Automation of Digital Historical Map Analyses

Tenzing W. Shaw*[1a] and Peter Bajcsy[a]
[a]National Center for Supercomputing Applications, University of Illinois at Urbana-Champaign

ABSTRACT

This paper addresses the problem of automating analyses of historical maps. The problem is motivated by the lack of accuracy and consistency in the current comparison process of geographical objects found in historical maps by visual inspections. The objective of our work is to compare shape characteristics of the Great Lakes region in a dataset of approximately 40 French and British historical maps created in the 17th through the 19th centuries. Our approach decomposes the visual inspection into steps such as object segmentation, spatial scale calibration, extraction of calibrated object descriptors and comparison of descriptors over time and multiple cartographer houses. The automation of object segmentation is achieved by template shape-based segmentation using the Hu moments as shape descriptors and ball-based region growing. The automation of spatial calibration is accomplished by detection and classification of lines along map borders and by mapping striped boundaries intersected by latitude and longitude lines into degrees of arc length. Thus, shape characteristics of segmentation results in pixels can be converted to geographical units, for example, an area of a lake in square miles. We report experimental evaluations of automation accuracy based on comparison with manual segmentation results, as well as the knowledge obtained from the area comparisons.

Keywords: Image Analysis, Segmentation, Map Scale Estimation, Historical Maps

1. INTRODUCTION

1.1 Problem Statement

An important question facing historians is how knowledge of different geographic regions varied between nations and over time. A specific example of this question is how to characterize differences in geographic knowledge of the Great Lakes Region possessed by the French and British from the 16th to the 19th century. This can be accomplished by examining French and British historical maps from different points in this time period, and judging the accuracy of these maps relative to modern geographic knowledge. Previously, this would have been done qualitatively, through visual inspection. The objective of this effort is to develop computer algorithms designed to automate map comparison and quantify the results of this comparison. The algorithmic workflow consists of template shape-based segmentation of geographic objects of interest followed by spatial calibration and shape analysis of the segmented objects as illustrated in Figure 1. Specifically, we have partially automated the processing of 38 (21 British and 17 French) digitized historical maps of the Great Lakes region. In each case, the algorithm segments each lake from cropped images, and computes its surface area in square miles. The differences between these calculated areas and the modern figures can then be taken as a measure of the accuracy of regional geographic knowledge at the time when each historical map was made.

* twshaw3@ncsa.illinois.edu; phone (217) 265-5387; fax (217) 244-7396; isda.ncsa.illinois.edu

Computer Vision and Image Analysis of Art II, edited by David G. Stork, Jim Coddington, Anna Bentkowska-Kafel,
Proc. of SPIE Vol. 7869, 78690A · © 2011 SPIE · CCC code: 0277-786X/11/$18 · doi: 10.1117/12.872875

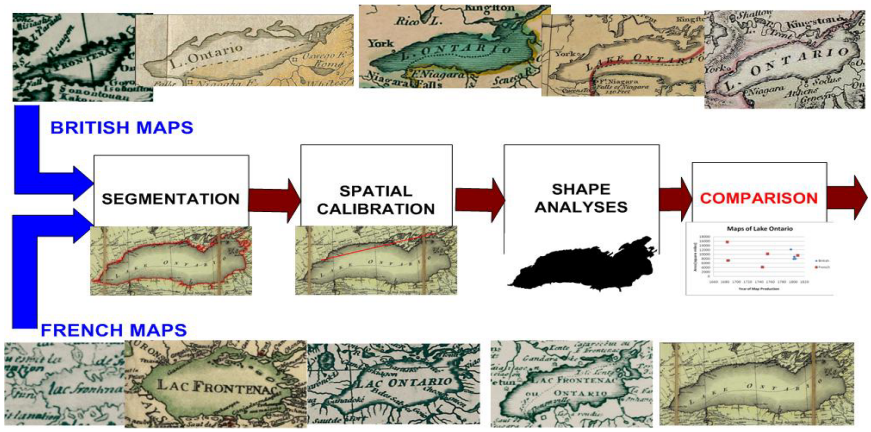

Figure 1. The algorithmic workflow for extracting cartographic information from historical maps.

1.2 Template Shape-Based Segmentation

We have developed a segmentation algorithm for extracting regions whose shape is most similar to that of a given example. The algorithm uses ball-based region-growing segmentation combined with the seven Hu moments[1] to evaluate shape similarity. The ball-based segmentation places a circular region into a seed location and grows the region subject to color homogeneity and spatial contiguity constraints. Each resulting region is described by the Hu moments and compared to the Hu moments of a given example. The algorithm searches over a space of parameters including the region growing criteria and seed placement.

1.3 Scale Detection

In order to compute extensive geographic quantities, such as lake area or shore length, it is necessary to determine the scale of the map under consideration, so that a conversion can be obtained between pixels and a physical unit such as degrees or miles. The common calibration technique is to use two points of known distance apart and manually estimate the scale. However, this technique is hard to automate because it assumes the presence of two known points and the consistency of their map presentation needed by the detection algorithm. We have employed another calibration technique based on examining the dashed neatline which is present around the border of most of the maps under consideration and the consistency of the neatline dashes with respect to latitude or longitude lines. This map calibration technique seems to be more suitable for automation.

2. SEGMENTATION

2.1 Overview

We have developed a segmentation algorithm for extracting regions whose shape is most similar to that of a given example. The algorithm uses ball-based region-growing segmentation combined with the seven Hu moments to evaluate shape similarity. The ball-based segmentation places a circular region into a seed location and grows the region subject to color homogeneity and spatial contiguity constraints. Each resulting region is described by the Hu moments and compared to the Hu moments of a given example. The algorithm searches over a space of parameters including the region growing criteria (intensity threshold and ball size) and seed placement, attempting to find the closest result (using the Euclidean distance metric in the space of Hu moments) to the example. The algorithmic workflow is illustrated in Figure 2.

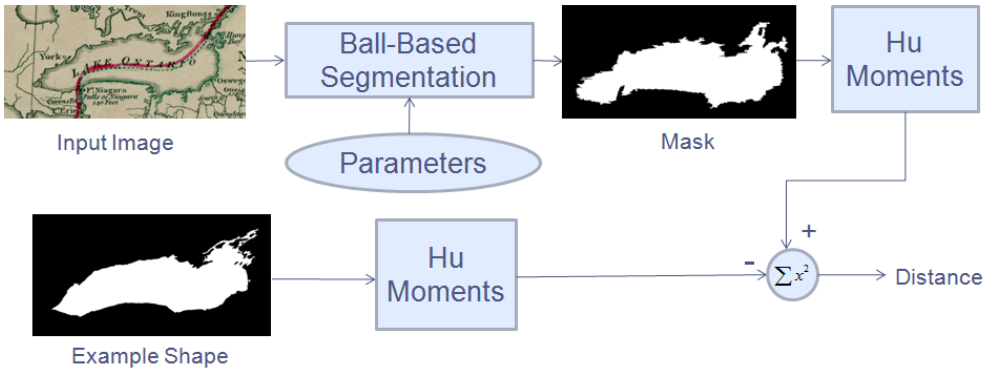

Figure 2. Segmentation algorithmic workflow

Note that since the objects of interest are assumed to be simply-connected, and initial segmentation results are not in general (due to noise, background clutter or text written over regions), we perform a step of post-processing similar to morphological closing[3], in which the ball-based region growing algorithm is applied to the initial result with a large ball-size and a seed location outside of the object, thereby eliminating "holes" in the shape. In order to improve the computational performance of the segmentation, we have used the method proposed by Yang and Algregtsen for speeding up calculation of the Hu moments[2].

2.2 Illustration of Parameter Search

To illustrate the method by which the segmentation algorithm chooses the best parameters, consider Figure 3 below, in which the target image is a modern map of Lake Ontario, and the example shape is derived from a different modern map. The six lower images represent each point in the parameter search at which a new minimum distance has been found (relevant parameters are given below each result). We observe that the algorithm is able to reject the initial incomplete results, and adjust the intensity threshold to include the brighter region in the northeast of the lake. In this case, the optimal threshold was found to be 50 while the optimal ball size was found to be 2.

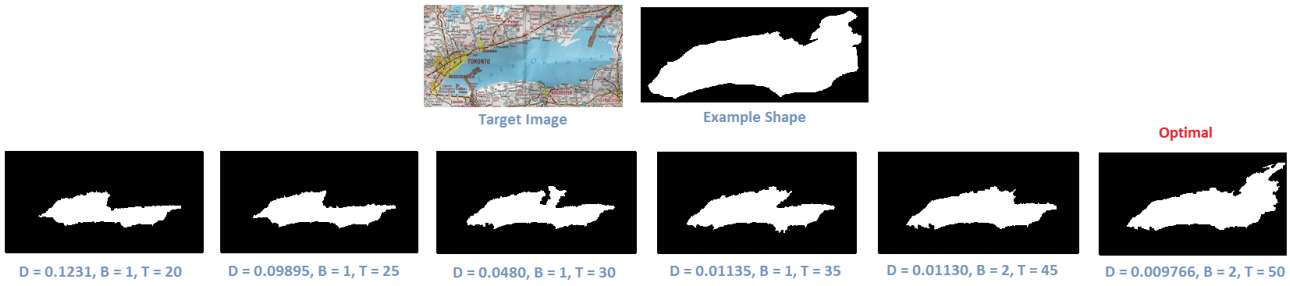

Figure 3. Illustration of search over parameters of underlying ball-based segmentation algorithm: D denotes the distance from the example in Hu moment space, B denotes the ball size, and T denotes the threshold.

2.3 Using Morphological Filtering to Improve Segmentation Results

A major challenge in segmenting geographic objects using a region-growing approach is the presence of clutter in the form of latitude and longitude lines, boundary following hash patterns, or text. This clutter can sometimes form a barrier to region growing, as is the case in the image of Lake Michigan at left in Figure 4. In this example, the seed is placed in the southern part of the lake, and a latitude line forms a barrier which prevents the region from growing into the northern part of the lake. Using morphological closing with a vertically oriented 5x1 structuring element, the latitude line can be removed, allowing the region-growing algorithm to segment the entire lake. Similarly, a 1x5 structuring element can be used to eliminate the longitude line blocking the narrow strip of lake in the southeast. This technique has been used as a pre-processing step to improve segmentation results.

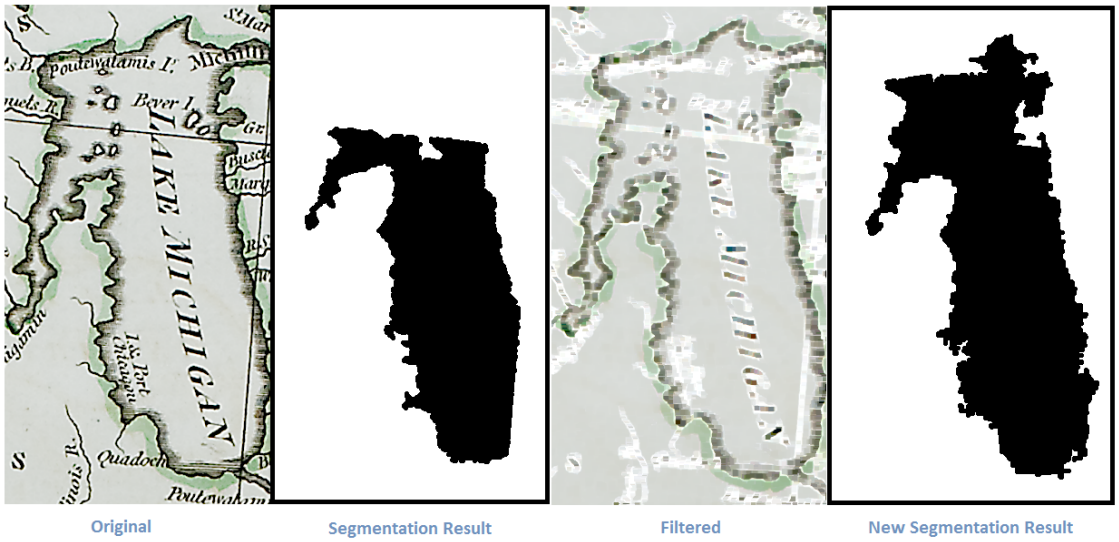

Figure 4. Removal of latitude and longitude lines using morphological closing with 5x1 and 1x5 structuring elements, respectively: the original image and corresponding segmentation result are shown at left while the filtered image and improved segmentation result (using the same parameters) are shown at right.

3. SCALE DETECTION

3.1 Manual Calibration

To manually extract the scale of a map, one can select two points on the map such that the distance between them is known, and compute the ratio of this distance to the distance in pixels obtained from the digital image. An example is shown in Figure 5, where the north-south extent of the Chesapeake Bay is used for calibration.

Our automatic scale detection algorithm works by examining the dashed neatline around the border of a map, and counting the number of dashes between successive intersections with latitude or longitude lines (see Figure 6). Due to mapmaking conventions, this number corresponds uniquely to the desired scale. Neatline analysis can be broken down into several constituent steps including boundary selection (to choose which side of the map to analyze), line detection, line classification (to distinguish the dashed neatline from parallel solid lines), dash-length calculation (to find the length in pixels of a single stripe of the dashed line), and transversal detection (to find intersections with latitude or longitude lines). The ratio of the distance in pixels between successive intersections to the length of a stripe in pixels will then produce the desired number. We will examine each of the steps in detail below.

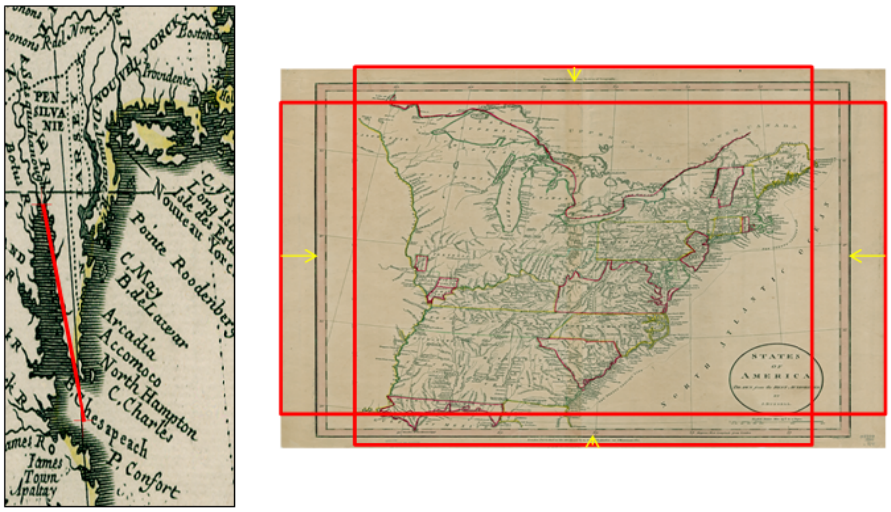

Figure 5. Manual calibration using Chesapeake Bay (left), boundary extraction (right)

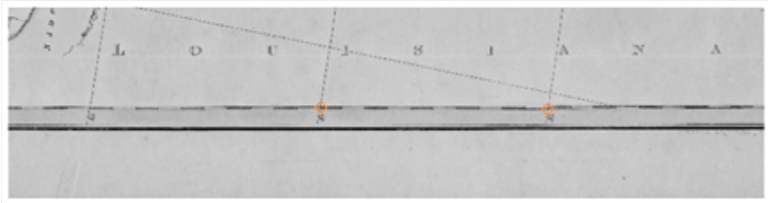

Figure 6. Intersections (circled in orange) of the dashed neatline with successive longitude lines

3.2 Automatic: Boundary Extraction and Selection

Before attempting to find the neatline, it is useful to extract the four map boundaries in order to shrink the search space and reduce clutter. In order to do this, simple intensity thresholding is used to locate the end of the uniform map background (see the yellow arrows in Figure 5). The four boundaries are then cropped (see red lines) so that the end of the background is centered in each region. Since only one boundary need be considered in the final analysis, the next step is to select the boundary with the least clutter. This can be accomplished by summing intensities perpendicular to each boundary, and taking the variance of the resulting signal. Since this signal would ideally be almost constant (each boundary should have roughly uniform cross-sections along its length), the boundary yielding the lowest variance is selected.

3.3 Automatic: Line Detection

Once a boundary has been selected and segmented, the next task is to locate the dashed neatline within this boundary. This can be accomplished by summing along the length of the boundary in order to determine a signal such as that shown by the green line in Figure 7. Observe that the valleys in this signal correspond to the locations of the two vertical lines: the solid dark line and the dashed neatline. In order to detect these valleys, a moving average (red line) is used, with a moving threshold a fixed number of standard deviations below (blue line). Each contiguous region for which the signal is below the threshold is hypothesized to correspond to a line.

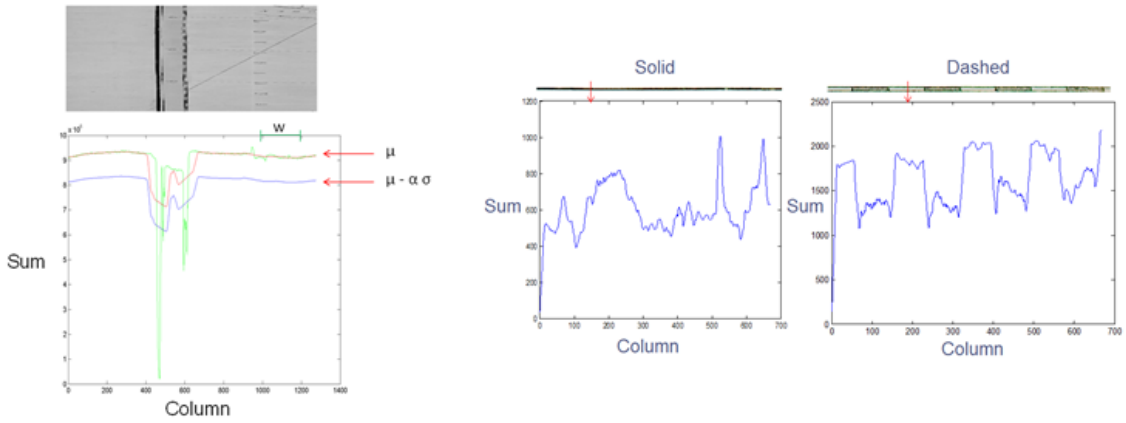

Figure 7. Line detection (left), solid vs. dashed line signals (right)

3.4 Automatic: Line Classification

Given the lines found in the previous step, the next task is to decide which one is the sought-after neatline. To do this, we designed a classifier to distinguish solid lines from dashed lines. The first step in the classification is to generate signals such as those shown in Figure 7. The signals represent sums of pixel intensities perpendicular to the lines (i.e. in the direction of the red arrows) as a function of position along the lines. Note that the signal corresponding to the dashed line resembles a noisy square-wave as one might expect, while the other signal has no discernable structure. In order to differentiate these two signals, we compute their autocovariances as shown in Figure 8. The energy (squared area) of the autocovariance from the origin to the first zero-crossing divided by the total energy provides a feature which has been found experimentally to provide good separation between the classes. From the plots in Figure 8, it is clear that this ratio should be much lower for dashed lines than for solid lines in general.

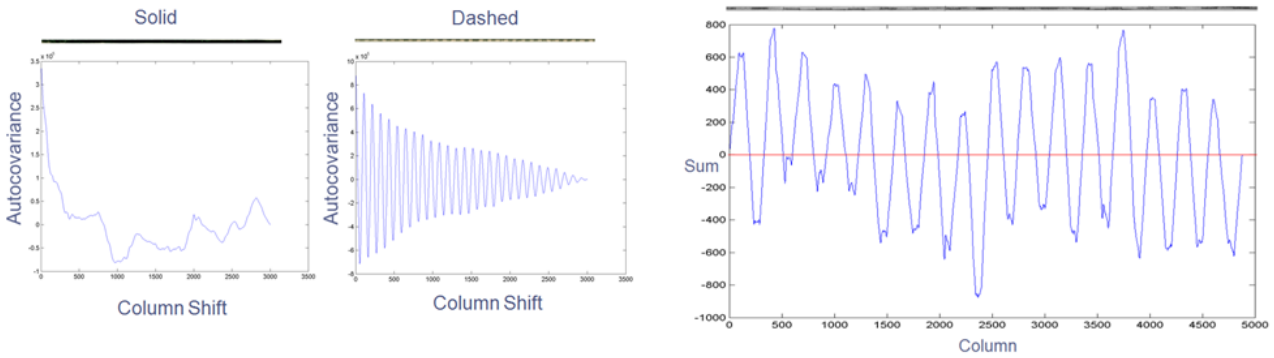

Figure 8. Autocovariance signals for classification (left), dash-length calculation (right)

3.5 Automatic: Dash Length Calculation

After the classification step, the neatline has been segmented from the image. One of the two quantities required for the final calculation is the number of pixels per dash. This can be determined by again generating a signal of sums perpendicular to the line and (after filtering to remove noise) thresholding the signal about its mean, as shown in Figure 8. Contiguous sequences of ones and zeros in the resulting binary signal are taken to be dashes, and their average length can be computed. This step can be refined by using the initial estimate of the dash length to define a window size, and then thresholding the signal again with a moving average using this window size.

3.6 Automatic: Transversal Detection

The second quantity required for the calculation of the final result is the distance in pixels between two successive intersections of latitude or longitude lines with the neatline. These intersections can be located by moving a point along the length of the neatline. For each position of the point, pixel intensities can be summed along lines with a range of angles passing through that point. If the current position of the point corresponds to an intersection, there will exist an angle such that the sum of intensities along a line at that angle is very low (i.e. when this line aligns with the latitude or longitude line involved in the intersection). By finding the minimum over all angles in the specified range of the line-sums of intensities as a function of position along the neatline, we get a signal such as that shown in Figure 9. Applying the same method as was used for line detection, we obtain the positions of the intersections, and hence the distances between them, which can be averaged to produce the desired result.

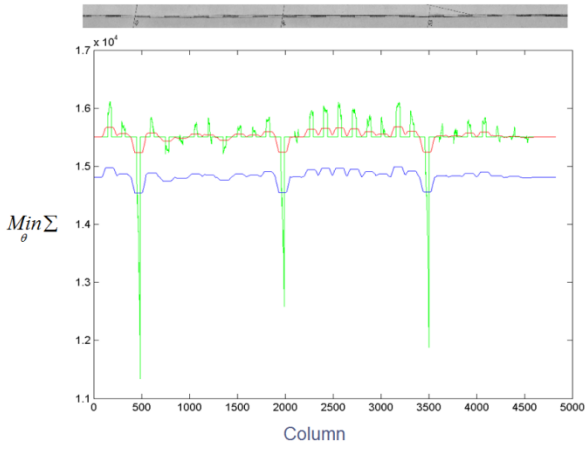

Figure 9. Signal for transversal detection

4. RESULTS

4.1 Segmentation Performance Evaluation

Segmentation performance can be evaluated through comparison with manually segmented masks. The MSE between the manual and automatic results is computed as illustrated in Figure 10, and is then normalized by the foreground area of the manual mask to produce a percent error.

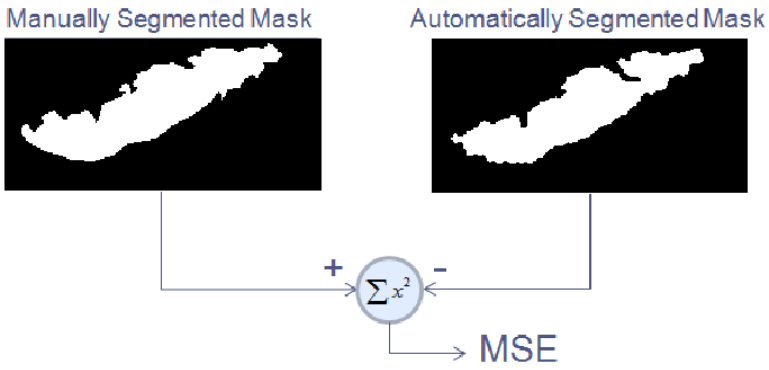

Figure 10. Segmentation accuracy evaluation by comparison with manual results

Once percent errors have been calculated for each instance, a segmentation result can be classified as successful if the corresponding percent error is less than 50%. This is the standard used to determine success in Table 1, which shows segmentation performance for each of the five lakes. As discussed in section 2.3, clutter such as text and latitude/longitude lines presented a significant challenge. Other obstacles included thick lake boundaries accompanied by hashing, as well as thin, faded boundaries in some maps.

Table 1. Segmentation results: MSEs are reported as percent errors relative to manually segmented masks.

Lake:	Erie	Huron	Michigan	Ontario	Superior
Number of Maps	36	34	38	37	37
Average MSE	73%	57%	59%	76%	50%
Number Successful	16	22	22	21	22
Average MSE for Successful Results	31%	34%	32%	35%	31%

4.2 Scale Detection Performance Evaluation

Map scale detection accuracy can be evaluated using the following standard: if the calculated number of dashes between two intersections rounds to the correct integer, then the algorithm has succeeded; otherwise, it has failed. Out of 25 maps which were used to test the scale detection algorithm, 16 (64%) were successful. Factors which caused the other maps to fail included severe clutter, which hampered transversal detection, and extremely low contrast, which hampered line detection. These challenges are illustrated in Figure 11.

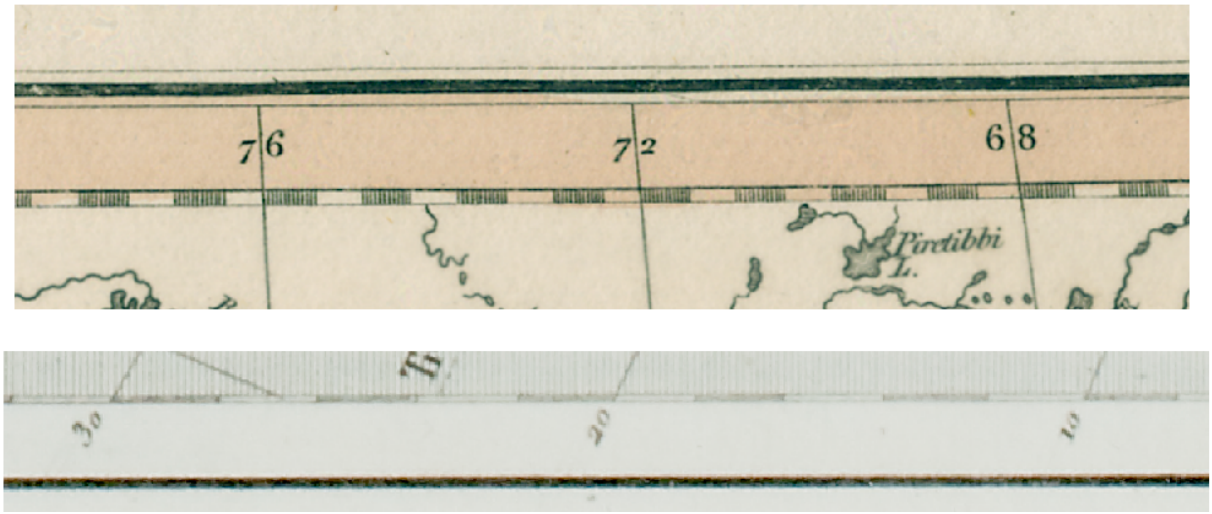

Figure 11. Clutter impinging on neatline: text, numbering, and cartographic objects (above), and low contrast (below)

4.3 Lake Area Results

Using the segmentation results which were classified as successful above, lake areas were calculated, and the errors of these calculated values relative to the modern estimates were averaged for each lake. Errors were calculated as the (absolute) difference between the calculated figure and the modern figure, as a percentage of the modern figure. The results are given in Table 2, along with other pertinent data such as mean creation dates (we expect maps created at later dates to be more accurate than earlier ones). Note that these results were generated using manual scale detection.

Table 2. Area calculation results

Lake:	Erie	Huron	Michigan	Ontario	Superior
Number of Maps (Overall)	16	22	22	21	22
Number of Maps (French)	9	10	11	9	10
Number of Maps (British)	7	12	11	12	12
Mean Creation Date (Overall)	1773	1769	1767	1768	1774
Mean Creation Date (French)	1762	1749	1746	1746	1758
Mean Creation Date (British)	1788	1785	1788	1785	1788
Actual Area (square miles)	9940	23010	22400	7540	49305
Average Percent Error (Overall)	99%	117%	70%	97%	83%
Average Percent Error (French)	102%	148%	97%	146%	144%
Average Percent Error (British)	94%	92%	44%	60%	32%

5. CONCLUSION

In automating extraction of cartographic information from images of historical maps, we had to overcome two automation obstacles: segmenting cartographic objects of interest and estimating map scale to report characteristics in physical units.

Figure 12 shows a plot of calculated areas of Lake Ontario vs. time, thereby giving an example of how the area data can be analyzed in order to compare British and French geographic knowledge over time. From this plot, we have inferred that although the French occupied the Great Lakes region sooner, French maps do not indicate any more accurate depictions of Lake Ontario than British maps. In the future, we plan to test the performance of the algorithms on a larger collection of historical maps.

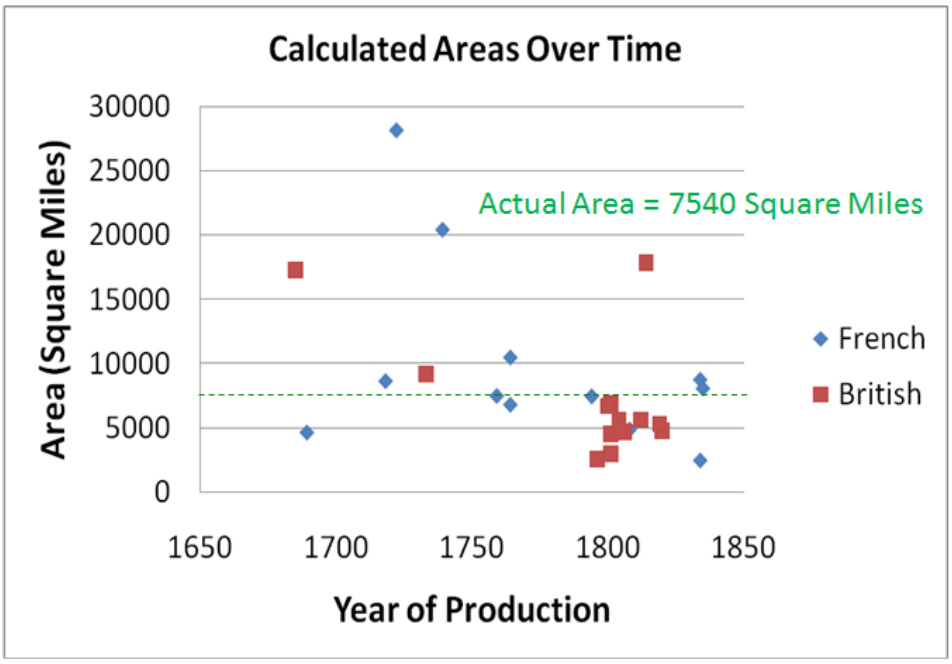

Figure 12. Calculated Areas of Lake Ontario in French and British Historical Maps over Time. Note that this graph was generated from a previous run of the algorithm with 14 British maps and 12 French maps.

6. ACKNOWLEDGEMENTS

This project has been funded by NCSA and NSF. We would like to acknowledge the NCSA Faculty Fellow program and the NSF ITS 09-10562 EAGER program for providing the funding. We would like to acknowledge the UIUC library and Betsy Kruger for providing the images of historical maps, and Michael Simeone and Robert Markley for consultations and insights about the map content, as well as help with finding the maps of interest.

REFERENCES

[1] M. K. Hu, “Visual pattern recognition by moment invariants,” IRE Transactions on Information Theory 8(2), (179-187) 1962.
[2] Yang, L. and Algregtsen, F., "Fast computation of invariant geometric moments: A new method giving correct results," Proc. IEEE Int. Conf. on Image Proc. 1, 201-204 (1994).
[3] Russ, John C., [The Image Processing Handbook, Fourth Edition], CRC Press, Boca Raton FL, 410 (2002).