of land categories from historical topographic maps will provide the opportunity to generate countrywide map layers, representing the location and extent of different land categories at different points in time. Such map layers can be applied in spatially detailed assessments of land use and land cover (LULC) changes. Being spatially explicit, the map layers can be used in assessments at any spatial level, from a specific location or plot to different spatial delineations, such as parishes, municipalities, watersheds or the whole country.

The topographical maps 'Høje målebords' (HMB) represent the first large-scale topographic mapping of Danish territory. The HMB map series consists of about 1100 sheets at a scale of 1:20,000 surveyed between 1842 and 1899 and was developed by the Danish military as a response to a growing need for detailed topographical information about the landscape for military purposes.

Digital map processing is a relatively young research area, which grew out of disciplines such as graphics recognition and document analysis (Chiang et al. 2016, Freeman and Pieroni 1980, Liu et al. 2019). Due to various graphical quality issues and the general complexity of the map contents, extraction of information from maps is generally more difficult compared to processing of other data sources, such as written documents or technical drawings (Lladós et al. 2002). In maps, layers of geographic features, such as roads, contour lines, text labels, and land categories, often overlap with each other, increasing the degree of content complexity and colour mixing. Furthermore, due to inconsistency of applied symbols and colours, processing of manually drawn maps is even more challenging (Leyk and Boesh 2009). General techniques for graphics recognition and document analysis can therefore not be directly applied to map processing. In most historical maps, colours, symbols, linear features and text items, represent explicit, overlapping layers of geographic information. Colours often represent thematic information of land categories, such as forest or water. Symbols, and to some degree text items describe different types of geographic characteristics. Linear features can represent geographic features, such as roads or watercourses, boundaries of land categories, administrative boundaries or topographical characteristics, such as contour lines. A major objective of digital map processing is to spatially isolate these overlapping layers of geographic information from each other and to filter out those map elements that obstruct the extraction of the geographical layer of interest.

Through 2019, resources were available to enable the authors to examine the possibilities of developing, undertaking and evaluating automated machine-readable geo-data layer production from HMB digital image sources, for five target land categories: water bodies, wind blown sand (dunes), wetland (mainly meadow), heath and forest (Levin et al. 2020), (*Figure 1*). The applied methodology was one of object-orientated image analysis (OBIA), largely reliant on expert knowledge derived rule based image segmentation and object labelling operations, rather than statistical classification methods. OBIA methods of that form are widely used in production of geo-data layers from image data as they enable high levels of control over mapping results (Blaschke et al. 2014, Chen and Weng 2018, Chen et al. 2018). For the mapping of water bodies, forest, heath and wetland the HMB colouration represents the major basis for formation of image objects. The core target class object mapping process thereby involves identification of the relevant image layers, followed by selection of an appropriate value to apply in threshold based image

object segmentation (Levin et al. 2020). However, the colour variability between the different map sheets (*Figure 2*) means that if constants (fixed values) are applied for the thresholding, results are generally poor across the set of map sheets. The HMB maps include the use of distinct symbols that associate with the target classes wetland and heath (also forest). Machine learning methods such as CNN are making increasingly marked in-roads into many areas of applied image work, often with impressive results (Ball et al. 2017). Occurrences of map symbols can be detected by application of the machine learning methods like CNN (Quan et al. 2018). In the HMB maps target class symbols lie within the zones of target class colouration. It was therefore postulated that CNN predictions of symbol locations can form a basis for determination of the appropriate threshold for good target class image segmentation. The aim of this paper is to present details of the CNN symbol prediction assisted thresholding approach, with a basic evaluation of its effectiveness.

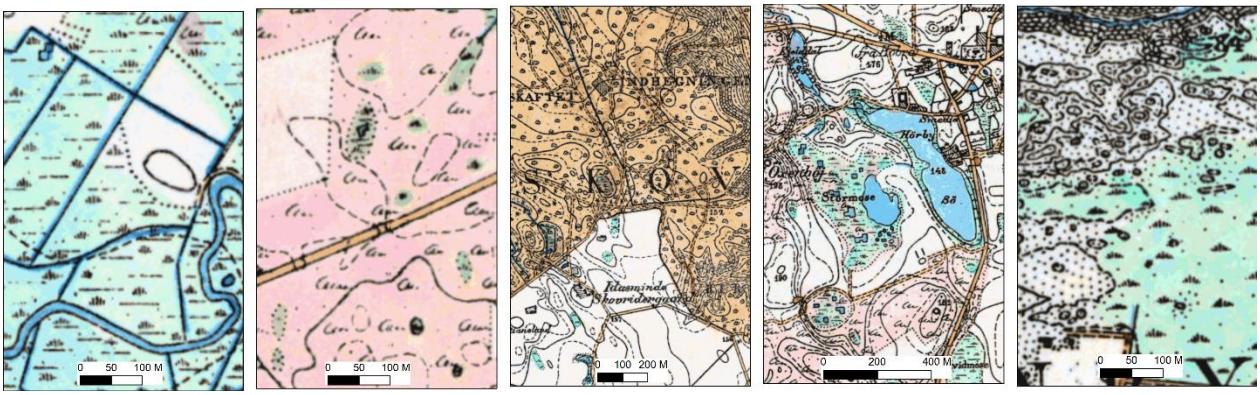


Figure 1. Examples from the HMB map series, that includes areas having the HMB representation of the HMB legend categories (left-right), wetland (blue-green), heath (pink), forest (brown), water bodies (blue) and dunes (even black stipple).

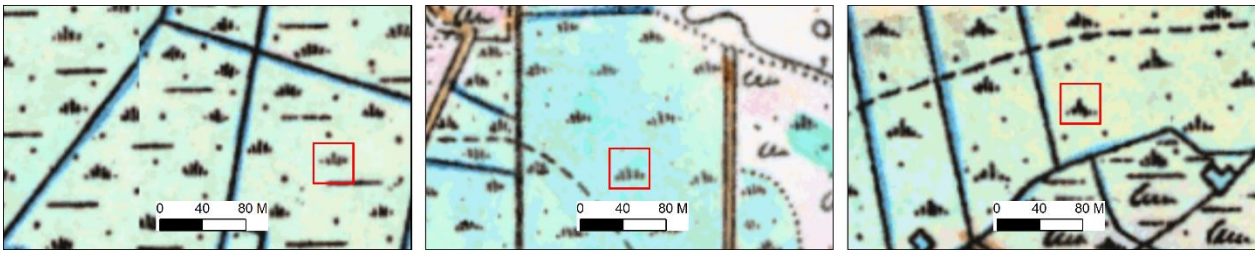


Figure 2. Examples from the HMB map series, showing variations in the colouration applied for the HMB legend categories wetland (blue-green); each image is displayed without display stretching. Also illustrated in these examples is the variability in the form of the wetland symbol (examples indicated by added red box)

Materials

The work reported here uses two study areas, selected to (a) enable use of the results of the developed information production method as an indication of their more general applicability and (b) to enable analyses of the resulting land category mappings for a meaningful set of parish extents. The first of these objectives implied choice of study areas that between them included extensive areas of all the target land categories and source map data with differences typical of

the range found across the full HMB map series. One study area is formed of the bounding-box of the parishes of Klejtrup, Nørre Onsild, Sønder Onsild, Lindum, Øls and Hvornum, in eastern Jutland. That study area, of approximately 195 km2, includes extensive areas of forest, wetland and heather. The second study area is formed of the bounding-box of the parishes of Hirtshals, Horne and Asdal, in northern Jutland. That study area, of approximately 90 km2, includes extensive areas of heath, wetland and dune. Because the HMB sheets do not follow parish boundaries, nine HMB sheets were needed to completely cover the selected parishes in the study area in eastern Jutland, while six HMB sheet were needed to for the northern study area. Therefore data was extracted from all of these HMB sheets, thus creating an extended study area, covering approximately 422 km2 and 13 parishes for the study area in eastern Jutland and approximately 200 km2 and seven parishes for the study area in northern Jutland (*Figure 3*).

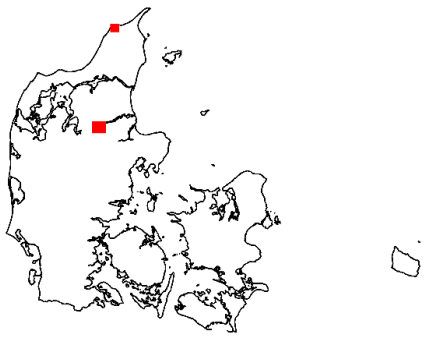


Figure 3. Locations of the two study areas.

There are two forms of HMB source material that the project that work reported here is part of, has used. One is from the digital archive of the Danish National Mapping Agency (SDFE) of scannings made of one set of HMB paper maps. These source data are publically available via SDFE web portal:
https://download.kortforsyningen.dk/content/dtkh%C3%B8je-m%C3%A5lebordsblade
as georeferenced (ETRS_1989_UTM_Zone_32N, resampled to a pixel size of 2 m) image data that have been mosaicked and then cut into a set of 20 x 20 km tiles as .tif (8-bit unsigned, LZW lossless compressed) files. The other source is the archive of HMB paper sheets held by the Danish Royal Library (KB) and are the original paper sheets, including margins. Use of items from this archive in this work data required scanning and geo-referencing, which was applied to just the map sheets needed in order to cover the project's study areas. Scanning of the KB sourced map sheets was done using a ScannTECH 600i-fb, with the standard settings of that scanner for colour artwork. A digitisation resolution of 600 dpi was used, which is appropriate for 1:20,000 scale source material (Tobler, 1988). The KB archive paper map sheets and the map sheets that were used for the SDFE scanning are not entirely the same raw material. Firstly, only approximately two-thirds of the sheets in the KB collection are full-colour with the remainder (mainly eastern Denmark) being just 3-colour (black, blue, brown), whereas the entire set that were scanned for the SDFE dataset are full colour. Where maps of both sources are full-colour, the actual colourations differ, with evidence that they relate to two sets of paper maps that were coloured independently, rather than just differential effects of time and storage conditions. The differences between the data from the two sources present interpretation challenges in a few places

but, countering that, the situation is one of beneficial complementarity. Therefore, wherever possible and meaningful, the thresholding is made on an image layer from both image sets.

Methods

Numerous raster image processing methods can be applied to enhance the image data with respect to one or other aspect of the colouration. In general, initial gentle kernel-based filtering is useful to smooth out single pixel anomalies and is applied in this work. Applied enhancements are Hue-Saturation-Intensity, histogram normalisation and arithmetic combinations of the R-G-B (in raw-smooted and in histogram normalised form) image data. Visual inspection indicates which derived image layer(s) are most likely to provide a satisfactory initial set of image objects for a target class. In general, the same derived image layers provides a good basis for class discrimination for each map sheet. For example, the relative greenness (G/(R+B)) based on the histogram normalised R-G-B layers are generally a potent means for mapping the wetland target class (*Figure 4*).

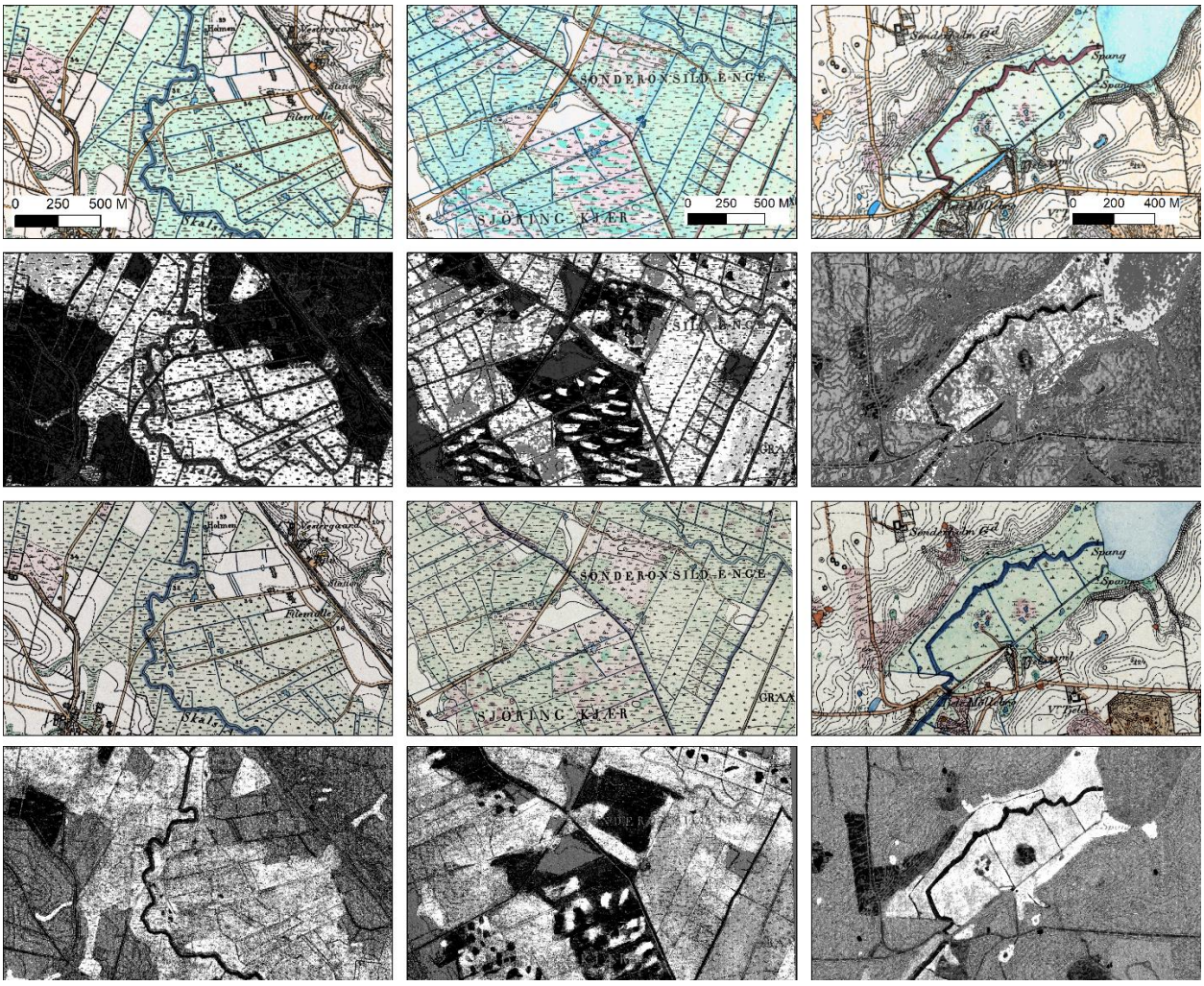


Figure 4. Three examples of the HMB data for areas with the legend class wetland, together (rows 2 and 4) with their representation as the derived image layer RGB-histogramNormalised-relativeGreeness; rows (1) and (2) are of data from the SDFE archive and rows (3) and (4) are of data from the KB map sheet collection

Simple threshold based image segmentation is used rather than alternative OBIA segmentation methods, such as multi-resolution segmentation (Baatz and Schäpe 2000), as the former involves fewer parameters and provides a high level of user control over the resulting objects.

An effective symbol CNN model, for an HMB symbol, is acquired by testing of sets of symbol target and non-target training data and CNN parameter settings for an independent set of HMB images. For a specific map sheet, the use of a CNN model based symbol mapping is enabled as simple thresholding of the symbol CNN heatmap to give a set of image objects, followed-by filtering-out of objects that did not meet size and shape expectations for the symbol. Remaining symbol objects are then grown to include a zone of the surrounding image data. This set of objects is then filtered with respect to a relatively broad range of expected object means in one or more image layers (*Figure 5*). Most of the remaining symbol associated objects should then relate to areas of the target class. These objects are then used to derive the threshold that would be most appropriate to apply for initial target class image objects across the current image data. The appropriate threshold is determined through stepwise threshold lowering with testing for under-mapping and over-mapping with respect to the grown symbol objects. This iterative process also flags cases where an image layer cannot be used to effectively map a target class. The derived threshold value is used as a baseline value to apply in various subsequent threshold based segmentation and filtering operations in order to derive a full mapping of the target class.

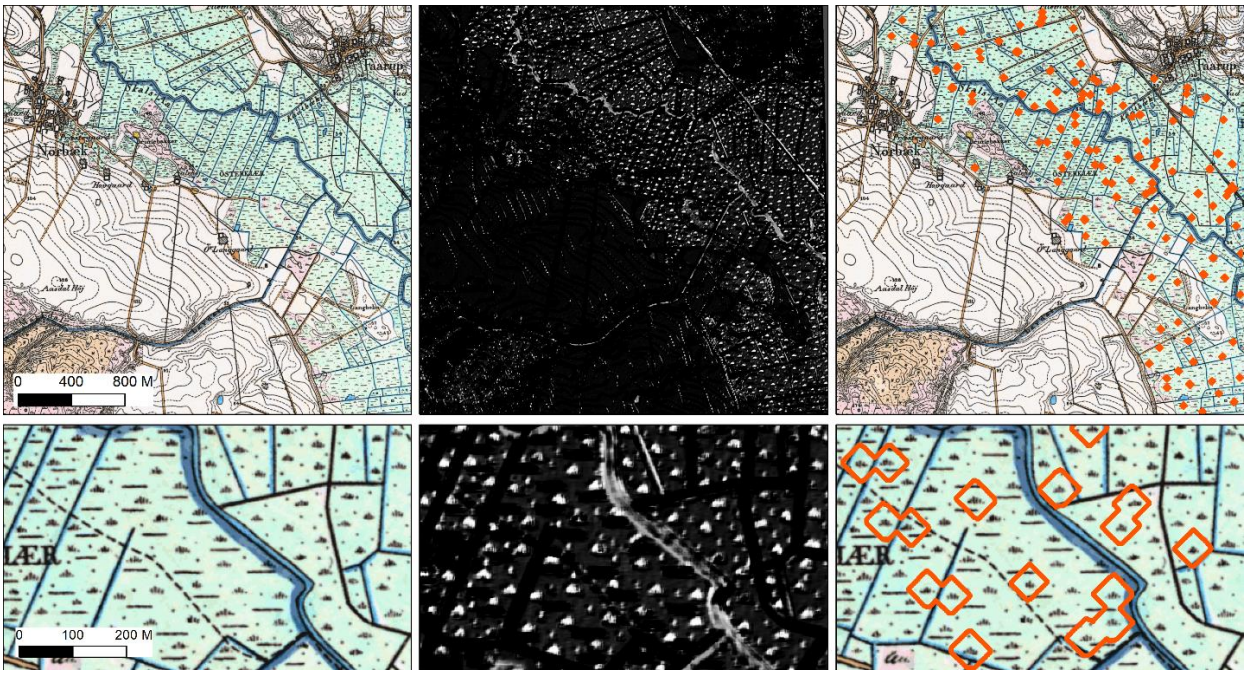


Figure 5. One of the examples from Figure 4 with the corresponding wetland symbol raw CNN heatmap (centre) and symbol heatmap hotspot objects, grown and filtered to zone objects (right).

Results

The accuracy of the vector target class map layers produced by the entire workflow, including various stages of morphological and contextual refinement of the initial image objects (Levin et al 2020), have been assessed against a 100 m grid set of reference point data, approximately 29,000 points in total. These reference data have been derived by independent HMB image visual interpretation of the target class membership of each point. The final results of that accuracy assessment will be reported in Levin et al. (2020). An earlier accuracy assessment of initial target class mappings indicated false-negative and false-positive levels of around 10% for most classes on most map sheets. Those detected patterns of errors were applied for further development of the workflow.

Here, as an assessment of just the contribution of CNN symbol mapping to the adaptive threshold setting, a simple visual assessment is made between thresholding with a pair (one each for SDFE, KB) of constants for three map sheets, and thresholding with pairs of variables, derived via the symbol CNN symbol predictions (*Figure 6*). For the left map sheet the constant based thresholding is nearly as effective as the CNN assisted variable thresholding, but for the other two map sheets it gives a clearly far inferior result.

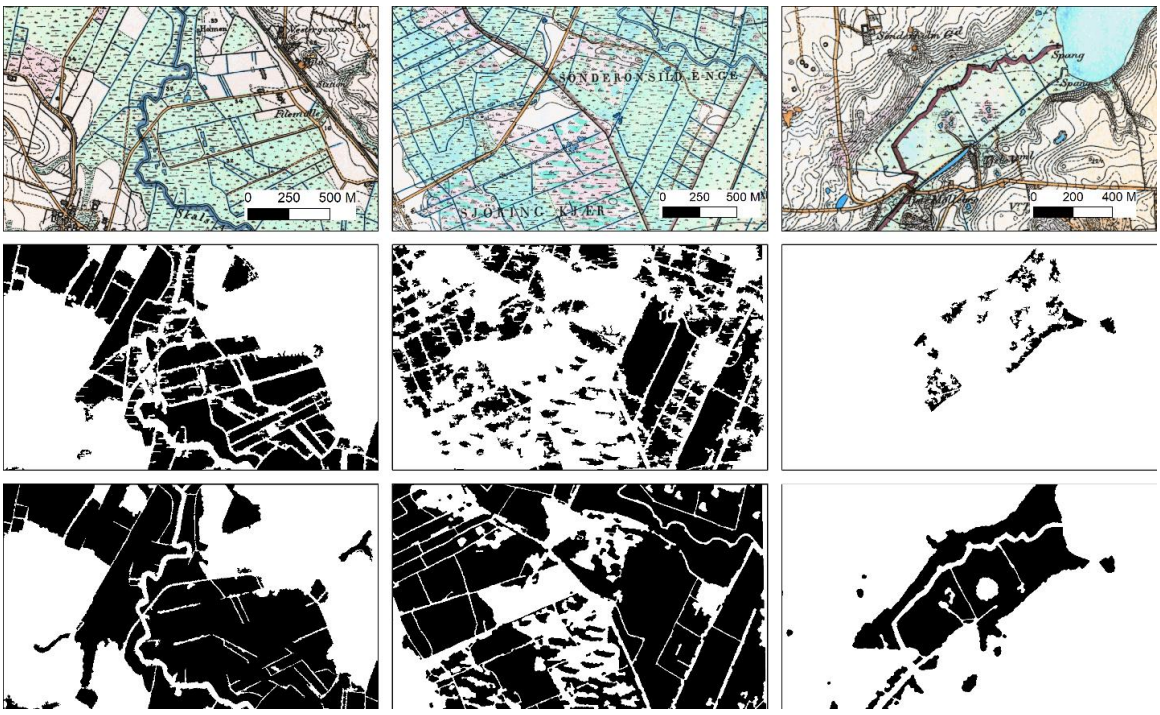


Figure 6. Result mapping of the target class wetland for three HMB map sheets (top row) with a constant threshold values for all three map sheets (middle row) and variable thresholds set via use of CNN symbol predictions (bottom row).

Discussion

Work with historical maps, including their conversion to machine-readable geospatial information, should not be viewed as just a technical, computing task, but also one that requires understanding of the historical context of the source materials and thereby how their contents should be

understood. The HMB maps are the primary source for detailed information about the landscape before the 20th century. However, despite the importance of the maps as a source for information about the landscape of the past, the surveys and production behind the maps have only to a limited degree been subject to research. Although the military origin of the maps and the possible implications for landscape analysis have been recognized (Brandt, 2004; Münier, 2009), our understanding of the land classification system is still limited to what is stated in the map legends (Korsgaard, 2006). This means that information about the categorisation and representation of land area types and landscape elements in the map is sparse and in principle limited to the signature tables published with the map series. Although some of the land area types shown in the legend seem to be directly comparable to modern LULC classification typologies, such as forest vegetation, recent work shows that classification of land area types has changed between different map types and over time (Svenningsen et al., 2015). These aspects of the HMB maps are thoroughly discussed by Levin et al. (2020). However, those considerations do not have a decisive impact upon the general principle of the technique for improved colouration segmentation described here.

The presented approach to the mapping of the heath and wetland target classes from the HMB data represents a robust integration of modern machine learning methods in a broader image analysis workflow. Therefore, the overall workflow is not reliant on CNN method alone to map the target classes, and indeed, does not require that the CNN method is highly efficient at mapping the target class symbol, since filters are applied to the initial CNN symbol mapping result. Therefore the approach might be regarded as "soft machine learning".

There are at least two further ways that the CNN symbol mapping can contribute, in a soft way, to an image object based workflow, such as the one developed and applied in this work. (1) Actual target class extent that are otherwise unmapped as target class objects (i.e. false negative errors) but having strong indication by the symbol's CNN heatmap of symbol presence, can form seeds for additional target class image objects. (2) Wholesale absence of symbol CNN heatmap symbol indications from larger target class image objects can indicate cases of over-mapping (false positive errors).

The approach presented in this paper is dependent upon there being a sufficient number (at least around 10) of target class symbols present and detected by the CNN model. If the target class is only minimally present in the image, this condition may not be met. That situation is flagged for in the workflow, requiring then use of a manually determined threshold for that map sheet. Pre- and post-hoc visual examination of map sheets and the resulting maps for anomalous conditions is part of the full workflow.

Whilst machine learning methods such as CNN are making increasingly marked in-roads into many areas of applied image work, often with impressive results (Ball et al. 2017), it is still prudent to develop hybrid approaches that also utilize other image analysis methods. Most forms of machine learning are still dependent on training data, which can represent a manual, time consuming component of the application. In principle, these methods can also self-learn as they are applied to more and more image examples. However, production of perfect CNN models should not become a goal for its own sake, unless it can be objectively expected that they will deliver the required mapping in an almost perfect form. A machine learning solution for production of perfect geospatial machine-readable information from historical maps, including extensive

spatial features, is yet to be demonstrated, and it is therefore prudent to acknowledge the limitations of current machine learning and develop the degree to which machine learning can assist other methods.

The methods described here have been developed and applied within a commercial software (Trimble eCognition ©, https://geospatial.trimble.com/products-and-solutions/ecognition). However, all of the steps could just as well be applied outside of that software, via various free and open-source programming languages and programming environments. The advantages of working within a commercial software are that the architecture of the component operations is efficiently programmed for fast processing, the different components are well integrated and the software's graphical user interface enables fast development of new ideas. The disadvantage, other than the cost, is that the developer is limited to the set of and implementation of operations as they have been programmed within the software, some parts of which may be just barely documented.

Conclusion

A method has been described and demonstrated whereby basic CNN machine learning map symbol detection results assist image object methods to make improved mapping of map legend areal units associated with distinct map colourations and associated map symbols.

References

Baatz, M. and A. Schäpe (2000). Multiresolution Segmentation – an optimization approach for high quality multi-scale image segmentation. In: J. Strobl, T. Blaschke and G. Griesebner (eds.) *Angewandte Geographische Informationsverarbeitung XII*. Heidelberg: Wichmann-Verlag, 12-23.

Ball, J.A., D. Anderson and C.S. Chan (2017). Comprehensive survey of deep learning in remote sensing: theories, tools, and challenges for the community. *Journal of Applied Remote Sensing* 11(4): 042609

Blaschke, T., G. J. Hay, M. Kelly, S. Lang, P. Hofmann, E. Addink, R., Queiroz Feitose, R., F. van der Meer, H. van der Werff, F. van Coillie and D. Tiede (2014). Geographic Object-Based Image Analysis – Towards a new paradigm. *ISPRS Journal of Photogrammetry and Remote Sensing* 87(0): 180-191.

Brandt, J. (2004) Anvendelse af topografiske kort til monitering af biotoper. *Geoforum Perspektiv* 3(5): 53–61.

Fritzbøger, B. and O. H. Caspersen (2002) Long Term Landscape Dynamics - a 300-Years, Case Study from Denmark. *Danish Journal of Geography* Special Issue 3: 13–27

Chen, G. and Q. Weng (2018). Special issue: Remote sensing of our changing landscapes with Geographic Object-based Image Analysis (GEOBIA). *GIScience & Remote Sensing* 55(2): 155-158.

Chen, G., Q. Weng, G. J. Hay, and Y. He (2018). Geographic object-based image analysis (GEOBIA): emerging trends and future opportunities. *GIScience & Remote Sensing* 55(2): 159-182.

Chiang, Y.-Y., S. Leyk, N. H. Nazari, S. Moghaddam and T.X. Tan (2016) Assessing the impact of graphical quality on automatic text recognition in digital maps. *Computers & Geosciences* 93, 21–35.

Dam, P. (2005). Videnskabernes Selskabs kort 1768-1805 - En introduktion til såvel analog som digital brug. *HisKIS' Digitale årsskrift* 2003-2005: 36–44

Korsgaard, P. (2006). *Kort som kilde : en håndbog om historiske kort og deres anvendelse*. Vejle (DK): Dansk Historisk Fællesråd : Sammenslutningen af Lokalarkiver.

Jensen, R. H. and K.M. Jensen (1979). Kulturlandskabet i Borris og Sdr. Felding — En kortbladsanalyse af et Vestjysk landbrugssamfund og en dokumentation for dets udvikling. *Danish Journal of Geography* 78-79: 61–99.

Kristensen, S.B.P., A. Reenberg and J. J. D. Peña (2009) Exploring Local Rural Lands-cape Changes in Denmark: A Human-Environmental Timeline Perspective. *Danish Journal of Geography* 109 (1): 47–67.

Levin, G., G. B. Groom, S. R. Svenningsen and M. Perner (In prep) *Automated production of spatial datasets for land categories from historical maps - Method development and results for a pilot study of Danish late-1800s topographical maps*. Aarhus (DK): Aarhus University, DCE - Danish Centre for Environment and Energy.

Leyk, S. and R. Boesch (2009) Extracting Composite Cartographic Area Features in Low-Quality Maps, *Cartography and Geographic Information Science*, 36: 71-79.

Liu, T., P. Xu and S. Zhang (2019) A review of recent advances in scanned topographic map processing. *Neurocomputing* 328: 5–87.

Lladós, J, E. Valveny, G. Sanchez and E. Marti (2002) Symbol recognition: Current advances and perspectives. In: D. Blostein and Y.-B. Kwon (eds.) *Graphics Recognition Algorithms and Applications* (Lecture Notes in Computer Science, Vol. 2390). Berlin: Springer, 104–128.

Münier, B., 2009. Landskabets udvikling siden midten af 1800-tallet. In: B. Odgaard and J. R. Rømer (eds.), *Danske Landbrugslandskaber Gennem 2000 År: Fra Digevoldinger Til Støtteordninger*. Aarhus (DK): Aarhus Universitetsforlag, 81–98.

Quan, Y., Y. Shi, Q. Miao, and Y. Qi (2018). A Combinatorial Solution to Point Symbol Recognition. *Sensors* 18(10): E3403.

Svenningsen, S.R., G. Levin and M. R. Jepsen (2015) Decrease in Danish semi-natural grassland – a social construct or a real-world change? *Danish Journal of Geography*, 115 (2): 157-166.

Tobler, W. (1988). Resolution, Resampling, and All That. In: H. Mounsey and R. Tomlinson (eds.), *Building Data Bases for Global Science*, London: Taylor and Francis, 129-137.

AUTOMATIC VECTORISATION OF HISTORICAL MAPS
International workshop organized by
the ICA Commission on Cartographic Heritage into the Digital

March, 2020
Department of Cartography and Geoinformatics
ELTE Eötvös Loránd University
Budapest, Hungary