Resolving Ambiguities in Toponym Recognition in Cartographic Maps

Alexander Gelbukh[1,3], Serguei Levachkine[2], and Sang-Yong Han[3*]

[1] Natural Language Processing Lab, and [2] Image Processing and Pattern Recognition Lab
Centre for Computing Research (CIC) - National Polytechnic Institute (IPN)
{gelbukh, palych}@cic.ipn.mx

[3] Computer Science and Engineering Department,
Chung-Ang University, Korea; * corresponding author
hansy@cau.ac.kr

Abstract. To date many methods and programs for automatic text recognition exist. However there are no effective text recognition systems for graphic documents. Graphic documents usually contain a great variety of textual information. As a rule the text appears in arbitrary spatial positions, in different fonts, sizes and colors. The text can touch and overlap graphic symbols. The text meaning is semantically much more ambiguous in comparison with standard text. To recognize a text of graphic documents, it is necessary first to separate it from linear objects, solids, and symbols and to define its orientation. Even so, the recognition programs nearly always produce errors. In the context of raster-to-vector conversion of graphic documents, the problem of text recognition is of special interest, because textual information can be used for verification of vectorization results (post-processing). In this work, we propose a method that combines OCR-based text recognition in raster-scanned maps with heuristics specially adapted for cartographic data to resolve the recognition ambiguities using, among other information sources, the spatial object relationships. Our goal is to form in the vector thematic layers geographically meaningful words correctly attached to the cartographic objects.

1 Introduction

Huge amount of geographic information collected in the last centuries is available in the form of maps printed or drawn on paper. To store, search, distribute, and view these maps in the electronic form they are to be converted in one of digital formats developed for this purpose. The simplest way of such conversion is scanning the paper map to obtain an image (a picture) stored in any of the raster graphic formats such as TIFF, GIF, etc. After that, a raster-to-vector conversion should be applied to include obtained vector maps into a Geographic Information System (GIS).

Though raster representation has important advantages in comparison with the hard copy form, it still does not allow semantic processing of the information shown in the map, for example:

- Search for objects: *Where is Pittsburgh? What large river is in Brazil?*
- Answering questions on the spatial relations: *Is Tibet in China? Is Nepal in Tibet? Is a part of Tibet in China?*
- Generation of specialized maps: *Generate a map of railroads and highways of France.*
- Scaling and zooming: *Generate a 1:125 000 map of Colombia. Show me more details at the point under cursor.*
- Compression: *Objects like points, arcs, or areas can be stored much more efficiently than pixels.*

Note that these are semantic tasks rather than image manipulation. E.g., when zooming in or out, objects and, most importantly, their names should appear or disappear rather than become smaller or larger. Indeed, when zooming out the area of London, the name *Greenwich* should not become small to unreadable but should disappear (and appear in an appropriate font size when zooming in).

This suggests storing and handling of a map as a database of objects (points, arcs, areas, alphanumeric, etc.)—vector database—having certain properties, such as size, color, geographic coordinates, topology, and name. Specifically, the name of the object is to be stored as a letter string rather than a set of pixels as originally scanned from the hard copy. Thus, such vector representation can solve the listed above semantic tasks, but only to some extent [1].

However, automatic recognition of such strings (toponyms) in the raster image of the map presents some particular difficulties as compared with the optical character recognition (OCR) task applied to standard texts such as books:

- The strings are out of context, which prevents from using standard spelling correction techniques based on the linguistic properties of coherent text. Moreover, often such strings are even not words of a specific modern language, which further limits applicability of the standard linguistic-based spelling correction methods.
- The background of the string in the map is very noisy since it can contain elements of geographic notation such as shading or hatching, cartographic objects such as cities or rivers, and even parts of other strings, e.g., name of a city inside of the area covered by the name of the country; see Figure 1.
- In addition, often the letters of the string are not properly aligned but instead are printed under different angles and along an arc; this happens with the names of linear and area objects, e.g., rivers or countries; see Figure 1.
- Unlike standard task, in toponym recognition it is not only required to recognize the string itself but also to associate it with a specific cartographic object, e.g., city, river, desert, etc.

On the other hand, in many cases additional information is available that can give useful cues for ambiguity resolution. One of such information sources is existing databases (usually available from the country Government, postal service, etc.) providing spatial relationships between entities (e.g., a list of cities classified by administrative units) or even exact coordinates.

In this paper we discuss how such additional information can be used to workaround the problems arising in recognition of the inscriptions in the maps, associat-

ing them with specific cartographic objects, and importing information on these objects from available databases. This paper reports work in progress. Its purpose is to identify the possible pitfalls and problems in this process, which we for simplicity illustrate on artificial examples. Discussion of real-world experimental results is beyond the scope of this paper.

First, we describe the general scheme of our method, which consists in combining the information from different sources of evidence (such as toponym databases, linguistic dictionaries, information on the fonts and distribution of the letters in the raster image, etc.) with subsequent verification of the obtained results. Then we discuss each individual source of evidence, as well as the verification procedure. Finally, conclusions are drawn and future work directions are outlined.

2 Previous Work

The text segmentation and its subsequent recognition in raster images are very difficult problems; they are complicated by the presence of the text embedded in graphic components and the text touching graphics [2]. These challenging problems have received numerous contributions from the graphic recognition community [3]. However, there have not been yet developed any efficient programs to solve the task automatically. Thus, in the most systems human operator is involved. For example, [4] proposes that the operator draws a line through the text, marking it as text and revealing its orientation.

In [5] and [6], the algorithms are developed to extract text strings from text/graphics images. However, both methods assume that the text does not touch or overlap with graphics. For maps, the problem is much more complex, since the touching or overlapping as well as many other character configurations are commonly presented in maps. That is why [7], [8], and [9] developed the methods for text/graphics separation in raster-scanned (color) cartographic maps.

In [9] a specific method of detecting and extracting characters that are touching graphics in raster-scanned color maps is proposed. It is based on observation that the constituent strokes of characters are usually short segments in comparison with those of graphics. It combines line continuation with the feature line width to decompose and reconstruct segments underlying the region of intersection. Experimental results showed that proposed method slightly improved the percentage of correctly detected text as well as the accuracy of character recognition with OCR.

In [7] and [8], the map is first segmented to extract all text strings including those that touch other symbols and strokes. Then, OCR using Artificial Neural Networks (ANN) is applied to get the coordinates, size, and orientation of alphanumeric character strings in the map. Then, four straight lines or a number of "curves" computed in function of primarily recognized by ANN characters are extrapolated to separate those symbols that are attached. Finally, the separated characters are input into ANN again for their final identification. Experimental results showed 95–97% of successfully recognized alphanumeric symbols in raster-scanned color maps.

In the present work, we use the output obtained with this method in combination with pre-existing geographical information in semantic analysis of ambiguities for "geographically meaningful" word formation. We focus on text processing rather than image processing.

The proposed system is based both on the traditional techniques used in the general-purpose OCR programs and on those we developed specifically for cartographic maps. Sections 4, 5, and 8 deal with the problems and solutions common to any OCR task. However, even in these cases there are some differences with respect to the usual OCR situation. The algorithm described in Section 4 (check against a dictionary of existing words) in our case has to deal with much more noisy strings than usual OCR programs developed for clean black-on-white running text. The same can be said of Section 5 (non-uniform spatial letter distribution): in maps the letters are often placed at significant distances one from another, cf. Figure 1; as well of Section 8 (check against the general laws of a given language): maps have many foreign or indigenous words that do not conform to the main language of the given territory.

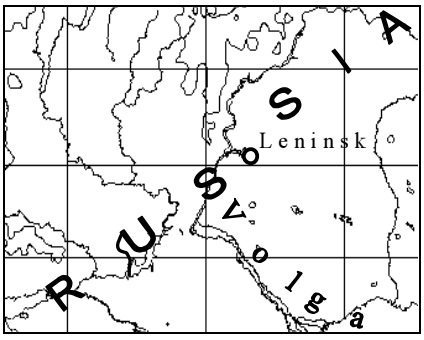

Fig. 1. Intersection of strings in a map

In contrast, Sections 6, 7, and 9 are specific for maps. In Section 6 (check against the geographic information, e.g., expected coordinates) consistency with the available information about the location of an object is used, which is obviously specific for cartographic maps. In Section 7 (check against the cartographic notation) information on the expected type of the object (river, mountain, etc.) is used. In Section 9 (global consistency check) it is verified that each object is recognized only once.

These techniques do not have direct analogs in standard OCR research and thus are contributions of our paper.

On the other hand, many techniques standard for usual text OCR use contextual information available in running text, such as morphological and syntactic analysis, semantic consistency verification [13]; paragraph layout detection, etc. These techniques are not directly applicable to recognition of toponyms in maps. However, the new techniques we introduce in the Sections 6, 7, and 9 play a similar role of verification of contextual consistency of the recognition results, but in the manner very specific to cartographic maps in contrast to the standard running text OCR.

3 Main Algorithm

We rely on a basic OCR procedure[1] (described in [1], [7], and [8]) that recognizes in the map individual letters and groups together the letters of a similar font and color located next to each other, thus forming in a hypothetical string. In this process, errors of various types can be introduced; our purpose is to detect and correct them.

The recognition algorithm for the whole map works iteratively. At each step, the basic OCR procedure selects for processing the longest clearly recognizable string *s*. Upon its recognition, the string is removed from the raster image so that the image becomes simpler and other strings, possibly disturbed by *s*, become clearly recognizable. Then the next most clearly recognizable string is selected, etc. The algorithm stops when no more letter strings can be found in the raster image.

This design allows for recognition of the names of large areas, which are usually represented by large letters scattered across the map, with many names of smaller objects between the letters of the area name. In the example shown in Figure 1, first the word *Leninsk* will be recognized and removed from the image, then the word *Volga*, and only then the letters *R-u-s-s-i-a* can be grouped together in a string.

Indeed, in the original image, the letter *L* of *Leninsk* disturbs the string *Russia* making it look like RUSLSIA or RUSLeninsk (since the direction of the string is not easy to determine). In this particular case the difference in orientation and size of the letters can be used to filter the $_L$ off the string RUS$_L$IA; however, taking into account such information would significantly complicate processing and anyway would produce unreliable results. Instead, we rely on a simpler solution: the smaller string *Leninsk*, not interleaved with any other string, is recognized first and is erased from the image. Now, the string RUSSIA is not disturbed and can be easily recognized.

The basic OCR procedure returns, for each string it recognizes, the string itself, e.g., "RUSSIA," and the geographic coordinates in the map of the frame containing each individual letter, e.g., R, U, etc. In fact, the word can be recognized with errors, e.g., "RNSoSIA," where U is erroneously recognized as N due to a nearby river, and the circle representing a city is erroneously taken for the letter o. Such errors are detected and corrected using the following algorithm.

1. The obtained string is looked for in a list (dictionary) of expected toponyms, which (if the word is found) provides the semantic information associated with it, such as the type of object (e.g., city, river), its spatial relationships (e.g., administrative unit it is in), and its geographic coordinates if available. This information is verified using different sources of evidence, such as spatial distribution of the letters in the raster image, the coordinates of the letters, etc.
2. In addition, similar strings (e.g., *RUSSIA*, *ASIA*, *Angola*, etc. for *RNSoSIA*) are looked for in the dictionary and for them, the same information is retrieved and the same check is performed, an additional source of evidence being the probability of the corresponding changes in the letters of the string.

[1] Our method does not depend on how text strings were extracted and recognized. Neither does it depend much on the type of graphic document being processed. It can be adapted to different subject domains.

3. The scores of all sources of evidence are multiplied to obtain the overall score of a variant. We assume that all sources of evidence give the probability of an error of the corresponding type (misrecognition of a letter, misplacing the name in the map, etc.), and that such errors are independent. Determining such probabilities is discussed in detail in [15].
4. The variant with the best score S_1 is considered.
5. If this best variant is good enough ($S_1 \geq \alpha$; α is a user-defined threshold), then:
   5.1 If the score of the best variant significantly differs from the score of the second best one ($S_1 / S_2 > \beta$, a user-defined threshold) then this variant is accepted and is added to the database together with its associated information.
   5.2 Otherwise, human intervention is requested, and the variants are presented to the operator in the order of their scores.
6. Otherwise ($S_1 < \alpha$), no correction is applied to the recognized string. It is checked against the linguistic restrictions of a given language, see Section 8.
   6.1 If no anomalies are found, it is considered a new toponym absent in our dictionary and is added to the database as is and is associated with a nearby object using an algorithm not discussed here.
   6.2 If an anomaly is found, the string is considered not recognized and human intervention is requested.
7. After all strings have been recognized, global check is performed, see Section 9.

Note that when the different sources of evidence are combined, they are taken with user-defined weights depending on the quality of the map, the reliability of the basic OCR procedure, etc. We here do not discuss the choice of these weights.

In the following sections, we will consider each source of evidence used in the presented algorithm, as well as the global check procedure.

4 Textual Information

We suppose that there is available a list (dictionary) D of toponyms that can be found in a map. The list can contain much more toponyms than the map in hand - for example, all cities of the country, all seas of the world, etc. Such a list can be compiled as a combination of different sources such as governmental statistical databases, police databases, analysis of newspapers available in Internet, etc.

For a given string s, e.g., *RNSoSIA*, a set of all strings similar to s in the dictionary D can be easily constructed [10]. By a similar string, a string s' is considered that differs from s in at most a certain number of the following disturbances: (1) substitution of a letter for another letter, (2) omission of a letter, and (3) insertion of a letter. With each such disturbance, a probability can be associated; in case of several disturbances, the corresponding probabilities are multiplied to obtain the overall probability of that s (*RNSoSIA*) was obtained from s' (say, *RUSSIA*) by this sequence of errors. For the string itself ($s' = s$ if it is present in D), the probability is 1.

The probabilities of the disturbances can depend on the specific letters involved. E.g., the probability of substitution of *I* for *J* is higher than *W* for *L*. Similarly, the

probability of omission of I is higher than that of M. In a cartographic map, the probability of insertion of o is high because of the notation for cities.

We do not discuss here in detail the issue of automatically learning the corresponding probabilities. If the map is large or of standard type and quality, a statistical model can be trained by processing a part of this or another map of similar quality and manually verifying the results. Or, the iterative procedure described in Section 10 can be used to automatically adjust the model to the specific map.

5 Spatial Letter Distribution Information

As we have mentioned, the basic OCR procedure returns the coordinates of each letter. This can give us two characteristics of the recognized string:

- Whether the letters are aligned along a straight line,
- The distance between each adjacent pair of letters.

Only the names of some linear and area objects (e.g., rivers or lakes), but not punctual objects (e.g., cities), should have non-linear letter alignment. Note that the information on the type of the object for a specific variant of error correction as described in Section 4 is available from the dictionary. If the string is not found in the dictionary and is associated with a nearby object in the map (see step 6 of the algorithm, Section 3), again, the type of the object is known. Note that non-linear alignment is admitted for non-punctual objects but not required.

The distance between adjacent letters gives information on the probability of insertion or deletion-type error. Deletion-type error (a letter is to be inserted to obtain a valid word) is probable if the distance between the two neighboring letters is about twice larger than the average distance between the letters in the string (it can be the space between words, too). Similarly, insertion type error (a letter is to be deleted from the string to obtain a valid word) is probable if the mean distance between the letter in question and its neighboring letters is about twice smaller than the average. Note that in these cases the corresponding correction of the word is not only acceptable but also required: the score of a string with this type of defects is decreased.

6 Geographic Information

When the original string or a spelling correction candidate string is found in the dictionary,[2] the dictionary can provide at least two types of spatial information on the corresponding object:

- Its inclusion in a larger area, such as province, state, etc. Areas form a hierarchy.
- Its geographic coordinates.

[2] Such geographical databases are available from the corresponding governmental organizations. For example, in Mexico large geographical databases are distributed by INEGI, the National Institute for Statistics, Geography and Informatics, see www.inegi.gob.mx.

This information can be used to filter out the candidates that are very close as to their spelling to the original string returned by the basic OCR procedure but are not located in the area in question. E.g., suppose the OCR procedure returned the string *Xalapa* in the area of Mexican State of Oaxaca. Such a string indeed exists in the list of Mexican cities, but the corresponding city is in the state of Veracruz. On the other hand, there is a city *Jalapa* precisely in the state of Oaxaca. With this, it should be considered more probable that the string *Xalapa* was a result of a recognition error and that the correct string is a similar string *Jalapa*.

Moreover, very frequently the dictionary contains several objects with the same name, of the same or different type. When analyzing a map of Canada, the object corresponding to a recognized string *London* is to be a small Canadian city and not the large British city, so that the correct number of inhabitants for the object could be imported from the dictionary to the database being constructed. When analyzing an inscription *Moscow* in the coordinates (57° N, 35° E), its interpretation as a river rather than city is more probable.

If only the hierarchical information is available ("Jalapa city is in Oaxaca state"), it can be used to filter out undesirable variants only if the coordinates are available for one of larger areas one or more steps up the hierarchy (but small enough to serve for disambiguation). Alternatively, it might happen that the corresponding larger area has been earlier recognized in the same map. However, due to the order of recognition from smaller to larger objects (see beginning of Section 3), this is hardly probable.

The corresponding check can be performed at the post-processing stage—global verification, see Section 9, when all areas have been recognized.

In the best case, full coordinate information is available in the dictionary for the object. Then the probability for the given inscription to represent the given object can be estimated as $a\exp(-bd^2)$, where b is a coefficient depending on the scale of the map and the fonts used, a is the normalizing coefficient, and d is the distance from the inscription to the object. This distance can be heuristically defined as follows:

- For a punctual object (such as a city) represented by only one coordinate pair p, the inscription is expected to be next to the point p. Thus, we can take d as the minimum distance from the point p to any of the frames containing the individual letters of the string.
- For linear objects (such as rivers) represented by a sequence of coordinate pairs p_i, the inscription is expected to follow the shape of the arc. Thus, we can take d as the average distance from the frames containing each letter to the broken line (or otherwise interpolated arc) defined by the points p_i. To put it in a slightly simplified way, to measure the distance between a letter and the broken line, the two adjacent points p_i, p_{i+1} nearest to the letter are found and the distance from the letter to the straight line connecting the two points is determined.
- For an area object **S** (such as a province) represented by a sequence of coordinate pairs p_i corresponding to its contour, the inscription is expected to be in the middle of the area and the letters are expected to be distributed by the whole area. Thus, we can take $d = \iint_{\mathbf{S}'} f(x, y)\,dxdy$, where $f(x, y)$ is the minimum distance

from the point (x,y) to any of the letters of the string. The integral is taken over the intersection $\mathbf{S}'$ of the area **S** and the whole area of the given map (in case a part of **S** proves to be out of the boundaries of the given map). Note that a similar integral along the contour would not give the desired effect. Since the number of candidates generated at the step 2 of the algorithm from Section 3 is rather small, and the area objects are much less numerous than other types of the objects in the map, we do not consider computational efficiency a major issue for our purposes. Neither precision is important for us. Thus, it is enough to compute the integral by, say, Monte-Carlo method.

In fact, the coefficient b (and thus a) in the formula above is different for these three cases. We do not discuss here selection of this scaling coefficient. More details on the quantitative evaluation of the corresponding probabilities can be found in [15].

7 Notational Information

Notation in the map can give additional information to filter out undesirable variants. In some maps, rivers are explicitly marked as "*river*" or "*r.*" and similarly mountains, peninsulas, etc. Specific font type, size, and color are usually associated with various types of objects (e.g., cities, and rivers). Though this information can provide very good filtering, it is not standard and is to be manually specified for each individual map, which limits the usefulness of such filtering capability in a practical application.

The practical system should provide the operator the means to specify such notational elements, at least the prefixes such as "*river.*" They are important for the comparison of the string with the dictionary. Indeed, for the string "*river Thames*" what is to be looked up in the dictionary is "*Thames*" and not "*river Thames*". Alternatively, such prefixes can be detected automatically in a large map:

- For each string consisting of several words, both the complete variant and the variants without the first or last word are to be tried.
- If for a specific type of objects (e.g., rivers) in most cases the string is found after taking of the word "river," then this is to be considered as notation for this type of objects.

Similarly the font features for a specific type of objects can be automatically learnt from a large map.

Finally, some precautions should be taken with such type of filters. For example, in Spanish rivers are marked as "*río*" 'river'; however the string *Río de Janeiro* should not be filtered out as a name of the city (if capital letters are not properly distinguished in the map).

8 Linguistic Information

The checks described in this Section are applied only to the strings not found in the dictionary for which the dictionary-based correction failed (no suitable similar string

was found in the dictionary), see Step 6 of the algorithm from Section 3. In this case, general properties of the given language can be used to detect (though not correct) a possible recognition error.

One of simple but efficient techniques of such verification is bigram (or trigram) control [12]. In many languages, not any pair (or triplet) of letters can appear in adjacent positions of a valid word. For example, in Spanish no consonant except *r* and *l* can be repeated; after *q* no other letter than *u* can appear, etc. The statistics of such bigrams (or trigrams) is easy to learn from a large corpus of texts. The multiplication of the bigram frequencies for each adjacent pair of letters in the word (and similarly for trigrams) gives a measure of its well-formedness, which can be compared with a user-defined threshold; if a bigram not used at all in the given language appears, the word is immediately marked as probably incorrect.

Other properties of words specific to a given language can be verified: e.g., in Japanese all syllables are open.

If a recognized string for which no variants of correction by the dictionary are found does not pass any of the linguistic filters, it is presented to the human operator for possible correction.

Note that since toponyms are frequently words of another language (say, indigenous languages) or proper names of foreign origin, linguistic verification can produce a large number of false alarms.

9 Global Constraint Verification

After all inscriptions in the map have been recognized, some global constraints should be checked.

Uniqueness. To each object, only one inscription should correspond. If two inscriptions have been associated with the same object, one or both of them is to be reassigned. Even though the information on the probability of each of the two candidates is available at this point and could allow for automatic selection of one of the candidates, we believe that such conflicts should not be arbitrated automatically but the human intervention is to be requested instead. Of course, the probability information can be used to suggest the most likely variant to the human operator.

An exception from this rule is linear objects such as long rivers. Several inscriptions can be assigned to such object provided that their text is the same, the distance between them is much larger than their lengths, and their length are much smaller than the length of the object (river).

Inclusion. The hierarchical information available from the dictionary (see Section 6) can be applied at this point. Recall that our algorithm recognizes the names of, say, cities before those of areas. So at the time of recognition of the string "*Xalapa*" the information "*Xalapa City is in Veracruz State*" could be checked since we did not know yet where Veracruz State is in the map. Now that all strings have been recognized, this information can be checked (now we know where *Veracruz* is) and the error discussed in Section 6 (*Xalapa* mistaken for *Jalapa* recognized in Oaxaca

State) can be detected. In this case, again, human intervention can be requested. Alternatively, the process of error correction can be repeated for this string, and then the global verification is repeated for the objects involved in the resulting changes.

Conclusion

In this work we focused on maps with texts (there are many maps with numerical labels, such as elevations, geographical coordinates, and so on; see [1], [7], and [8] for discussion on this type of maps).

We have shown that the problem of recognition of inscriptions in the map, assigning them as names to specific objects (e.g., cities), and importing—using these names as keys—properties of these objects (e.g., population) from existing databases involves both traditional techniques of image recognition and methods specific for cartographic maps processing. Our algorithm combines various sources of evidence, including geographic coordinates and object inclusion hierarchy, to choose the best candidate for error detection and correction.

One obvious direction of future development is refining the heuristics used in the discussed sources of evidence and adding new sources of evidence. For example, the basic recognition procedure can return the probability (the degree of certainness) of each letter in the string, or even a list of possible letters at the given position in the string along with their respective probabilities. The idea is that if the basic recognition procedure is certain that the letter in question is exactly the one it recognized (as opposed to just looking like this), the letter should not be changed in error correction, and vice versa.

Another issue possibly to be addressed in the future is the computational complexity, especially the method used to compute the integral in Section 6.

Yet another possible modification of the algorithm is an attempt to recognize all strings before error detection and correction. In many cases this can allow to apply the hierarchical information during the main cycle of the algorithm and not at the stage of post-processing, see the discussion in the item 2 in Section 9 and [14]. This seems to be most promising in the context of our approach. In fact, the measures of similarity and dissimilarity ("distances") between hierarchical variables proposed in [14] can be used as additional source of evidence to assign correct name to the geographical location under consideration. We discuss this assignment procedure in [15].

However, the most important direction of future research is automatic training of the statistical models, automatic learning of the notational information, and automatic determination of the parameters used in various heuristics of our method.

The parameters could be adjusted by application of the algorithm iteratively while varying the parameters, say, according to the gradient descending method. For this, however, an automatically computable measure of the quality of the result is to be determined. The training of the statistical model and learning of the notation can be done using an iterative re-estimation procedure [10].

Acknowledgments

The work was partially supported by Mexican Government (CONACYT, SNI, CGPI-IPN) and the ITRI of the Chung-Ang University. The first author is currently on Sabbatical leave at the Chung-Ang University. Third author is corresponding author.

References

1. Levachkine, S., Velázquez, A., Alexandrov, V., Kharinov, M.: Semantic Analysis and Recognition of Raster-scanned Color Cartographic Images. Lecture Notes in Computer Science, Vol. 2390. Springer-Verlag, Berlin Heidelberg New York (2002) 178-189
2. Doermann, D.S.: An Introduction to Vectorization and Segmentation. Lecture Notes in Computer Science, Vol. 1389. Springer-Verlag, Berlin Heidelberg New York (1998) 1-8
3. Nagy, G.: Twenty Years of Document Image Analysis in PAMI. PAMI. Vol. 22, No. 1. (2000) 38-62
4. Ganesan, A.: Integration of Surveying and Cadastral GIS: From Field-to-fabric & Land Records-to-fabric. Proc. 22nd ESRI User Conference, 7-12 July, 2002, Redlands, CA, USA (2002), see gis.esri.com/library/userconf/proc02/ abstracts/a0868.html.
5. Fletcher, L.A., Kasturi, R.: A Robust Algorithm for Text String Separation from Mixed Text/Graphics Images. PAMI. Vol. 10, No. 6. (1988) 910-918
6. Tan C.L., Ng, P.O.: Text Extraction using Pyramid. PR. Vol. 31, No. 1 (1998) 63-72
7. Velázquez, A.: Localización, Recuperación e Identificación de la Capa de Caracteres contenida en los Planos Cartográficos. *Ph.D. Thesis*. Centre for Computing Research-IPN. Mexico City, Mexico (2002) (in Spanish)
8. Velázquez, A., Levachkine, S.: Text/Graphics Separation in Raster-scanned Color Cartographic Maps. In: Levachkine, S., *et al*. (eds.): Proc. International Workshop on Semantic Processing of Spatial Data (GEOPRO 2002), 3-4 December 2002, Mexico City (2002)
9. Cao, R., Tan, C.L.: Text/Graphics Separation in Maps. Lecture Notes in Computer Science. Vol. 2390, Springer-Verlag, Berlin Heidelberg New York (2002) 168-177
10. Gelbukh, A.: Syntactic Disambiguation with Weighted Extended Subcategorization Frames. Proc. Pacific Association for Computational Linguistics (PACLING 1999), 25-28 August, 1999, Canada (1999) 244–249
11. Gelbukh, A.: Alexander Gelbukh. Exact and approximate prefix search under access locality requirements for morphological analysis and spelling correction. Computación y Sistemas, vol. 6, N 3 (2003) 167–182, see www.gelbukh.com/CV/Publications/ 2001/ CyS-2001-Morph.htm.
12. Angell, R.C., Freund, G.E., Willett, P.: Automatic Spelling Correction using a Trigram Similarity Measure. Inf. Processing & Management. Vol. 19, No. 4. (1983) 255-261
13. Hirst, G., Budanitsky, A.: Correcting Real-Word Spelling Errors by Restoring Lexical Cohesion. Natural Language Engineering, 2004 (to appear)
14. Levachkine, S., Guzman, A.: Hierarchies as a New Data Type for Qualitative Variables. Data Knowledge Engineering (DKE) (to appear)
15. Gelbukh, A., Levachkine, S., Sang Yong Han: Using Sources of Evidence to Resolve Ambiguities in Toponym Recognition in Cartographic Maps. In: Levachkine, S., *et al*. (eds.), Proc. International Workshop on Semantic Processing of Spatial Data (GEOPRO 2003), 4-5 November 2003, Mexico City, Mexico (2003)