Automatic Alignment of Geographic Features in Contemporary Vector Data and Historical Maps

Weiwei Duan
Computer Science Department
University of Southern California
weiweidu@usc.edu

Yao-Yi Chiang
Spatial Sciences Institute
University of Southern California
yaoyic@usc.edu

Craig A. Knoblock
Information Sciences Institute
University of Southern California
knoblock@isi.edu

Vinil Jain
Computer Science Department
University of Southern California
viniljai@usc.edu

Dan Feldman
Spatial Sciences Institute
University of Southern California
df_670@usc.edu

Johannes H. Uhl, Stefan Leyk
University of Colorado
Department of Geography
{johannes.uhl,stefan.leyk}@colorado.edu

ABSTRACT

With large amounts of digital map archives becoming available, the capability to automatically extracting information from historical maps is important for many domains that require long-term geographic data, such as understanding the development of the landscape and human activities. In the previous work, we built a system to automatically recognize geographic features in historical maps using Convolutional Neural Networks (CNN). Our system uses contemporary vector data to automatically label examples of the geographic feature of interest in historical maps as training samples for the CNN model. The alignment between the vector data and geographic features in maps controls if the system can generate representative training samples, which has a significant impact on recognition performance of the system. Due to the large number of training data that the CNN model needs and tens of thousands of maps needed to be processed in an archive, manually aligning the vector data to each map in an archive is not practical. In this paper, we present an algorithm that automatically aligns vector data with geographic features in historical maps. Existing alignment approaches focus on road features and imagery and are difficult to generalize for other geographic features. Our algorithm aligns various types of geographic features in document images with the corresponding vector data. In the experiment, our alignment algorithm increased the correctness and completeness of the extracted railroad and river vector data for about 100% and 20%, respectively. For the performance of feature recognition, the aligned vector data had a 100% improvement on the precision while maintained a similar recall.

CCS CONCEPTS

• **Information systems → Geographic information systems**;

KEYWORDS

Vector data alignment, Map processing, Historical documents

GeoAI'17, November 7–10, 2017, Los Angeles Area, CA, USA

ACM ISBN 978-1-4503-5498-1/17/11…$15.00
https://doi.org/10.1145/3149808.3149816

1 INTRODUCTION

Historical maps store abundant and valuable information about the evolution of natural features and human activities, such as changes in hydrography, the development of railroad networks, and the expansion of human settlements. Such knowledge represents a unique resource that can be extremely useful for researchers in the social and natural sciences to better understand how human and environment have evolved over time. A number of organizations scanned historical maps and stored them in digital archives. Many existing geographic feature recognition systems [1, 4–6, 8, 10, 11, 13, 15, 21] help convert the information in maps into a structured format by training machine learning models. However, one persistent limitation in current recognition systems is the need of the manual step for labeling or sampling to provide training data. A recognition process with such manual input requirements does not scale well for processing large numbers of maps stored in digital archives (e.g., the USGS Historical Topographic Series contains over 180,000 scanned maps).

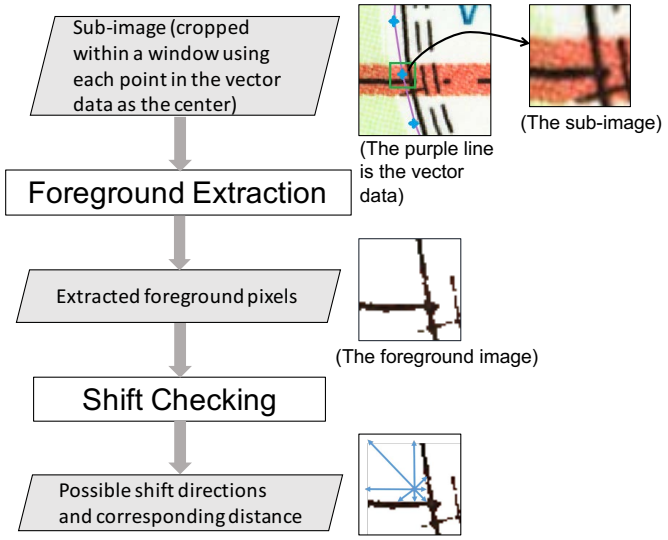
In our previous work, we built a prototype of a fully automatic geographic-feature-recognition system. The system eliminates the manual step by exploiting the fact that the map content usually changes gradually between map editions. For example, the geometry of railroads in a raster map in 2001 should be similar or the same as the geometry of contemporary railroad data (railroad vector data). Hence, our system uses contemporary data (vector data) to find the locations of the geographic feature of interest in historical maps automatically. If the vector data are aligned with the geographic features in maps, the system can generate representative training samples (images of geographic features) in which the geographic feature is at the center of the sample. Because the vector data and maps were generated in different years and from different sources, the vector data do not always align with the geographic feature in historical maps. Some training samples generated by using the vector data contain the geographic feature on the edge of the images or no geographic feature. As a result, the training samples would not have a consistent representation of the geographic feature. In this paper, we present an automatic alignment algorithm that corrects

the misalignment between vector data and raster maps to provide representative training data for the geographic feature recognition process.

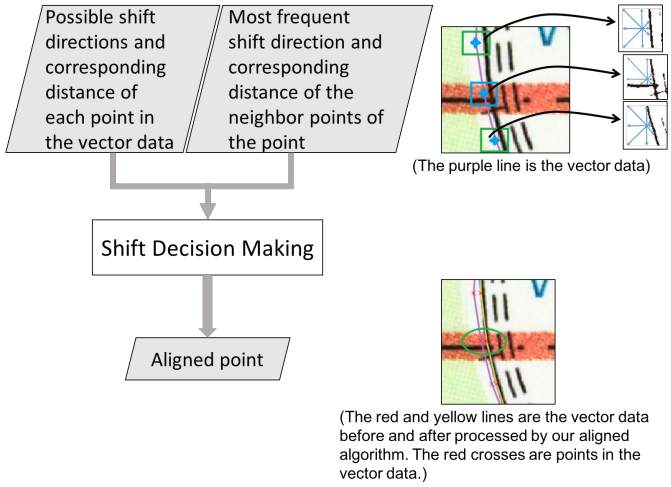
There are some existing algorithms that solve the misalignment problem between vector data and other types of geospatial data. The algorithms can be categorized into three groups: vector-and-vector-data alignment, vector-and-raster-data alignment, and raster-and-raster-data alignment. Our work falls into the category of vector-and-raster-data alignment. Most existing methods dealing with vector-and-raster-data misalignment problem work on aligning a specific geographic feature (e.g., roads) in imagery and vector data [2, 3, 19, 20], while our algorithm can align various types of geographic features in document images (maps) with vector data. In addition, the goal of our alignment algorithm is to find the accurate local positions of geographic feature to generate representative training samples for the recognition. In contrast, the goal of other methods is to improve the global positional accuracy. For example, Chen et al. [2] built the algorithm to conflate road vector data and roads in raster maps. Their method aligns points in vector data with the intersections of roads in maps. However, their method cannot guarantee that the line segment between two intersections aligns with the road on the map.

In this paper, we present an alignment algorithm that automatically aligns various types of geographic features in raster maps with the corresponding vector data from different data sources. The algorithm is based on two assumptions. The first assumption is that the misaligned vector data are close to the corresponding geographic features in maps. Based on this assumption, our algorithm searches the geographic feature of interest around a small neighborhood from the vector data. The second assumption is that nearby points in the vector data of linear geographic feature should have a similar or the same type of misalignment (the misalignment orientation and offsets). The middle figure on the top row in Figure 1(a) shows an example of the assumptions. (Figure 1 shows the workflow of our algorithm, which will be explained in the third section.) The purple line with blue dots (points in the vector data) is a segment of railroad vector data. The black line with the cross next to the purple line is a segment of a railroad. The purple line does not align with the black line, but it is very close to the black line, and all of the points in the vector data should be shifted towards east.

The contribution of this paper is an algorithm that automatically aligns linear vector data with multiple geographic features in raster maps. The aligned vector data enable the recognition system to generate representative training samples automatically to achieve accurate recognition results.



(a) The workflow of first two steps of the alignment algorithm



(b) The workflow of the third step of the alignment algorithm

Figure 1: The workflow of the vector data alignment algorithm

2 RECOGNITION SYSTEM CONTEXT

Our fully automatic geographic feature recognition system consists of three main components: data generation, recognition, and post-processing. Figure 2 shows the architecture of the system.

In the first component, the system generates training data without the manual work by using the contemporary vector data. The vector data guides the system to find the locations of the geographic feature of interest, and then the system generates the training data by cropping the map using a sliding window along the vector data. In the recognition process, the system employs the Convolutional Neural Networks (CNN). CNN models have shown impressive performance in image recognition [9, 16, 17]. The advantage of using the CNN model is that its generality compared to other machine learning models for image recognition like the Support Vector Machine (SVM). The CNN model can recognize different geographic features or the same geographic feature represented by different symbols without explicitly defining different feature descriptors to represent the geographic feature of interest. It can learn the feature descriptors by itself during the training. This advantage also makes processing large map archives efficient without manually defining different features to present different geographic features. In the post-processing process, the system employs a priori semantic knowledge of a particular type of geographic feature to improve the recognition results. For example, railroads should be a long

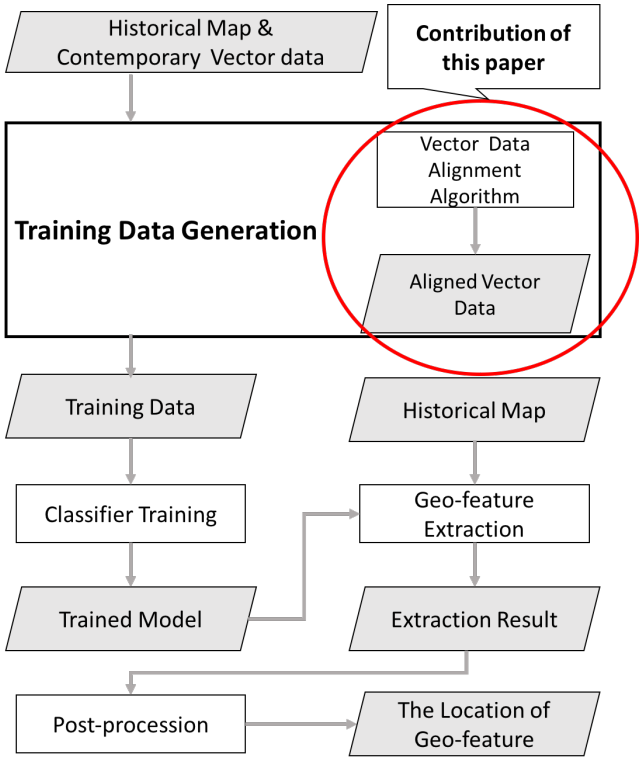
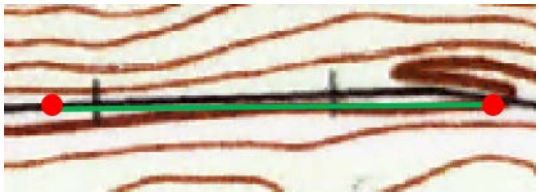
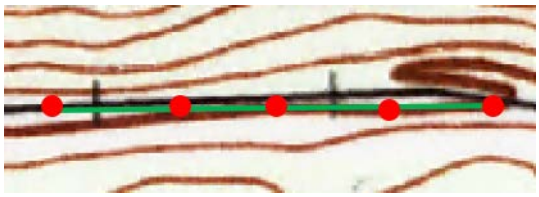


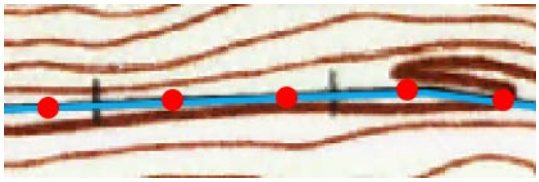
Figure 2: The workflow of the geographic feature recognition system



(a) The vector data (green) before interpolation



(b) The vector data (green) after interpolation



(c) The aligned vector data (blue) after interpolation

Figure 3: An example of interpolating points

continuous line. The system discards the recognition result if it is a short segment.

3 VECTOR DATA AND RASTER MAPS ALIGNMENT ALGORITHM

There are three steps in our alignment algorithm. Figure 1 shows the flowchart of the algorithm. Figure 1(a) shows the first two steps of the algorithm, and Figure 1(b) shows the third step. The figures next to inputs and outputs are examples of them. The first step is foreground extraction. The algorithm extracts the foreground in the sub-image, which includes the geographic feature of interest and could also contain other geographic features or map background. The second step is shift checking. The algorithm checks if the center pixel is the foreground pixel and calculates the closest foreground pixel along each direction. The last step is shift-direction and distance determination.

For the purpose of having a small neighborhood in which the transformation for correcting misalignment is consistent for points in the neighborhood, the algorithm generates interpolated points for each line segment in the vector data. Our algorithm uses the vector data after interpolation to crop sub-images as the inputs for the alignment algorithm. Our algorithm uses a length threshold to determine if the algorithm should interpolate points for a line segment. If the length of the line segment in the original vector data is longer than a length threshold, the algorithm interpolates points between the begin and the end points of a line segment to keep the distance between the begin and end point short. Then the system groups a certain number of points as a neighborhood along the geographic feature of interest. The neighboring points of a point are the ones in the same neighborhood. Figure 3 uses railroads as an example to illustrate why and how our algorithm interpolates points in a line segment. Red dots in Figures 3(a), 3(b), and 3(c) are points in the vector data. Figure 3(a) and 3(b) show the vector data in green before and after interpolation respectively. Although the points in Figure 3(a) are on the railroads, the vector data has the misalignment problem because the railroad is not a straight line. Our algorithm interpolates more points on the line segment. Figure 3(c) is the alignment vector data in blue after processed by our alignment algorithm using the interpolated vector data.

3.1 Foreground Extraction

The first step of the vector data alignment is extracting foreground pixels in the sub-image cropped by each point as the center within a window. The system uses the GrabCut algorithm without manual intervention to extract the foreground. The original GrabCut algorithm [14] is an iterative, interactive algorithm. The only human intervention in GrabCut is in initialization. GrabCut requires users to label the foreground pixels at the beginning. After initialization, GrabCut models the pixels as a graph. In each iteration, GrabCut uses a min-cut algorithm to segment the graph and an entropy function to calculate loss. The iteration ends when the entropy function converges. Our algorithm automatically initializes GrabCut by providing samples of the foreground and background.

Our algorithm initializes foreground pixels automatically by detecting the color range of the geographic feature of interest. If the pixel color is in range, the algorithm labels the pixel as a foreground pixel for initializing GrabCut. There are three steps in the color detection method. In the first step, the algorithm divides the HSV

color space into continuous ranges. We call each color interval a bin. In the second step, the algorithm calculates the number of pixels in each bin with an increasing buffer size of the vector data. In the third step, the algorithm calculates the growth rate of each bin. The growth rate is the difference between the number of pixels in the current bin and the number of pixels in last corresponding bin divided by the number of pixels in the last corresponding bin. For example, the growth rate of the black bin in the 3-pixel buffer in Figure 4(a) is the difference between the number of pixels in the black bin in the 3-pixel buffer and the number of pixels in the black bin in the 1-pixel buffer divided by the number of pixels in the black bin in the 1-pixel buffer. The algorithm chooses the bin with the highest growth rate for each buffer size. The algorithm detects the color range of the geographic feature of interest by analyzing the change of the bin with the highest growth rate. The assumption is that the vector data should be near the geographic feature of interest in the map. The bins with the highest growth rate in the first few buffers should be the ones containing the color of the geographic feature of interest. After the first few buffers, the buffers extend to the background or other geographic features, so the bin with the highest growth rate changes. The algorithm uses the range of the bin with the highest growth rate in the first few buffers as the color range of the geographic feature of interest to initialize GrabCut.

We use railroads as an example to illustrate how the algorithm detects the color of the geographic feature of interest. The algorithm divides the HSV color space into 15 bins. They are red, orange, yellow, chartreuse green, green, spring green, cyan, azure, blue, violet, magenta, black, gray, and white. Figures 4(a) - 4(d) show the growth rate charts of first four buffers. In practice, the algorithm removes the bins whose number of pixels is smaller than 1% of the total number of pixels. Figure 4(e) is an example of a railroad segment. The white lines in the left figure in Figure 4(f) is the 1-pixel buffer, and the right figure in Figure 4(f) shows the pixels in the left figure within the 1-pixel buffer. Figures 4(f) - 4(j) show that the buffer gradually covers the entire railroad when buffer size increases from 1 to 5 pixels and then extends to the area surrounding the railroads when the buffer size is larger than 5 pixels. In Figure 4(a) - 4(c), the black bin has the highest growth rate. When the buffer size is 7-pixel, Figure 4(i) shows that the buffer extends to the red area, the red bin has the second highest growth rate in Figure 4(c). When the buffer is 11-pixel, the bin with the highest growth rate becomes red. The bin with the highest growth rate in first three buffer is the color range of railroads.

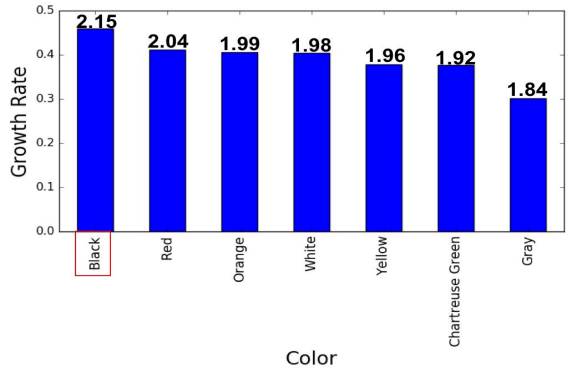
3.2 Shift Checking

This shift-checking step consists of two parts. The first part is checking if the center pixel of the cropped image (i.e., the location of a point) is a foreground pixel. In the second part, the algorithm searches foreground pixels in eight directions. The eight directions are north, south, west, east, northwest, northeast, southwest, and southeast to the center pixel. Checking all eight directions makes sure that the algorithm can find foreground pixels. The more directions the algorithm checks, the probability of finding a corresponding foreground pixel is higher. In the experiment, we found that eight directions are enough to find a corresponding foreground pixel for alignment. If there are foreground pixels along a direction, the algorithm saves the distance between the closest foreground pixel to the center pixel as the shift distance along that direction. Figure 5 uses railroads as an example to show how the shift checking works. Although the center pixel is a foreground pixel, the algorithm still needs to search foreground pixels along each direction, because the foreground pixels identified from GrabCut could be pixels of another geographic feature or map background. Figure 6 shows an example of this case. Figure 6(a) is the cropped image as the input for the GrabCut algorithm. The red point on Figure 6(a) is the center of the image window. The center is at a elevation contour instead of the railroad. Figure 6(b) is the foreground extracted by GrabCut. The red circle points out the center pixel of the foreground image, which is a pixel of the elevation contour.

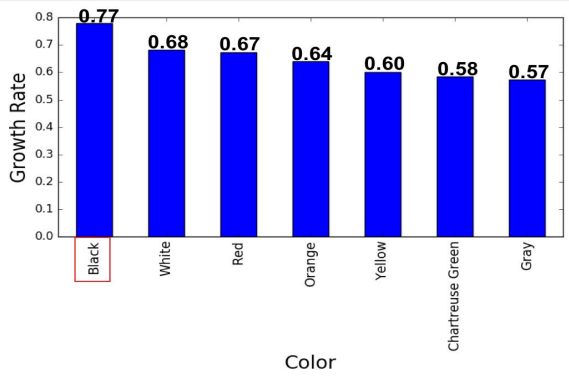
3.3 Shift Direction and Distance Determination

The algorithm uses points in the vector data and their neighbors to decide the shift direction and shift distance for alignment. Because of the assumption that a point should have a similar or the same type of misalignment as its neighboring points, considering neighboring points can help the algorithm transform the point correctly when it has several transformation options. Figure 8(a) shows an example about the assumption. The green line is the vector data, and points in the vector data are in the red in Figure 8(a). The black line with crosses next to the vector line is a segment of railroads, the geographic feature of interest. If the point in the blue circle is the point that needs to make a transformation, the shift direction is West according to its neighbors. The shift option provided by a neighborhood is the most frequent direction (MFDir) and the most frequent distance (MFDis). We define the most frequent direction (MFDir) as the direction towards which the majority neighboring points have the shortest shift distance. The algorithm finds MFDir by choosing the direction towards which the most neighboring points have the minimum shift distance. For example, there are five points, a, b, c, d, and e in Figure 7. The point c in red is the target point, the remaining points are its neighbors. The number on each arrow is the shift distance toward each direction. The direction with the shortest shift distance for a, b, and e is West, while the shortest shift distance for d is Southwest. The MFDir for point c is West in this case. For MFDis, the algorithm uses a subset of the neighboring points in which their shift direction with the shortest shift distance is MFDir. The algorithm chooses the shift distance that most points in the subset move towards MFDir as MFDis. In Figure 7, the subset is a, b, e. The shift distance of a, b towards West is three pixels, while the shift distance of e towards West is five pixels. In this case, MFDis for c is three pixels.

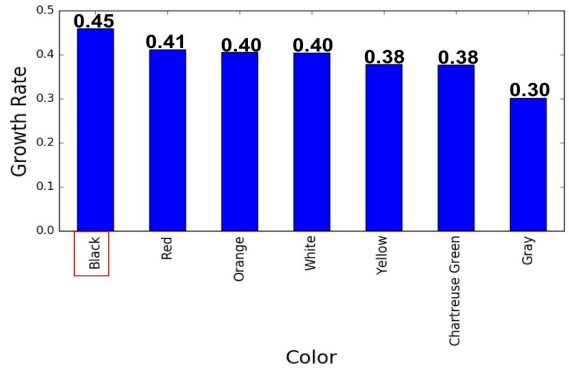
There are four alignment conditions. The first condition is that the difference between the shift distance of a point towards MFDir and its MFDis is smaller than a length threshold. The length threshold is set according to the assumption that the vector data is near the geographic feature of interest. If the difference is larger than the length threshold, the direction could not be the correct alignment direction. If the difference is smaller than the length threshold, the algorithm moves the point MFDis towards MFDir. Figure 8(a) is an example of this condition. The red point in the blue circle is on an elevation contour pixel. MFDir for the point is West. The shift distance of the point towards West is similar to MFDis. The point shifts MFDis towards West. If the difference between the shift distance of



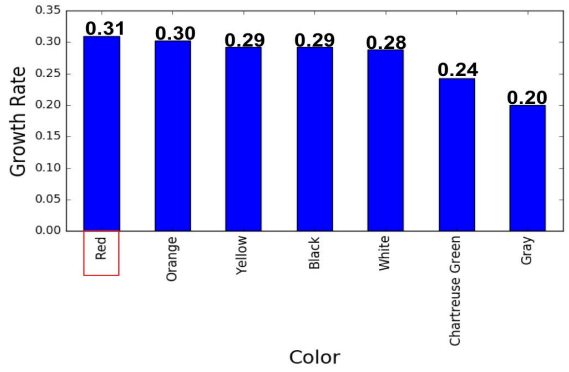
(a) Histogram of the growth rate in the 3-pixel buffer



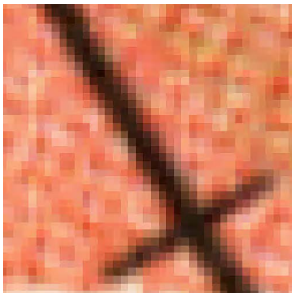
(b) Histogram of the growth rate in the 5-pixel buffer



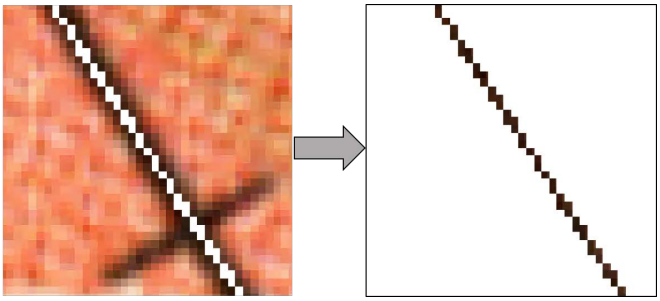
(c) Histogram of the growth rate in the 7-pixel buffer



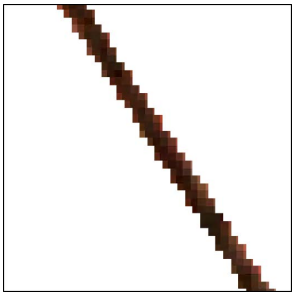
(d) Histogram of the growth rate in the 11-pixel buffer



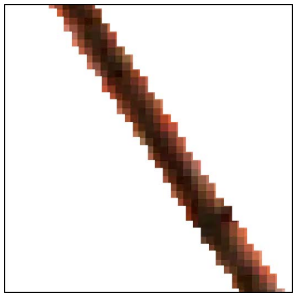
(e) An example of a railroad segment



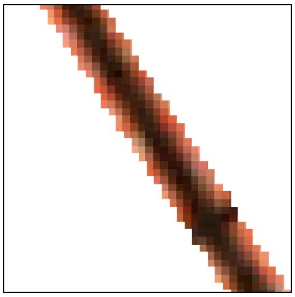
(f) The left figure is an example of the 1-pixel buffer (the white line) on the railroad segment. The right figure shows the pixels within the 1-pixel buffer of the left figure.



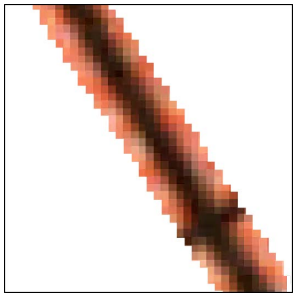
(g) An example of the pixels within the 3-pixel buffer



(h) An example of the pixels within the 5-pixel buffer



(i) An example of the pixels within the 7-pixel buffer



(j) An example of the pixels within the 11-pixel buffer

Figure 4: An example of color detection of railroads

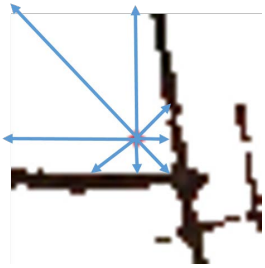


Figure 5: An example of the shift checking process

(a) A example image for the GrabCut algorithm

(b) The foreground of the left figure extracted by the GrabCut algorithm

Figure 6: An example of the center pixel not on geographic feature of interest

the point towards MFDir and MFDis is larger than the threshold, the point cannot shift towards MFDir. The rest of three conditions deals with this situation. The second kind of alignment condition is that the point is on the foreground pixel. In this case, the system keeps the point in the original position. Figure 8(b) is an example of this condition, where MFDir is West. The point is the red point in the blue circle. The algorithm cannot find a foreground pixel towards West, so the algorithm keeps it at the original position. The third condition is that the point is on the background pixel, and there is no foreground pixel in the sub-image. The algorithm removes the point. Figure 8(c) is an example of rivers about this situation where the contemporary vector data contain newly built canals while the historical maps do not. In fourth alignment condition, the point is on the background pixel, and there are foreground pixels in the sub-image. The algorithm moves the point towards the direction with the shortest shift distance. Figure 8(d) is an example of this case. The neighbors of red railroad point in the blue circle are on the railroads, so its MFDis is set to infinity (i.e., its neighbors do not have foreground pixels in the eight directions). Hence the point in the blue circle shifts towards West with the shortest shift distance.

4 EXPERIMENT

We tested our alignment algorithm on railroad and river geographic features in a large-scale map. Our goal to align vector data is to improve the geographic feature extraction performance. For that purpose, we tested the performance of our recognition system on railroads using the vector data processed by the alignment algorithm and compared the performance with other two groups of vector data. One group is the original vector data downloaded from US Census, the other group is the manually aligned vector data.

4.1 Data Preparation

We downloaded the map from the USGS website. The location of the map is Bray in California circa 2001. We downloaded the 2016 railroad and river vector data from the US Census website.

The test map contains rich and diverse graphical conditions, which is used to test the alignment and extraction performance with a variety of graphical variations. For example, the geographic features around the feature of interest are diverse. Figure 9 shows the diverse neighborhood around railroads. In Figure 9, from left to right, they are elevation contours, gravel roads, regular roads, and rivers around railroads. The second aspect is that the map contains diverse geographic features. There are railroads, elevation contours, rivers, roads, trails, highways, and buildings on the map. Figure 10 shows examples of some geographic features in the map. From left to right, they are railroads, elevation contours, rivers, gravel roads, and regular roads.

In the geographic feature alignment experiment, for the purpose of having small neighborhoods and fitting curved features in the map, the algorithm interpolated points in the vector data. The vector data downloaded from US Census did not have a fixed distance between two points in a line segment. The distance of a segment after interpolation was around 50 pixels in the map. Our algorithm grouped 10 points in the vector data as a neighborhood. In another word, each point had nine neighboring points. For the sub-image, the input for alignment algorithm, the algorithm cropped sub-images within the 47*47-pixel window. The center of each sub-image was a point in the vector data. To have the geographic feature of interest in the sub-image, we choose the 47*47-pixel window that was enough to include the geographic feature of interest in the sub-images.

To generate training data containing railroads for the CNN model, the system cropped sub-images along the lines in the map aligning with the vector data with a one-pixel step. The size of training samples is 47*47. In the experiment, we tested on three groups of training data. The training data containing railroads were generated from three sets of vector data. The training data in the first group was generated by the vector data downloaded from the US Census website (the original group). The training data in the second group was generated by the vector data processed by our alignment algorithm (the aligned group). The training data in the third group was generated by vector data manually aligned with the railroads (the ground truth group). Samples in which the railroad is at the center are called positive samples. Otherwise, they are called negative samples.

4.2 Vector Data Alignment Experiment and Analysis

We tested our alignment algorithm on railroads and rivers. We used correctness and completeness proposed by Heipke et al. [7] as the metrics to evaluate the alignment performance. The definition of correctness is the length of true positives divided by the sum of the lengths of true positives and false positives. The definition of completeness is the length of true positives divided by the sum of the lengths of true positives and false negatives. The top figure in Figure 11 shows the true positives and false positives in the correctness, and the bottom figure in Figure 11 shows the true positives and false negatives in the completeness. The maximum of

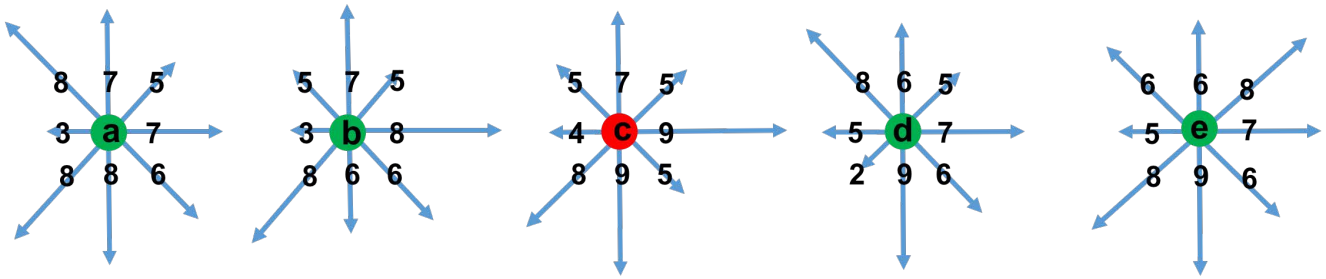


Figure 7: An example to illustrate the most frequent direction (MFDir) and the most frequent distance (MFDis)

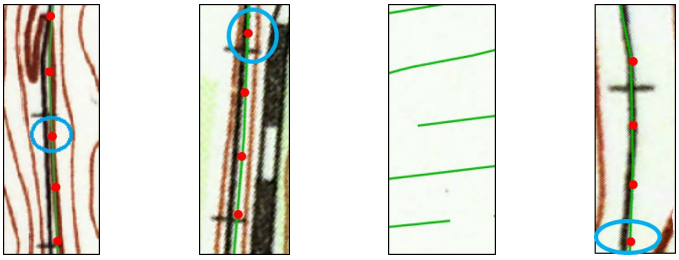


Figure 8: Examples of alignment conditions

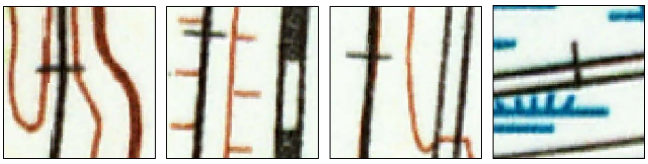


Figure 9: Examples of the diverse background around railroads

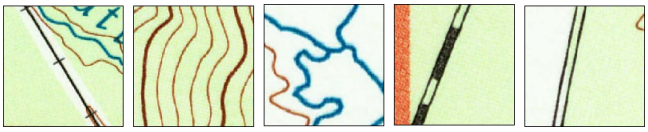


Figure 10: Examples of geographic features in the map

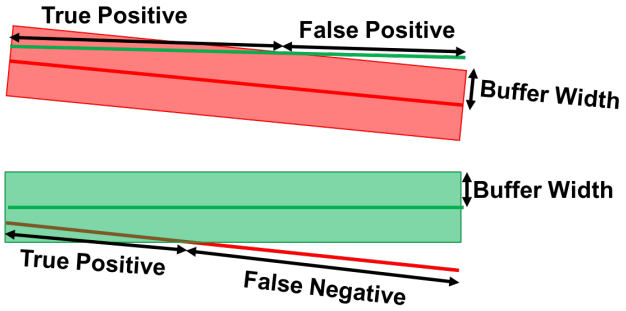


Figure 11: An example of explaining correctness and completeness (The red line is the extracted result, the green line is the reference.)

Table 1: The Alignment Performance of Railroads varied in the buffer size from 1-pixel to 7-pixel

	Buffer	1	3	5	7
Correctness	Original	13.7%	22.9%	36.4%	45.4%
	Aligned	63.8%	91.3%	93.7%	94.4%
Completeness	Original	11.4%	18.0%	28.6%	37.2%
	Aligned	24.1%	63.5%	67.2%	68.3%

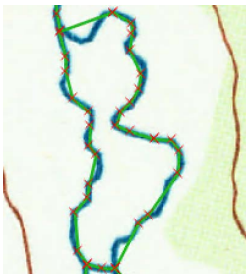
Table 2: The Alignment Performance of Rivers varied in the buffer size from 1-pixel to 7-pixel

	Buffer	1	3	5	7
Correctness	Original	40.0%	63.9%	68.3%	70.0%
	Aligned	38.7%	80.4%	85.3%	87.7%
Completeness	Original	54.6%	86.8%	91.0%	92.5%
	Aligned	80.9%	91.4%	92.7%	92.8%

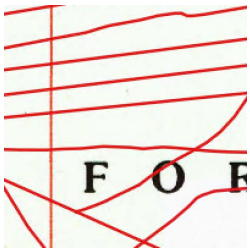
the correctness of both railroads and rivers was 100%. The maximal value of the completeness for railroads was 82.89%, and for rivers was 98.06%, because the original vector data did not cover the entire geographic feature.

Tables 1 and 2 show the alignment performance of railroads and rivers separately. We evaluated the performance of alignment using different buffer sizes on the ground truth. The results from increasing the buffer from 1-pixel to 7-pixel showed how far away the aligned vector data were from the center line of the geographic feature. For example, the 3-pixel buffer indicates that the vector data is within one pixel of the center line of the geographic feature.

From Table 1, we can see that the correctness and completeness of the aligned vector data were more than twice higher than those of the original railroad vector data. The correctness of original vector data was only 45.4% when buffer size was seven, which indicated more than half of the original vector data was three pixels away from the railroad center line. We could also see the same pattern from the completeness of original vector data. The correctness and completeness of aligned vector data had a large increase from the 1-pixel buffer to the 3-pixel buffer, which indicated most of the aligned vector data was on the center line or one pixel away from the center lines of railroads.

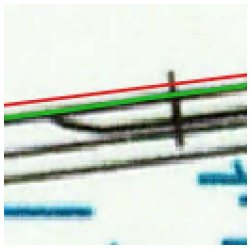


(a) An example of sharp turns (The vector data are in green. The red crosses are points in the vector data.)

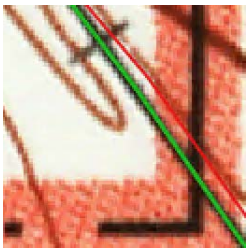


(b) An example of no rivers on the map (The vector data are in red.)

Figure 12: Examples of river vector data

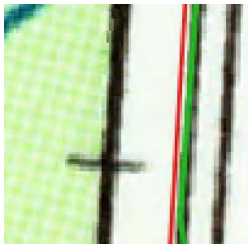


(a)

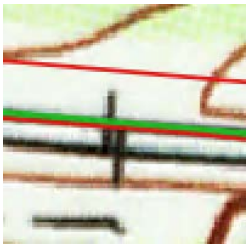


(b)

Figure 13: Examples of misalignment problems (The original vector data are in red. The aligned vector data are in green.)



(a) An example of the first group of alignment error



(b) An example of the second group of alignment error

Figure 14: Examples of alignment errors (The original vector data are in red. The aligned vector data are in green)

From Table 2, the correctness of the aligned river vector data increased about 15% compared to the correctness of the original vector data. The correctness and completeness of both original and aligned vector data had a large increase from the 1-pixel buffer to the 3-pixel buffer, which indicates that most of the vector data were on the center lines of rivers or 1 pixel away. The river vector data had less misalignment problem than the railroad vector data. The river correctness and completeness of the original vector data were higher than correctness and completeness of the original railroad vector data. The correctness of the aligned rivers vector data was slightly lower than the correctness of aligned railroads vector data because the length of each segment that we set cannot fit the sharp curves in rivers. We set the length of each segment is around 50 pixels, which is not short enough to fit the curves along rivers. Figure 12(a) shows an example with sharp turns. The red crosses are points in the vector data. The completeness of original vector data and aligned one were similar because the main misalignment problem for rivers was that there were no rivers around the locations that the vector data covered. Figure 12(b) shows an example of the problem. The red lines in Figure 12(b) are the river vector data, which does not have corresponding features on the map.

Overall, our alignment algorithm achieved high correctness and completeness. Figure 13 shows the misalignment examples where our algorithm worked well. In Figure 13, red lines are the original vector data, and green lines are the aligned vector data. The first misalignment example is the vector data on the background and close to the geographic feature of interest (Figure 13(a)). The black line with crosses in Figure 13(a) is a railroad segment. The second misalignment example is that some points in the vector data are on another geographic feature (Figure 13(b)). The brown lines in Figure 13(b) are elevation contours, and a part of the original vector data locates on the contour lines. After applying the alignment algorithm, the vector data in both examples aligned with the railroad automatically. These two examples were solved by the alignment condition in our algorithm that the difference between shift distance and MFDis is within a distance threshold.

We categorize the alignment errors in the experiment into two groups. The first group is that another geographic feature was closer to the vector data than the geographic feature of interest. Figure 14(a) shows an example of railroads about this case. The left black line with crosses is a railroad segment, while the right black line is a road segment. The original vector data in red is closer to the road. The vector data in green aligned with the road after implementing our algorithm. The reason for this misalignment was that the alignment algorithm shifted points by finding the shift direction with the shortest shift distance. The shift direction with the shortest shift distance, in this case, was towards another geographic feature, road. The second group is parallel lines. Figure 14(b) shows an example of parallel railroads. Both vector lines moved to the same railroad segment after applying the algorithm. The algorithm did not consider points in the parallel line as neighbors. The algorithm could be extended to detect the conflict considering the shift directions of parallel lines.

4.3 Geographic Feature Extraction Experiment and Analysis

We tested if the automatic alignment algorithm could improve the performance of geographic feature extraction on railroads and evaluated the extracted result using precision and recall. We did three group of experiments. In both Tables 3 and 4, "original" means the training data containing railroads was generated by the vector data downloaded from the US Census, "aligned" is the group in which the training data containing railroads was generated by the vector data aligned by our alignment algorithm. In the "ground truth" group, the training data containing railroads was generated by the vector data manually aligned with railroads. We quantified the quality of training data generated by the three sets of vector data using correctness and completeness. Correctness indicates if the center pixels of positive training samples are railroads, and completeness indicates the diversity of positive training samples. Table 1 shows the correctness and completeness of the original

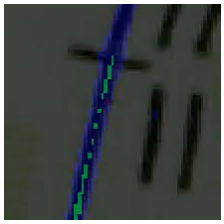
Table 3: The performance of railroads extraction

Vector	Original	Aligned	Ground Truth
Precision	20.25%	28.46%	36.81%
Recall	97.84%	95.09%	99.97%
F_1	33.56%	43.81%	53.81%

Table 4: The performance of railroads extraction after post-processing

Vector	Original	Aligned	Ground Truth
Precision	29.30%	57.98%	71.84%
Recall	98.38%	92.36%	99.97%
F_1	45.14%	71.23%	83.60%

(a) The extraction result of the original (purple) and aligned (green) group



(b) The extraction result of the aligned (green) and ground truth (blue) group

Figure 15: Examples of extraction results.

and aligned railroads vector data. In addition, the correctness and completeness of ground truth railroad vector are 100%. The quality of training data gets better with the order of the original, the aligned, and the ground truth group.

In Table 3, the increase of precision indicates that the quality of training data had improved the extraction result. The recall in the aligned group was lower than the recall in the original group because some railroad segments were misclassified as negatives in the aligned group. Figure 15(a) is an example of this case. The bottom layer is the map. The layer with the purple line is the extraction result using the original vector data. The layer with green line is the result using the aligned vector data. We found that some railroad segments were not or partially recognized by the system using the aligned vector data. Figure 15(b) shows an example. The blue line in figure 15(b) is the recognition result in the ground truth group, and the green line is the result from the aligned group. Because of the partial recognition in the aligned group, the true positives in the aligned group were less than the ones in the ground truth group, and the false negatives in the aligned group were more than the ones in the ground truth group. Overall, the recognition performance of our system in the aligned group improved by 10% in the F_1 score.

Besides the increased precision, the extraction quality was better when the training data more consistently represented the geographic feature (i.e., using the aligned vector data for labeling the training samples). The width of extracted railroads in the original group was much wider than the actual width of railroads on the map, while the width of extracted railroads in aligned and ground truth group was similar to the actual railroad width. Figure 16 shows

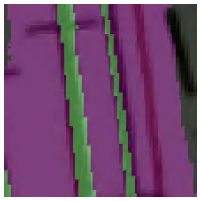
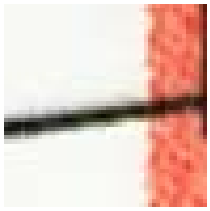
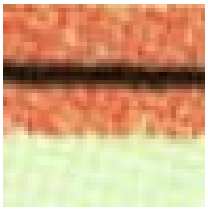


Figure 16: An example of the extraction result of the original group (purple) and the aligned group (green)



(a) A sample with railroads (47*47 pixels)



(b) A sample with roads(47*47 pixels)

Figure 17: Examples of testing samples.

the comparison of extraction results of the original and aligned group, in which there are parallel railroads on the test map. The purple layer is the recognition result using the original group and the layer with green lines is the result of using the aligned group. The original group could only recognize the parallel railroads as a wide railroad segment, while the aligned group could extract two parallel railroads.

The reason that the precision was low in all three groups was that some geographic features were similar as railroads within the image window. Figure 17 shows an example. Figure 17(a) is the sample containing railroads, while figure 17(b) is the sample that contains roads. The system misclassified many testing samples containing roads like this as railroads.

In the post-processing, we used the semantic knowledge of railroad to remove some false positives. The semantic knowledge we employed in the experiment was that the railroad is a long continuous linear geographic feature. In the post-processing, the system removed the short line segments based on the semantic property of the railroad.

Table 4 is the evaluation of the extraction result after post-processing. The recognition performance of our system in the aligned group increased around 55% on the F_1 score compared with the performance in the original group. The precision increased by removing small false-positive segments. The recall in the aligned group decreased a little compared with the one before post-processing, because of the partial recognition. The green line in Figures 15(a) and 15(b) is a partially recognition example. The true-positive small segments were removed as well in this case. The recall in the original group increased a little is because of the close operator used in post-processing. The system applied the close operator (a dilation followed by an erosion image operator) to make small true positive segments connected. This operator made some non-railroad pixels in the corners recognize as railroad pixels. Because of the wide width of extracted railroad from the original group, these corner pixels extend to the next parallel railroad in some place. In that

case, the number of true positives increased a little, which is the reason that the recall after post-processing increased a little.

5 RELATED WORK

In this paper, we presented an algorithm that automatically aligns vector data with linear geographic features in historical scanned maps. The alignment algorithm significantly improved the feature extraction performance.

Some other researchers work on extracting geographic features (roads) in aerial or satellite images by using Deep Convolutional Neural Networks (DNN). They use the road vector data to generate training samples and evaluate extraction results. Wang et al. [18] built the DNN to extract roads in satellite imagery by using road vector maps to generate training data. Because there were not enough points in the original vector maps to fit the road curves, they refined the vector data by interpolating points to avoid roads turning sharply. They had better extraction result by refining the vector maps than using the original vector maps. Mnih et al. [12] also extracted roads by using the road vector data to generate training data and did the evaluation. Because there were some noises in the road vector data, they used relaxed metrics to evaluate the results. Both studies show that the vector data alignment significantly influences the feature extraction performance.

Some other researchers proposed algorithms to solve the vector-and-raster-data misalignment problem. Chen et al. [2, 3] invented an algorithm that can automatically conflate road vector data with orthoimagery. Their method has three steps. First, the algorithm finds intersection point pairs. Second, the algorithm detects inaccurate point pairs. In the last step, they align these points with other objects in the imagery. Their method can only apply on roads because their used intersections, a specific characteristic of roads to align road vector data with imagery, while our approach does not assume a specific feature type, and we tested on both railroads and waterlines. Another difference between our works and the others is the data type. They work on satellite imagery, while we work on document images. Wu et al. [19, 20] proposed an algorithm dealing with road vector data alignment with aerial imagery. Their approach consists of four steps. In the first step, the algorithm detects road feature around the road vector data. In the second step, the algorithm clusters roads according to the orientation of the roads. In the third step, the algorithm selects a reliable roads subset. In the last step, the algorithm minimizes the cost function with the selected subset. The final selected subset is the aligned road vector data.

6 DISCUSSION AND FUTURE WORK

We presented an algorithm to accurately align linear vector data with historical maps with high correctness and completeness. The aligned vector data helps our recognition system achieve better performance than using the original vector data.

We plan to improve the alignment algorithm by solving the two groups of errors found in the experiment section. To prevent the vector data from aligning with other nearby geographic features, we plan to set the number of points in one neighborhood greater than 10. For handling parallel lines, we plan to find a way to include the points in the nearby parallel line as a neighborhood when aligning a point. We also plan to relax the criteria that the linear geographic feature needs to be smooth to align vector data with sharp turns.

For the feature recognition system, we are going to improve the recognition performance in two ways. In the experiment, we found that many false positives were very similar to the true positives within the window. Hence, one way that we plan to improve the system is to try different window sizes to find more representative training samples. In the related work, we found some researchers use DNN to extract geographic features, which has good performance. We plan to build a DNN model and test if the model can improve the recognition performance.

ACKNOWLEDGEMENTS

This material is based on research supported in part by the National Science Foundation under Grant No. IIS 1563933 (to the Univesity of Colorado at Boulder) and IIS 1564164 (to the university of Southern California). We gratefully acknowledge the support of Microsoft Corporation with the Azure for Research Award and NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

REFERENCES

[1] V Bucha, S Uchida, and S Ablameyko. 2006. Interactive road extraction with pixel force fields. In *ICPR.*, Vol. 4. IEEE, 829–832.
[2] C-C Chen, C A Knoblock, and C Shahabi. 2006. Automatically conflating road vector data with orthoimagery. *GeoInformatica* 10, 4 (2006), 495–530.
[3] C-C Chen, C A Knoblock, and C Shahabi. 2008. Automatically and accurately conflating raster maps with orthoimagery. *GeoInformatica* 12, 3 (2008), 377–410.
[4] Y-Y Chiang and C A Knoblock. 2011. Recognition of multi-oriented, multi-sized, and curved text. In *ICDAR.* IEEE, 1399–1403.
[5] Y-Y Chiang and C A Knoblock. 2013. A general approach for extracting road vector data from raster maps. *IJDAR* (2013), 1–27.
[6] D B Dhar and B Chanda. 2006. Extraction and recognition of geographical features from paper maps. *IJDAR* 8, 4 (2006), 232–245.
[7] C Heipke, H Mayer, C Wiedemann, and O Jamet. 1997. Evaluation of automatic road extraction. *International Archives of Photogrammetry and Remote Sensing* 32, 3 SECT 4W2 (1997), 151–160.
[8] T C Henderson and T Linton. Raster map image analysis. In *ICDAR.* IEEE, 376–380.
[9] A Krizhevsky, I Sutskever, and G Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems.* 1097–1105.
[10] S Leyk and R Boesch. 2010. Colors of the past: color image segmentation in historical topographic maps based on homogeneity. *GeoInformatica* 14, 1 (2010), 1–21.
[11] Q Miao, P Xu, T Liu, Y Yang, J Zhang, and W Li. 2013. Linear feature separation from topographic maps using energy density and the shear transform. *IEEE Transactions on Image Processing* 22, 4 (2013), 1548–1558.
[12] V Mnih and G Hinton. 2010. Learning to detect roads in high-resolution aerial images. *ECCV* (2010), 210–223.
[13] A Pezeshk and R L Tutwiler. Contour line recognition & extraction from scanned colour maps using dual quantization of the intensity image. In *SSIAI.* IEEE, 173–176.
[14] C Rother, V Kolmogorov, and A Blake. 2004. Grabcut: Interactive foreground extraction using iterated graph cuts. In *(TOG)*, Vol. 23. ACM, 309–314.
[15] R Samet and E Hancer. 2012. A new approach to the reconstruction of contour lines extracted from topographic maps. *JVCIR* 23, 4 (2012), 642–647.
[16] Ali Sharif R, H Azizpour, J Sullivan, and S Carlsson. 2014. CNN features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops.* 806–813.
[17] K Simonyan and A Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556* (2014).
[18] J Wang, J Song, M Chen, and Z Yang. 2015. Road network extraction: A neural-dynamic framework based on deep learning and a finite state machine. *IJRS* 36, 12 (2015), 3144–3169.
[19] X Wu. 2011. Method for automatic alignment of raster data with vector data in a geographic information system. (Jan. 11 2011). US Patent 7,869,667.
[20] X Wu, R Carceroni, H. Fang, S Zelinka, and A Kirmse. 2007. Automatic alignment of large-scale aerial rasters to road-maps. In *Proceedings of the 15th annual ACM international symposium on Advances in geographic information systems.* ACM, 17.
[21] D Xin, X Zhou, and H Zheng. 2006. Contour line extraction from paper-based topographic maps. *JICS* 1, 5 (2006), 275–283.