CAR-TR-747
CS-TR-3386

IRI-9017393
NGT-30130
December 1994

MAGELLAN: Map Acquisition of GEographic Labels by Legend ANalysis

Hanan Samet
Aya Soffer

Computer Science Department and
Center for Automation Research and
Institute for Advanced Computer Studies
University of Maryland
College Park, MD 20742

Abstract

A system named MAGELLAN (Map Acquisition of GEographic Labels by Legend ANalysis) is described that utilizes the symbolic knowledge found in the legend of the map to drive geographic symbol (or label) recognition. MAGELLAN's output serves as input to a geographic information system (GIS). MAGELLAN first scans the geographic symbol layer(s) of the map. The legend of the map is located and segmented. The geographic symbols (i.e., labels) are identified, and their semantic meaning is attached to them. An initial training set library is constructed based on this information. The training set library is subsequently used to classify geographic symbols in input maps using statistical pattern recognition. User interaction is required at first to assist in constructing the training set library to account for variability in the symbols. The library is built dynamically by entering only instances that add information to it. The library is stored in an appropriate spatial data structure, and a highly efficient nearest neighbor finding algorithm is used to search it. MAGELLAN then proceeds to identify the geographic symbols in the input maps automatically. MAGELLAN can be fine-tuned by the user to suit specific needs. An experimental study was conducted on a large set of data and recognition rates of over 93% were achieved.

Key terms: Map recognition, Document analysis, Object recognition, Geographic Information Systems

The support of the National Science Foundation under Grant IRI-9017393, and the National Aeronautics and Space Administration under Grant NGT-30130, is gratefully acknowledged, as is the help of Sandy German in preparing this paper.

1 Introduction

The paper map has long been the traditional representation of spatial data. Today, we are seeing the emergence of geographic information systems (GIS) as a replacement. One of the central issues in GIS is data acquisition. In particular, we must find ways of converting the data that is stored on maps to a representation that is compatible with the GIS. The most common means of performing this conversion is by use of a digitizing tablet. This process is very time consuming, expensive, and inaccurate. Recently, optical scanners have been put to use for this purpose. Once the maps have been scanned, the raster data is converted into vector format with very heavy user intervention in order to assure the quality of the conversion [17].

There has been some research in recent years on automating this process. Most researchers have focussed on raster-to-vector conversion [12]. Unfortunately, this does not yield accurate and useful results. One of the reasons for the poor performance is that the conversion must be based on some knowledge, which we term *map recognition*. The need for map recognition results from the fact that a paper map is an abstraction. The information found in maps is mainly symbolic rather than an accurate graphical description of the region covered by the map. For example, the width of a line representing a road has little to do with its true width. Instead, most often, it is determined by the nature of the road (highway, freeway, rural road, etc.). The color and size of a city's name on the map convey information about the population of the city. Many graphic symbols are used to indicate the locations of sites such as hospitals, post offices, recreation areas, scenic areas, etc. The key to this symbolic information may be found on the map itself in the map's legend.

In this paper we describe a system called MAGELLAN (Map Acquisition of GEographic Labels by Legend ANalysis) that uses the legend of the map to drive geographic symbol (or label) recognition. MAGELLAN first locates the legend of the map and segments it. The geographic symbols are identified, and their semantic meanings are attached to them. An initial training set library is constructed based on this information. The training set library is subsequently used to classify geographic symbols in input maps using statistical pattern recognition [7]. User interaction is required at first to assist in constructing the training set library to account for variability in the symbols. Subsequently, MAGELLAN proceeds to identify the geographic symbols in the input maps automatically.

Another problem in map recognition is the high level of occlusion of geographic symbols as a result of the map-making process. A map is composed of several layers. The symbols in each layer do not occlude each other. However, once these layers are composed, the objects from the different layers may intersect and occlude each other making the problem of segmentation and object recognition very complex. To alleviate this problem, we have developed a *layered approach* to map recognition. The inputs to MAGELLAN are raster images of the separate map layers. These layers may not always be as readily available as integrated layer maps, but the extra work required to get them is well worth the effort. The results of map recognition will often be an order of magnitude better than those that would result from using the composite map. Thus much less human time will be required to verify and correct the results of the automatic process.

Most of the prior research in map recognition has concentrated on skeletonization and vectorization methods [1, 19]. Some research has been done on separating the layers of scanned maps [18, 20]. These maps included road maps [9] and cadastral maps [5, 10]. In the latter, the focus was on locating polygons representing parcels of land, buildings, and roads. As mentioned above, the emphasis of our approach is on utilizing the legend of the map to build a system to extract the symbolic information in the map layers, rather than trying to vectorize the entire map and interpret every object found in it. Our application involved tourist symbols such as campsites, hotels, recreation

areas, etc., but we have also used our methods to handle other types of graphical documents such as floor plans [15].

The rest of this paper is organized as follows. Section 2 outlines the main components of MAGELLAN. Section 3 describes the legend-driven training of MAGELLAN. Section 4 discusses the geographic symbol classification process. Section 5 presents our evaluation method along with experimental results. Section 6 contains concluding remarks.

2 MAGELLAN System Overview

The input to MAGELLAN is a raster image of the symbols layer. This map image is divided into *tiles* of size 512×512 pixels (Figure 1). These tiles are processed one-by-one. MAGELLAN (Figure 2) has two phases, the legend acquisition phase and the symbol classification phase, corresponding to the processing of legend and non-legend tiles, respectively.

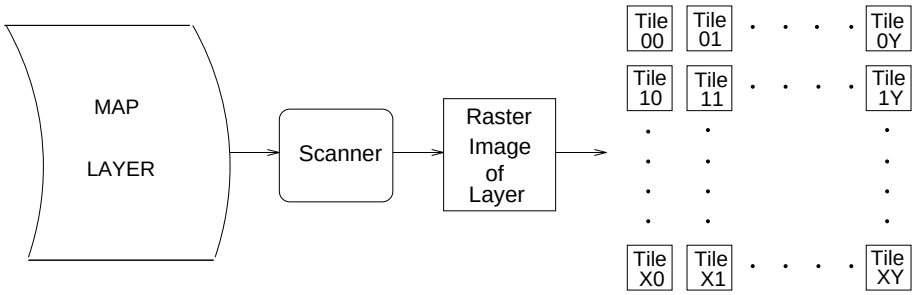

Figure 1: Map layer acquisition and splitting process.

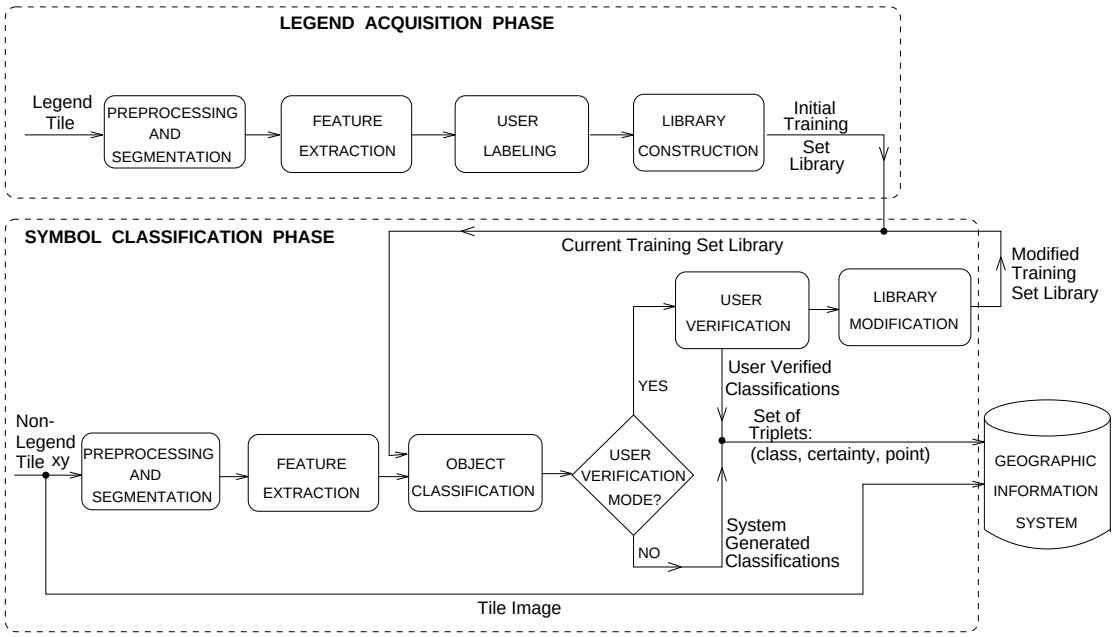

Figure 2: Block diagram of MAGELLAN (Map Acquisition of GEographic Labels by Legend ANalysis)

2.1 Legend Acquisition Phase

The purpose of the legend acquisition phase is two-fold. The first is for the user to indicate which symbols of the legend are of importance to the application. These symbols are termed *valid symbols*. Any other symbols that are found in map tiles but were not pointed out by the user at this stage are termed *invalid symbols*. The second purpose of the legend acquisition phase is to construct an initial training set library that is subsequently used in the symbol classification phase. This initial training set library contains a feature vector corresponding to one instance of each valid symbol along with its semantic meaning (also termed *classification*). This is the classification that MAGELLAN should output for each instance of this valid symbol that is subsequently input into it. For invalid symbols, a special classification termed *undefined* is used (i.e., invalid symbols should be classified as undefined by MAGELLAN). All other classifications (i.e., those that correspond to valid symbols) are termed *valid classifications*.

The legend acquisition phase is initiated by identifying the tile(s) containing the legend. These tile(s) serve as the input to the legend acquisition phase. Next, the legend tile(s) are segmented. A feature vector is computed for each connected component. The user identifies those symbols that are found in the legend that are of importance to the application. The semantic meaning (*classification*) is attached to the feature vectors corresponding to these symbols. An initial training set library is constructed containing one instance of each valid symbol. This entire process is done manually at present. Automating parts of this process such as locating the legend and deriving the semantic meanings of the symbols from the legend is a subject for future research.

2.2 Symbol Classification Phase

Each non-legend tile is input into the symbol classification phase. This phase may operate in two modes. In the *user verification mode*, the user verifies the classifications before they are input to the GIS. The training set is modified to correct the erroneous classifications. In the *automatic mode*, the classifications are generated by the phase and input directly to the GIS. The user determines the mode in which the phase operates. In general, the first tiles will be interpreted in user verification mode. Once the user is satisfied with the recognition rate achieved, the phase is placed in automatic mode to process the remaining tiles. There are five basic modules in this phase.

2.2.1 Preprocessing and Segmentation Module

In the preprocessing and segmentation module, various image processing techniques are applied to enhance the image. These may include applying noise reduction filters, edge closing, thinning, etc. The image is then segmented into its constituent elements using a connected component labeling algorithm (e.g., [13]). The output of this module is a labeled image in which each pixel has a region number as its value. Regions that are smaller than a certain threshold are labeled 0.

2.2.2 Feature Extraction Module

The input to this module is the labeled image output by the preprocessing and segmentation module. For each region in the labeled image, a set of *features* is computed. MAGELLAN uses some global (e.g., first invariant moment, circularity, eccentricity) and some local shape descriptors (e.g., intersections, gaps) [11] that we have identified as features that best discriminate between geographic symbols. These features are invariant to scale, orientation, and translation. The results of the feature computation are composed into a *feature vector*. The center of gravity (i.e., centroid)

of each region is also computed. The x and y coordinate values of this location are termed a *location vector*. The output of this stage is a *feature descriptor* that is composed of the feature vector which is a point in the n-dimensional *feature space*, and the location vector which is a point in the two-dimensional *location space*.

2.2.3 Object Classification Module

The input to this module is the feature descriptor (feature and location vectors) output by the feature extraction module, the current training set library, and some parameters set by the user. The training set library, constructed by the legend acquisition phase, initially consists of one feature vector for each geographic symbol along with its semantic meaning (i.e., its *classification*). This is the class that MAGELLAN should assign to each instance of the same symbol. Depending on the quality of the raster image, this may suffice. However, in most cases, the instances of each symbol do vary and thus several instances of each symbol are required in order to build a representative training set library that will produce reasonable recognition rates. The current training set library is used to assign candidate classifications to each input feature vector. A value approximating the certainty of the correctness of these classifications is attached to each classified object (see Section 4 for more details). The output of this module consists of the classifications that were made, the certainties of the classifications, and the locations of the corresponding symbols on the map. In terms of a GIS, the output is point data where the classifications and certainty values are alphanumeric attributes of the point.

2.2.4 User Verification Module

This module is only active when MAGELLAN is operating in user verification mode. The input to this module consists of the feature descriptors and the classifications obtained from the object classification module. An image containing the classifications having the highest certainty value for each symbol in the input image is composed. This image is displayed next to the original input image. The user, based on visual inspection, indicates which symbols have been classified incorrectly, and informs MAGELLAN of the proper classification for each such symbol. This module outputs the location, classification (as approved or corrected by the user), and a certainty of 1 for all the symbols found in the input map tile. This information is passed to the GIS. In addition, this module outputs those feature vectors corresponding to the symbols that MAGELLAN misclassified along with their correct classifications. This information is passed on to the library modification module.

2.2.5 Library Modification Module

This module is only active when MAGELLAN is operating in user verification mode. The input to this module is a list of feature vectors along with their corresponding classifications. These vectors are used to classify subsequent input symbols, and thus they comprise a part of the training set for the classification process. Notice that only feature vectors of symbols that could not be classified correctly using the current training set library are added to the library. Feature vectors of correct classifications are not added to it. This method of dynamically constructing the library ensures that the library will remain small without compromising the results of classifications. The reason for this is that there is no redundant information in the library. Hence it is very efficient and yields results similar to those obtained using a condensing technique to minimize the size of the library [7]. The library is stored as an adaptive k-d tree [8, 14] (see Section 3 for more details about

the storage and retrieval methods employed for managing the library). The output of this module is the current library, which is used by subsequent invocations of the object classification module.

2.3 Geographic Information System

The input to the GIS is a set of points, corresponding to the locations of the symbols found in the maps. For each point, its possible classifications and a certainty value approximating the correctness of each classification are given. This input comes either directly from the object classification module or from the user verification module depending on the mode in which MAGELLAN is operating.

3 Managing the Training Set Library

Realizing that a feature vector is a point in n-dimensional space, we use methods borrowed from computational geometry and spatial data structures to spatially sort the training set library. The data structure that we use is the adaptive k-d tree of Friedman, Bentley, and Finkel [8, 14]. This is a variant of a binary search tree with two pieces of information stored at each node: a dimension number d indicating the discriminating dimension, and a discriminating value v which is usually the median of the coordinate values in dimension d of the set of points stored below this node. The discriminating dimension is chosen to be the dimension for which the spread of coordinate values of that dimension is a maximum. The spread is measured as the distance from the minimum to the maximum coordinate value normalized with respect to the median. All points with coordinate values less than or equal to the discriminating coordinate value are inserted in the left subtree, and all points with coordinate values greater than the discriminating coordinate value are inserted in the right subtree. This process is continued recursively until only a few points are left in the set, at which time the decomposition ceases and the result is termed a *bucket*. The components of the bucket are represented by a linked list. Figure 3 shows an example of some instances of geographic symbols in a two-dimensional feature space (i.e., a feature vector with two components) with the corresponding adaptive 2-d tree.

In order to build an adaptive k-d tree, all points must be known a priori. The training set library in MAGELLAN is built in stages while MAGELLAN is operating in user verification mode, as described in Section 2. Initially, the training set library contains only one instance of each symbol, as output by the legend acquisition phase. Later, for each tile that is interpreted in user verification mode, only those instances of symbols that were not classified correctly by the current training set library are added to it. At this stage, the adaptive k-d tree is reconstructed in order to ensure that it remains balanced. Since MAGELLAN is operating in user verification mode, it is not very fast at this point. Therefore, it is reasonable to spend a little more time to rebuild the adaptive k-d tree thereby ensuring that future classifications will be made as efficiently as possible.

4 Classifying Geographic Symbols

Geographic symbols identified in the map tiles are classified using a modification of the *weighted several-nearest neighbor classifier* [4] termed a *weighted bounded several-nearest neighbor classifier*. This classifier makes use of two constants: α, which is a neighborhood-size factor that determines the search range for nearest neighbors, and β, which is a bound that determines the maximum distance allowed between the feature vector of an input symbol and its several-nearest neighbors in the training set library. This classifier first finds the feature vector F^{L_N} in the training set

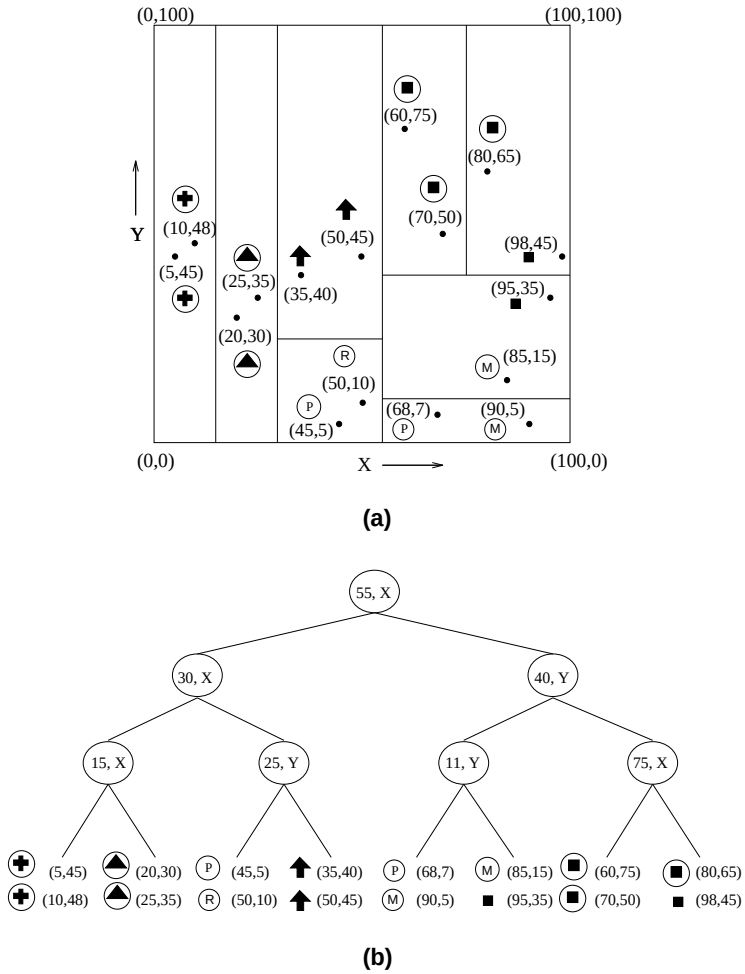

Figure 3: Adaptive k-d tree: (a) Set of symbol instances in 2-space. (b) Corresponding 2-d tree.The 2-space corresponds to two features of the symbol; not the location of the symbol.

library (TSL) that is nearest to the feature vector of the input symbol F^I. Let D be the weighted Euclidean distance between F^{L_N} and F^I given by

$$D = dist(F^{L_N}, F^I) = \sqrt{\sum_{i=1}^{N} w_i (F_i^{L_N} - F_i^I)^2}.$$

where $F_i^{L_N}$ is the i^{th} feature of the training set library vector, F_i^I is the i^{th} feature of the input vector, and w_i is the weighting factor of i^{th} feature of the feature vector. The weighing factor is computed so that features with a smaller variance have a larger weight as described in [6]. Next, the classifier finds the set φ_I^L of all training set library feature vectors F^L whose distance to F^I is less than the smaller of α times D, and β. Formally:

$$\varphi_I^L = \{F^L \mid F^L \in TSL \wedge dist(F^L, F^I) < \min(\alpha \times D, \beta)\}.$$

The range defined by $\min(\alpha \times D, \beta)$ is termed the *$\alpha\beta$-neighborhood*. Each feature vector $F^L \in \varphi_I^L$ is given a vote, whose strength is inversely proportional to its distance from the feature vector of the input symbol F^I, given by

$$Vote_{F^L} = \frac{1}{dist(F^L, F^I)}.$$

The votes of all feature vectors that belong to the same classification C_i are summed giving

$$Votes_{C_i} = \sum_{F^L \in \varphi_I^L \wedge class(F^L)=C_i} Vote_{F^L}.$$

If $\varphi_I^L = \emptyset$ (i.e., the distance to the nearest neighbor was $> \beta$), then the input symbol is classified as undefined. A certainty value between 0 and 1 is computed for each candidate classification C_i found in the $\alpha\beta$-neighborhood of the input feature vector. This value approximates the certainty that the input vector belongs to C_i. The certainty value is calculated by normalizing the value of $Votes_{C_i}$ with respect to some minimal and maximal acceptable values of $Votes_C$ for any of the possible candidate classifications C. The maximal acceptable vote value is determined by selecting a minimal required distance $d_{\min}$, for a "sure" classification (i.e., if $dist(F^L, F^I) < d_{\min}$, then F^I will be assigned the training set library classification corresponding to F^L with certainty 0.999). Hence, the maximal value for $Votes_C$ is $1/d_{\min}$. The minimal acceptable vote value is determined by selecting a maximal allowed distance $d_{\max}$ for a classification to be considered as a candidate (i.e., if $dist(F^L, F^I) > d_{\max}$, then the training set library classification corresponding to F^L will not be considered as a candidate classification for F^I at all). Hence, the minimal value for $Votes_C$ is $1/d_{\max}$. The motivation for calculating the certainty values in this manner is that the certainty values must rank the candidate classifications with respect to one another not only in one invocation of the classifier, but must do so with respect to the candidate classifications of all other invocations of the classifier as well. Therefore, some global method of calculating certainty values is required.

The nearest neighbors of the training set library that are within the $\alpha\beta$-neighborhood are found using a modification of the priority k-d tree search algorithm [3]. This algorithm visits the buckets of the k-d tree in increasing order of their distance from the input feature vector. The search is complete when the distance from the input feature vector to the closest remaining bucket is outside of the range determined by α, and β. The classification module outputs all of the candidate classifications along with their certainty. The classification with the highest certainty value is considered the best classification for the input vector using the weighted bounded several-nearest neighbor classifier.

Figure 4 demonstrates the classification process. It uses the set of symbol instances in 2-space given in Figure 3 as the training set library. Let $X = (34, 31)$ be the feature vector (in 2-space) of an input symbol. Let $\alpha = 2$, $\beta = 11$, and $w_1, w_2 = 1$ (i.e., both features are assigned the same weight in the distance computation). The classifier assigns candidate classifications to X as follows. First, the feature vector F^{L_N} in the training set library (TSL) that is nearest to the feature vector of X (F^I) is found. In this case, $F^{L_N} = (35, 40), D = \sqrt{82} \approx 9.055$. The classifier next finds the set φ_I^L of all library feature vectors F^L whose distance to F^I is less than the smaller of α times D, and β (i.e., in the $\alpha\beta$-neighborhood). In this case, $sizeof(\alpha\beta\text{-neighborhood}) = \min(18.11, 11) = 11$. Thus, $\varphi_I^L = \{(35,40),(25,35)\}$, where (35,40) is an instance of the symbol "arrow" (holiday camp) and (25,35) is an instance of the symbol "triangle" (camping site). $Votes_{arrow} = 1/\sqrt{82} \approx 0.11$, $Votes_{triangle} = 1/\sqrt{97} \approx 0.10$. Thus, X will be assigned classification arrow with a higher certainty than classification triangle. Recall that the certainty value is calculated by normalizing the value of $Votes_{C_i}$ with respect to some minimal and maximal acceptable values of $Votes_C$ for any of the possible candidate classifications C. These values are determined according to $d_{\min}$ and $d_{\max}$, the minimal and maximal acceptable values for $dist(F^L, F^I)$. Thus, to calculate the certainty values for this example we must assign some values to these parameters. Assuming that $d_{\min}$ and $d_{\max}$ are set to $\sqrt{8}$ and 24, respectively, we get $\max(Votes_C) = 1/\sqrt{8} \approx 0.353$ and $\min(Votes_C) = 1/24 \approx 0.042$. Therefore, $certainty(X \in arrow) = 0.218$ and $certainty(X \in triangle) = 0.186$ since we normalized 0.10 and 0.11, respectively.

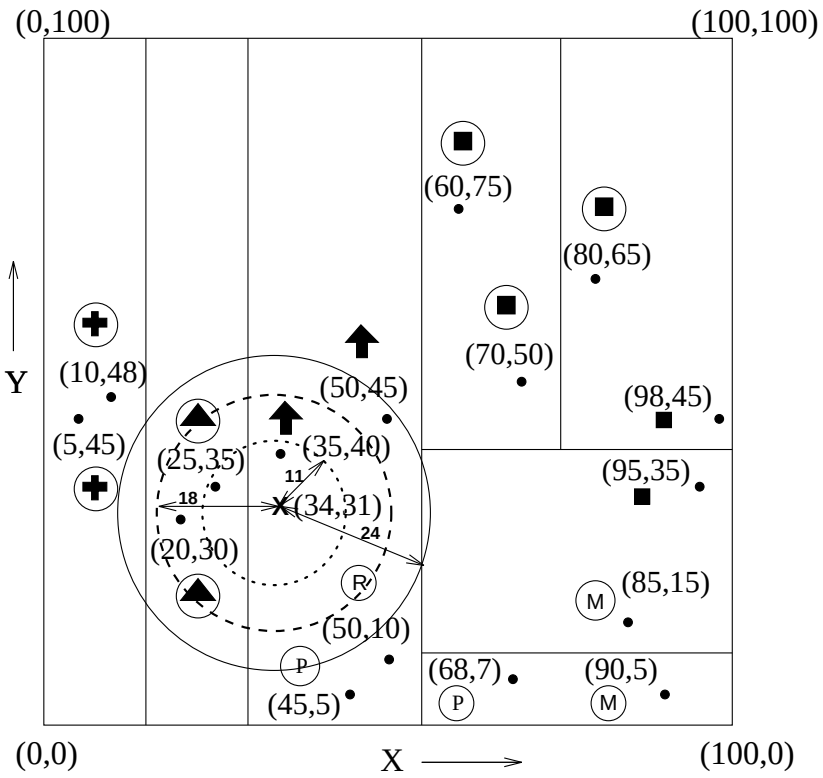

Figure 4: Example of classifying an input symbol X using the training set library given in Figure 3 with a weighted bounded several-nearest neighbor classifier.
$\alpha = 2$, $\beta_1 = 11$, $\beta_2 = 24$. The dotted circle is the $\alpha\beta_1$-neighborhood, the dashed circle is the $\alpha\beta_2$-neighborhood.

However, if we let $\beta = 24$, then $sizeof(\alpha\beta\text{-neighborhood}) = \min(18.11, 24) = 18.11$. Point (20,30) which is another instance of "triangle" will now also be included in φ_I^L. Thus, $Votes_{arrow} = 1/\sqrt{82} \approx 0.11$, $Votes_{triangle} = 1/\sqrt{97} + 1/\sqrt{197} \approx 0.17$. In this case, X will be assigned classification triangle with a higher certainty than classification arrow. Assuming the same $d_{\min}$ and $d_{\max}$, we get $certainty(X \in arrow) = 0.218$ and $certainty(X \in triangle) = 0.411$.

5 Experimental Study

MAGELLAN was tested on the red sign layer of the GT3 map of Finland. The scale of the map is 1:200000. The layer was scanned at 240dpi. Figure 5 is a portion of the map's legend relevant to the sign layer. Figure 6 is a sample tile while Figure 7 shows the extracted red sign layer. Notice that there are many symbols in the map tile that are not found in the legend. These are mainly numbers, names, and markers that are related to other layers. These symbols, termed *invalid symbols*, should all be classified by MAGELLAN as undefined as explained in Section 2.1. The layer was split into 425 tiles of size 512×512. These tiles were examined in a random order to give MAGELLAN a chance to see a large variety of symbols while operating in user verification mode as some symbols tend to appear clustered in the map. The legend was identified manually, and the classifications were attached to the feature vectors representing the geographic symbols that MAGELLAN should identify. There were 22 such symbols. MAGELLAN processed the first 60 tiles in user verification mode. At that stage, the training set contained 100 instances of symbols and the

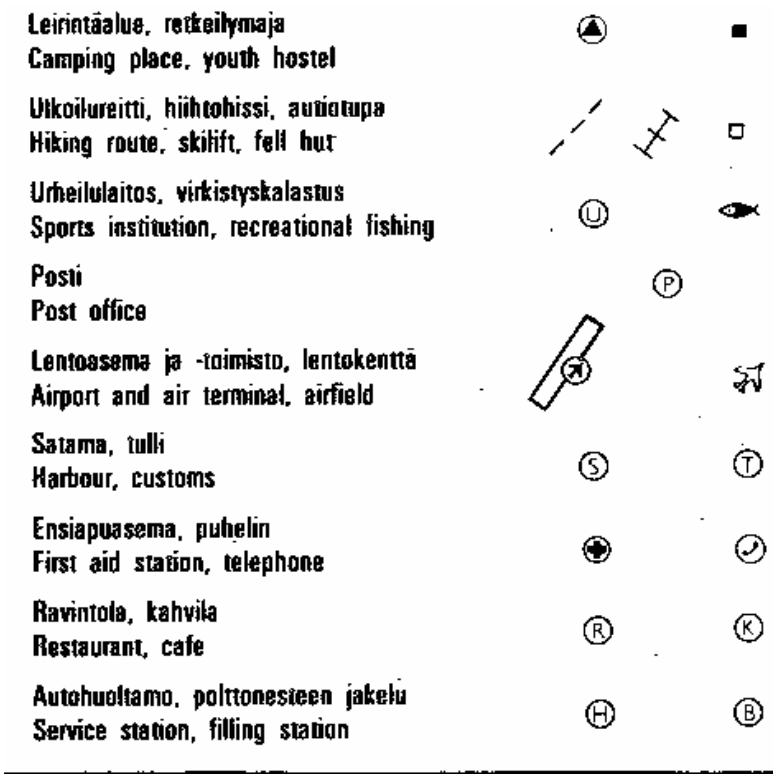

Figure 5: Legend portion containing tourist symbols

current recognition rate was deemed adequate. The remaining tiles were processed automatically. See Section 5.3 for the results of this fully automatic recognition.

The results of this classification were input into SAND [2] (denoting spatial and non-spatial database). It is a home-grown extension to a relational database, developed at the University of Maryland, where the tuples may correspond to geometric entities such as points, lines, polygons, etc. having attributes which may be both of a locational (i.e., spatial) and non-locational nature. Both types of attributes may be designated as indices of the relation. For indices built on locational attributes, SAND makes use of suitable spatial data structures. Attributes of type image are used to store images. Query processing and optimization is performed following the same guidelines of relational databases extended with a suitable cost model for accessing spatial indices and performing spatial operations. Map tiles can be retrieved from SAND according to their contents by means of spatial and non-spatial queries. For example, the user may request to display all tiles containing camping sites within three miles of fishing sites. See [16] for more details.

5.1 Evaluation Method

In order to evaluate MAGELLAN, the following three error categories, common in optical character recognition (OCR) evaluation, were defined:

- *substitution errors* — a valid input symbol was assigned an incorrect valid classification (e.g., picnic site instead of post office).
- *deletion errors* — a valid input symbol was classified as undefined.

Figure 6: Example map tile—all layers

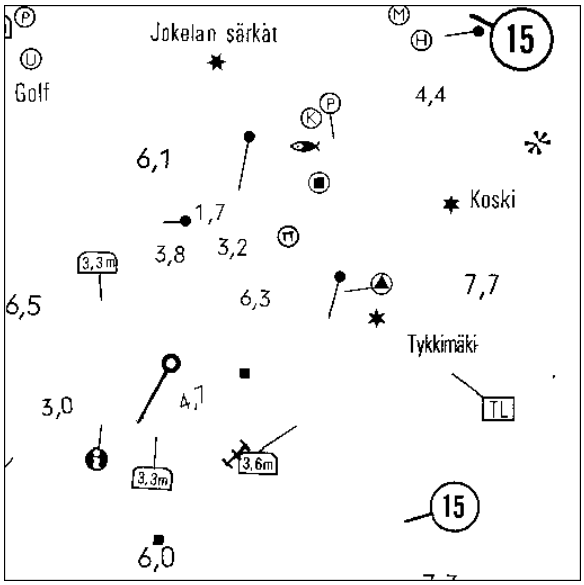

Figure 7: Example map tile—red sign layer

- *insertion errors* — an invalid input symbol was assigned one of the valid classifications rather than being classified as undefined.

Recall that MAGELLAN outputs all of the candidate classifications of the feature vector corresponding to an input symbol that were found in its $\alpha\beta$-neighborhood (see Section 4). As part of our experiment, we were interested in determining whether inserting into the GIS more than one candidate classification per symbol with an appropriate certainty value yields better results than just inserting the class with the highest certainty value. In this case, we insert the same point more than once with different certainty values and classifications. If any of these candidate classifications is the correct one, then this is not counted as a substitution error since the correct classification is in the GIS, although not with the highest certainty value. In order to account for the fact that we now have additional points with erroneous classifications in the GIS, we define an additional error category.

- *addition errors* — an input symbol (either valid or invalid) was assigned more than one valid classification. Each classification after the first one is counted as one addition error.

Substitution errors and deletion errors may cause the GIS to overlook tiles that should be retrieved for a given query. Insertion errors and addition errors may cause the GIS to retrieve superfluous map tiles for a given query. In the context of a GIS or an image database, the impact of insertion and addition errors is not severe. Recall that the purpose of map recognition is to enable the GIS to retrieve just those map portions that are relevant to a given query. Thus, retrieving too many tiles is not as harmful as missing tiles. The user can always weed out those tiles that do not actually conform to the query.

5.2 Experiment Description

50 sample tiles were chosen from the tiles that were not used for training MAGELLAN. These tiles were input into MAGELLAN. For each symbol in each tile, MAGELLAN output all of the candidate

classifications of the neighbors in the $\alpha\beta$-neighborhood of the symbol's feature vector along with a certainty value as described in Section 4. In the first stage of the experiment, only the candidate classification with the highest certainty value was compared to the correct classification as found in the raw image. The number of errors of each type for each tile was recorded. In the next stage, all of the candidate classifications were considered. The number of errors of each type was recorded for two cases. In the first case, only the best and second-best (highest and second-highest certainty values) classifications were considered. In the second case, all classifications were considered. The certainty values assigned by MAGELLAN for each correct classification and incorrect classification were also noted.

This experiment was repeated five times varying the value of β, the maximum distance in the normalized (having unit width) feature space allowed between the feature vector of an input symbol and its neighbors in the training set (termed the *search bound*). Any symbol whose nearest neighbor is not within this search bound is classified as undefined. The values selected for β (the search bound) were $0.02, 0.05, 0.1, 0.2, 0.4$. These values are relative to a search space having unit width. As the search bound increases, more neighbors representing more classifications will be found in the range. Therefore, we expect to have fewer substitution and deletion errors, at the cost of more insertion and addition errors. The value of α (the neighborhood size factor) was set to 2 in all of the experiments. In other words, for each experiment, the classifier considers all of the feature vectors in the training set library whose weighted Euclidean distance from the feature vector of the input symbol is less than the smaller of 2 times the distance to the input vector's nearest neighbor, and the particular value of β for that experiment. The value 2 was chosen empirically as a reasonable value for α. Varying this value will most likely change the experimental results but not the general trend.

5.3 Results of Experiment

Figures 8 and 9 show the recognition rate for valid and invalid symbols, respectively. Recall that valid symbols are those that the user indicated as important to the application in the legend acquisition phase. Any other symbols that are found in map tiles but are not instances of symbols that were pointed out by the user at that stage are invalid symbols. The number of valid and invalid input symbols in the 50 test tiles was recorded. All percentages reported here are of these numbers. The valid symbol recognition rate indicates what percent of the valid input symbols were assigned the correct classification. The invalid symbol recognition rate indicates what percent of the invalid input symbols were in fact classified by MAGELLAN as undefined. The "Highest" plot shows the recognition rate when considering only the classification with the highest certainty value. The "Second" plot shows the rate when considering the classifications with the highest and second-highest certainty value. The "All" plot shows the rate when considering the classifications of all of the neighbors in the $\alpha\beta$-neighborhood regardless of their certainty value.

As expected, as the search bound value increases, the valid symbol recognition rate increases and the invalid symbol recognition rate decreases. The reason for this is that there are potentially more candidate classifications within the $\alpha\beta$-neighborhood when the search bound is larger. Therefore, the chance that a feature vector corresponding to a symbol from the correct classification for a valid input symbol lies in the neighborhood increases. Similarly, the chance that a feature vector corresponding to some valid symbol from the library will lie in the neighborhood of an invalid input symbol also increases. This results in the invalid symbol being assigned a valid classification rather than undefined thereby decreasing the invalid symbol recognition rate. Recall that a symbol is classified as undefined if it has no neighbors in its $\alpha\beta$-neighborhood.

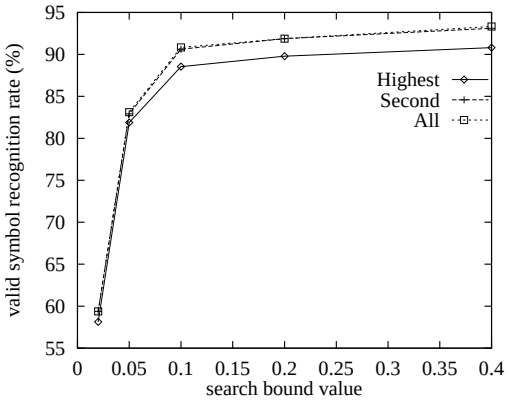

Figure 8: Valid symbol recognition rate for various search bound values.

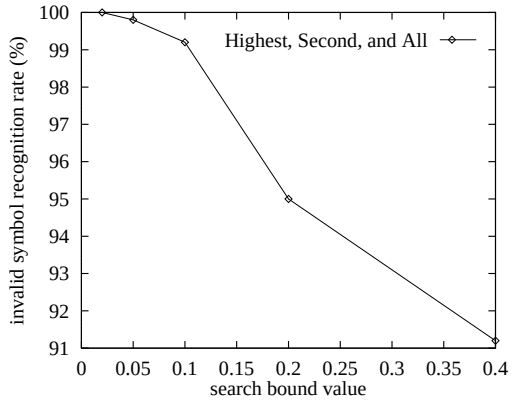

Figure 9: Invalid symbol recognition rate for various search bound values.

From Figure 8 we also see that considering the first two candidate classifications (i.e., those with the highest and second-highest certainty values), and considering all of of the candidate classifications in the neighborhood increases the valid symbol recognition rate. The requirement in this case is only that the correct classification be in the neighborhood regardless of its rank among the candidates. Therefore, more valid symbols will be assigned the correct classification. Notice however that the difference between the valid symbol recognition rate when considering just the first two candidate classification and the valid symbol recognition rate when considering all of the candidate classifications is very small. This means that in almost all of the cases where there was more than one candidate classification and the best classification was incorrect, the second-best classification was the correct one. For the invalid symbol recognition rate, considering only the best or all candidate classifications is irrelevant since an invalid symbol is classified as undefined only if there are no candidate classifications in the neighborhood.

Although considering more than one candidate classification improves the valid symbol recognition rate, it also has a negative side effect. In particular, it introduces addition errors since more than one classification may be recorded in the GIS per symbol. Figure 10 shows the percent of the total number of classifications that were output by MAGELLAN that are attributed to multiple classifications for the same input symbol (i.e., an addition error occurred) for various search bound values. For small search bound values this number is small. However, it grows significantly for larger search bound values, with slightly larger values when considering all candidate classifications rather than just the first two candidate classifications.

Figures 11, 12, and 13 enable us to analyze the causes of the erroneous valid symbol classifications. These figures show the rate of the various error types as a function of the search bound value when considering the classification with the highest certainty value, the classifications with the highest and second-highest certainty values, and all classifications, respectively. Observe that the substitution error rate is only slightly affected by the search bound value, whereas the deletion error rate is highly affected by it. From this observation we may conclude that the increase in the valid symbol recognition rate with an increase in the search bound value is mainly attributed to a sharp reduction in the number of deletion errors instead of a significant change in the number of substitution errors. There were very few cases where a substitution error was corrected by increasing the value of the search bound constant. This can only happen if as a result of the larger $\alpha\beta$-neighborhood, more feature vectors corresponding to instances of the correct classification are found in the neighborhood and they are close enough for their vote to increase the certainty value

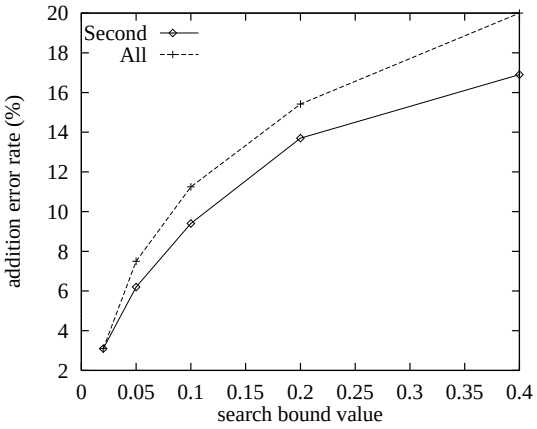

Figure 10: Percent of the classifications output by MAGELLAN that are attributed to multiple classifications for the same symbol (i.e., an addition error occurred) for various search bound values.

significantly. On the other hand, some deletion errors become substitution errors when β is increased. Thus, the substitution error rate actually increases with β. Therefore, the only way to decrease the substitution error rate is to consider more than one of the candidate classifications found among the neighbors in the $\alpha\beta$-neighborhood as can be seen in in Figure 14.

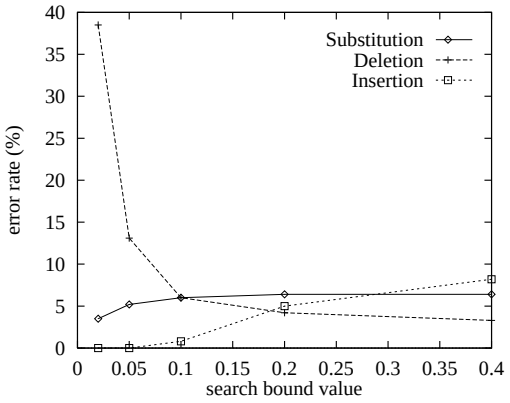

Figure 11: Error rates when considering only the classification with the highest certainty value for various search bound values.

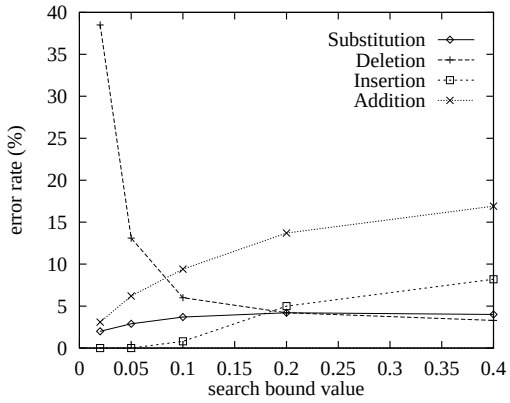

Figure 12: Error rates when considering the highest and second-highest classifications for various search bound values.

The search bound value of 0.1 appears to be best for this data set. A valid symbol recognition rate of 91% and an invalid symbol recognition rate of 99% were achieved with this window size (assuming the classifications with highest and second-highest certainty values in the $\alpha\beta$-neighborhood were taken into account). In addition, 10% of the symbols that were assigned valid classifications were results of multiple classifications for a valid symbol (i.e., an addition error). The ideal search bound value should however be selected according to the requirements of the application. If it is critical for the GIS not to miss any tiles when responding to a query, then a larger search bound value should be selected. This will result in incorporating more insertion and addition errors into the GIS, while ensuring that a minimum number of deletion errors are incorporated into the GIS. This means that the GIS overlooks fewer tiles at the cost of retrieving superfluous tiles that will need to be weeded out manually. If accuracy is not as important as the time required to weed out

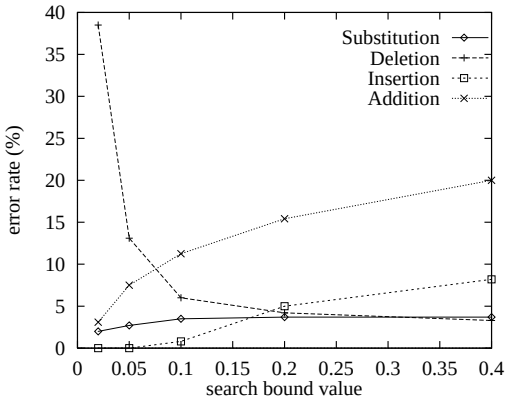

Figure 13: Error rates when considering all candidate classifications for various search bound values.

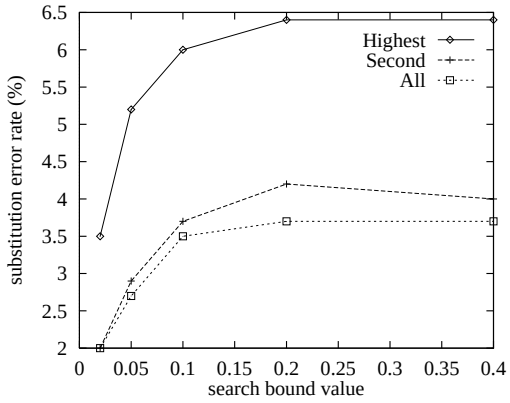

Figure 14: Percent of input valid-symbols assigned the wrong valid classification (i.e., a substitution error occurred) for various search bound values.

the tiles manually, then a smaller search bound value should be selected. As our results indicate, it is possible to achieve valid recognition rates of over 93% with a large enough search bound, and invalid symbol recognition rates of 100% with a small enough search bound. From these results, it is also apparent that only the candidate classifications with the highest and second-highest certainty values should be considered (i.e., transferred to the GIS). The improvement in the valid symbol recognition rate when considering all candidate classifications rather than just the first two was very small, while the addition error rate became larger. Thus, considering all candidate classifications does not seem to be beneficial.

Figure 15 shows the certainty values given to valid input symbols that were assigned the correct classification (i.e., no error occurred). Similarly, Figure 16 shows the certainty values given to invalid input symbols that were assigned some valid classification rather than being classified as undefined, and to valid input symbols that were assigned the incorrect classification (i.e., an erroneous classification due to an insertion or substitution error). The majority (more than 50%) of correct classifications were given a certainty value above 0.9, whereas the certainty values of the erroneous classifications are concentrated at the lower end of the certainty value range. In MAGELLAN, the user may set the minimum certainty value required in order to pass a candidate classification to the GIS. For this data set, selecting a minimum certainty value of 0.3 would result in eliminating many of the substitution and insertion errors, hence automatically weeding out most of the superfluous tiles while overlooking only a small number of the required tiles.

6 Concluding Remarks

A system called MAGELLAN (Map Acquisition of GEographic Labels by Legend ANalysis) has been described. MAGELLAN utilizes the fact that most of the data found in maps is symbolic and that the key to understanding the symbols can be found in the legend of the map. MAGELLAN is efficient and flexible. The training set library is built dynamically by entering only instances that add information to it thereby creating a small but effective training set library. The training set library is stored in an appropriate spatial data structure, and a highly efficient nearest neighbor finding algorithm is used to search it thereby enabling quick classification. Input symbols are classified using a weighted bounded several-nearest neighbor classifier. Users may fine-tune the

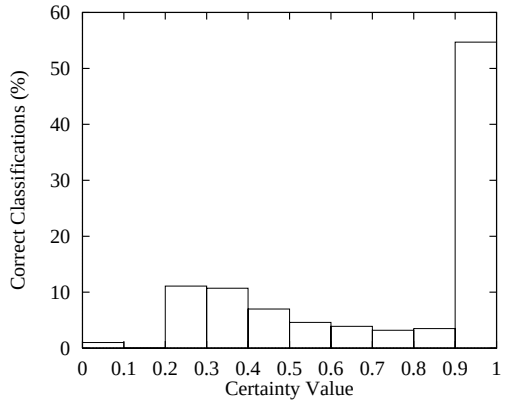

Figure 15: Correct valid symbol classifications per certainty value range.

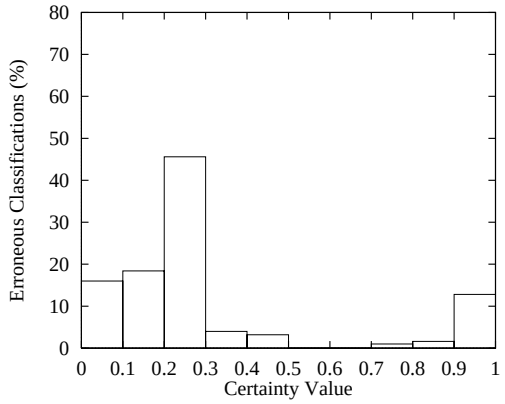

Figure 16: Erroneous classifications per certainty value range.

performance of MAGELLAN to their requirements by setting the search bound value and minimum certainty values to fit their particular applications.

An experimental study was conducted on a large set of data. The experimental results showed that MAGELLAN can achieve recognition rates of 93% with little user intervention, and with more intervention, may reach 100% recognition rates. Although it might seem that simple template matching [11] would have been sufficient here, this is not the case. Different instances of the same symbol may vary in scale and orientation. Therefore, we have chosen to use statistical pattern recognition with features that are invariant to scale, orientation, and translation. At present, the legend acquisition process is mostly manual. Automating it is a subject for future research. MAGELLAN can be easily adapted to interpret other graphical documents and we have used similar methods for the interpretation of floor plans [15]. MAGELLAN is designed for map layers containing geographic symbols. In order to provide full map recognition, other methods need to be developed to interpret layers containing additional types of symbolic information such as roads, bodies of water, etc. Once this is done, the results can be integrated into a GIS to provide a comprehensive tool to utilize the vast amount of data that is found in paper maps.

Acknowledgements

We are grateful to Karttakeskus Map Center, Helsinki, Finland for providing us the map data.

References

[1] S.V. Ablameyko, B.S. Beregov, and A.N. Kryuchkov. Computer-aided cartographical system for map digitizing. In *Proceedings of the Second International Conference on Document Analysis and Recognition.*, pages 115–118, Tsukuba Science City, Japan, October 1993.

[2] W.G. Aref and H. Samet. Optimization strategies for spatial query processing. In G. Lohman, editor, *Proceedings of the Seventeenth International Conference on Very Large Data Bases*, pages 81–90, Barcelona, Spain, September 1991.

[3] S. Arya, D.M. Mount, N.S. Netanyahu, R. Silverman, and A. Wu. An optimal algorithm for approximate nearest neighbor searching. In *Proceedings of the Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 573–582, Arlington, VA., January 1994.

[4] J.L. Blue, G.T. Candela, P.J. Grother, R. Chellappa, and C.L. Wilson. Evaluation of pattern classifiers for fingerprints and OCR applications. *Pattern Recognition*, 27(4):485–501, April 1994.

[5] L. Boatto, V. Consorti, M. Del Buono, S. DiZenzo, V. Eramo, A. Esposito, F. Melcarne, M. Meucci, A. Morelli, M. Mosciatti, S. Searci, and M. Tucci. An interpretation system for land register maps. *Computer*, 25(7):25–33, July 1992.

[6] G.L Cash and M. Hatamian. Optical character recognition by the method of moments. *Computer Vision, Graphics, and Image Processing*, 39(3):291–310, September 1987.

[7] P. Devijver and J. Kittler. *Statistical Pattern Recognition*. Prentice-Hall, Englewood Cliffs, NJ, 1982.

[8] J.H. Friedman, J.L. Bentley, and R.A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software*, 3(3):209–226, September 1977.

[9] M. Ilg. Knowledge-based interpretation of road maps. In *Proceedings of the Fourth International Symposium on Spatial Data Handling*, pages 25–34, Zurich, Switzerland, July 1990.

[10] R.D.T. Janssen, R.P.W. Din, and A.M. Vossepoel. Evaluation method for an automatic map interpretation system for cadastral maps. In *Proceedings of the Second International Conference on Document Analysis and Recognition*, pages 125–128, Tsukuba Science City, Japan, October 1993.

[11] M.D. Levine. *Vision in Man and Machine*. McGraw-Hill, New York, 1982.

[12] D.J. Peuquet. An examination of techniques for reformatting cartographic data part 1: The raster-to-vector process. *Cartographica*, 18(1):34–48, January 1981.

[13] A. Rosenfeld and A.C. Kak. *Digital Picture Processing*. Academic Press, New York, second edition, 1982.

[14] H. Samet. *The Design and Analysis of Spatial Data Structures*. Addison-Wesley, Reading, MA, 1990.

[15] H. Samet and A. Soffer. Automatic interpretation of floor plans using spatial indexing. In S. Impedovo, editor, *Progress in Image Analysis and Processing III*, pages 233–240. World Scientific, Singapore, 1994.

[16] H. Samet and A. Soffer. Integratiing images into a relational database system. Technical Report CS-TR-3371, University of Maryland, College Park, MD, October 1994.

[17] J. Star and J. Estes. *Geographic Information Systems*, chapter 6, pages 85–91. Prentice-Hall, Englewood Cliffs, NJ, 1990.

[18] S. Suzuki and T. Yamada. MARIS: Map recognition input system. *Pattern Recognition*, 23(8):919–933, August 1990.

[19] N. Tanaka, T. Kamimura, and J. Tsukumo. Development of a map vectorization method involving a shape reforming process. In *Proceedings of the Second International Conference on Document Analysis and Recognition*, pages 680–683, Tsukuba Science City, Japan, October 1993.

[20] T. Xu and X. Lin. A new algorithm separating string from map images. In *Proceedings of the Second International Conference on Document Analysis and Recognition*, pages 910–913, Tsukuba Science City, Japan, October 1993.