Label-EZ™ -- Software for Automated Cartographic Text Placement

Herbert Freeman
President, MapText, Inc.
Plainsboro, NJ 08536 USA

Abstract

The most time-consuming task of producing a map - after data collection - is that of labeling the point, line, and area features depicted on it. It is a task that has been performed by skilled human cartographers for centuries but one that has proven remarkably resistant to automation. The paper describes a research and development effort extending over nearly 25 years to develop a software system that would automate this task. The software now available – MapText's Label-EZ – embodies the cartographer's rules and conventions and places the text for each feature according to a user-specified scenario, resolving placement conflicts and ambiguities over all feature layers in accordance with set placement priorities. Placement alternatives are automatically explored if the initial placement choice is not available because of lack of space, in a manner analogous to what a human cartographer could be expected to do under the same circumstances. The paper reviews the issues posed by the text placement problem and describes the solution for it that is now available.

1 Introduction

In the late 1970s, a problem was called to my attention that was said to be defying all attempts at computer solution – the automated labeling of the lines, points, and areas of a map. I was intrigued by the problem and accepted it as a challenge. After a number of years, with the assistance of many graduate students and researchers, we were able to demonstrate that the problem was amenable to solution. A company was formed to develop a commercially viable solution. After years of further development, it became possible to bring a product to market – now in use worldwide – that effectively accomplishes automated cartographic text placement. It reduces the time to label a map from weeks to a matter of seconds. The solution represents the judicious application of techniques drawn from pattern recognition, image processing, and computational geometry to solve a vexing practical problem.

A map is a medium of communication, and the effectiveness with which it communicates spatial information depends in large measure on the quality with which the displayed features are labeled. A map should render the information of interest clearly, rapidly, and without ambiguity. Cartographers have refined the art of map making over hundreds of years and have established an extensive body of cartographic skills, conventions, and quality standards. Any automatic system for text placement must conform to the same conventions and aim at the same level of cartographic quality.

Maps and charts come in a wide range of styles, depending on the purpose for which they are intended. We have city street maps, highway maps, cadastral (property ownership) maps, election district maps, soil maps, forest lease maps, utility maps (e.g., for water, electricity, and gas distribution), telephone coverage maps, nautical charts, and aeronautical charts. In scale these can vary from 1:1,000 for a large-scale local-area map to 1:10,000,000 for one displaying an entire continent.

Maps are used to depict a large variety of different kinds of information – villages, towns, cities, political boundaries, highways, secondary roads, railroad networks, land-use indications, as well as topographic and hydrographic information. The different kinds of information are stored in a computer database and organized as *layers*, which are then assembled (figuratively "overlaid") to form a particular map.

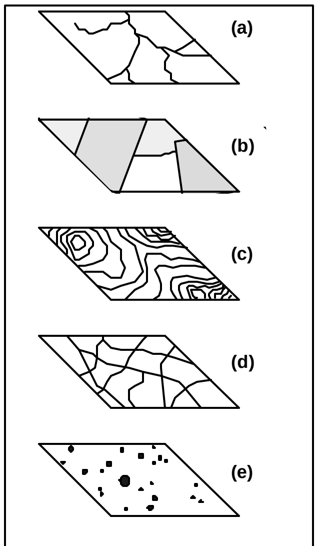

Fig. 1. Map layers

This is illustrated in Fig. 1, where the layers shown represent (a) hydrography, (b) political subdivisions, (c) topography, (d) a highway network, and (e) municipalities. All such information is represented in terms of one of three feature types: *area features* (i.e., polygons representing, lakes, mountain ranges, states, provinces, etc.), *point features* (i.e., point symbols representing small cities, villages, and mountain peaks), and *line features* (i.e., lines and curves representing rivers, highways and railroad lines.) Area features are *scaled*. For line features, only the length is scaled; the width is symbolized (i.e., its value bears no relation to the width of the geographic entity represented and is independent of the map scale.) Point features are represented by symbols and not scaled at all. Note that the choice of feature type is dependent on the scale. Thus a river may be depicted as an area feature on a map of, say, 1:12,000 but appear as a line feature when the scale is much smaller, say, 1:100,000. Similarly a city may be shown as an area feature at the large (1:5,000) scale but convert to a point feature as the scale is reduced.

2 The Text Placement Problem

For a map to be useful, its features must be labeled; that is, text must be placed on or near the features bearing the name of the town, street, state, etc. represented. This seemingly simple task is, in fact, remarkably complex. To communicate well, a map's labeling should be such as to make the spatial relationships among the features easily perceivable and understandable. Toward this objective, cartographers have over centuries developed a host of rules, guidelines and conventions, and any successful automatic system must emulate these.

Map text placement would be much simpler if it could be done once for each layer and then permanently stored with the layer. Unfortunately this is not possible. As different combinations of layers are assembled, the labels from one layer may overlap those from another, or there may be ambiguities of reference among labels and features derived from different layers. Making name placements that would be proper for an assembly that uses all possible layers and then assigning the labels to their respective layers is also not feasible. The placing of labels in a map is just as much influenced by the presence of other names and features as it is by their absence. The proper placement of labels for some layer *A* would be different if the map were or were not also to contain layer *B*.

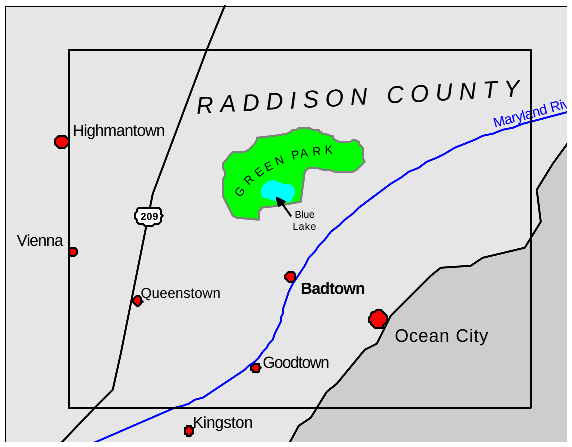

Fig. 2. Text placement must be deferred until after extents are fixed.

The labeling of a map thus must be deferred until all desired layers have been assembled, all features properly symbolized (line width, line color, fill color, etc.), the scale fixed, and the extents decided upon. That is, text placement must necessarily be the *last step* in map preparation. This is demonstrated by Fig. 2.

Let us assume that the inner rectangle in Fig. 2 represents the desired map extent, and the outer border represents the extent of the available data. The figure shows what would occur if features were labeled before the final map extent is chosen. If labeling had been done for the full map, then when the inner extent were selected, the name "Highmantown" would be included in the map even though the town would lie outside. For "Vienna", the opposite would occur: the name would be outside but the town would be inside, and for the line feature "Maryland River" the label would be cut in half.

It is self-evident that the placement of text must necessarily be different for area, line and point features, and the first step in text placement is to separate the features into *feature classes* according to type. This is normally, however, not sufficient. Even for the same feature type, different placement preferences are likely to apply. Thus a user may wish to place the text for state highways in a different manner than, say, for hiking trails. The first step in text placement then is to form feature classes containing features of the same type and such that all the features in the class are to have their text placed in precisely the same manner and with the same priority relative to the labeling of features from other classes.

3 Text Placement Rules

What are the rules for text placement? An excellent compendium of text placement rules has been presented by Imhof [1]. Applicable to all features is the requirement that no label overlap another and that labels not overlap point features. Of equal importance is the requirement that there be an unambiguous association between a label and the feature to which it refers. Beyond these basic requirements is the desire for easy readability, easy comprehension of spatial relationships, and adherence to accepted cartographic conventions. Cartographers tend to place a lot of information in a map and there is severe competition for space. The problem of spatial conflict resolution, deciding how to allocate space and what compromises need to be made to optimize the overall quality of the map is one of the main challenges of cartographic text placement.

Area Features

For area features, the text should be placed inside and ideally – if the feature is sufficiently large – be spread out and curved to conform to the shape and extent of the area. This is illustrated by the area feature "Green Park" in Fig. 2. If the area is too small to accommodate the text, one solution is to *leader* the text from outside, as illustrated by "Blue Lake" in the same figure. An alternative to leadering is *keynumbering*, where a sequential number is placed in the area and the number, together with the area name is then listed in a table somewhere on the map sheet.

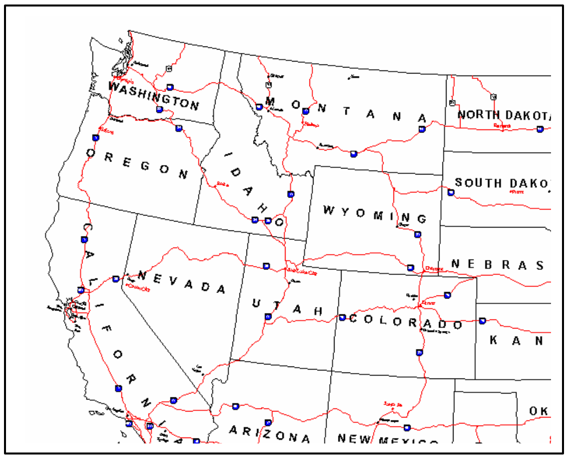

Fig. 3. Names placed to conform to curved constant-latitude lines.

For maps in which area features predominate (e.g., maps showing the counties of a state) it may be preferred to place the text in a straight line, horizontally if possible or angled if space does not permit horizontal placement. In the case of a large-scale map, the horizontal (i.e., the "east-west" direction) is normally a straight line parallel to the bottom edge of the map. However, for a small-scale map – say, one covering a major portion of a continent, it is generally preferred to project the geographic data so that the constant-latitude lines – representing the horizontal – are curved. In that case, area feature names, depending on their length, should be either tangent to the local constant-latitude line or curve with it, as illustrated in Fig. 3.

Various special placement situations need to be considered, such as when an area feature consists of two or more disjoint parts, when an area feature is strongly elongated (e.g., a wide river), and when one area feature contains another. In the last-named case, depending on the nature of the interior area features, the text for the outer feature may or may not be allowed to overlap an interior one. Overlap would be permissible – perhaps even mandatory – for a lake within a state boundary but would not be permissible for a county that is totally enclosed within another county. The two cases are illustrated in Fig. 4.

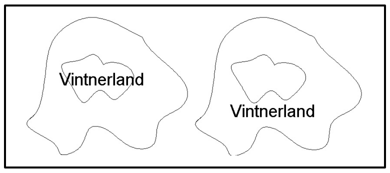

Fig. 4 Two different situations for placing text for an area feature.

For automated text placement software to be effective it must have the sophistication to determine the placement strategy appropriate to a host of situations like this.

Point Features

For point features, the text should be placed close to the feature, with horizontal placement generally preferred. It is convenient to identify 9 primary locations, as illustrated in Fig. 5. The point symbol is identified by the small circle in the center, with locations 0 through 7 surrounding it. Location 8, centered at the point coordinates, would apply if instead of a symbol the text itself is to mark the point location. The user can chose one location or give a number of choices in decreasing order of preference. There is some flexibility about the locations, and the software will make micro-adjustments in a location (as suggested by the dashed lines surrounding location 0) if this would result in successful placement.

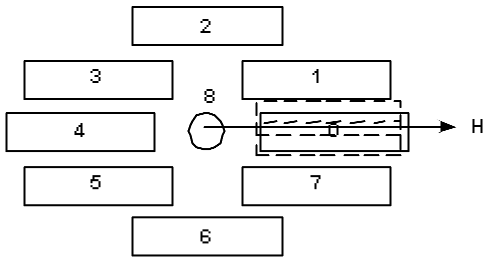

Fig. 5. The primary locations for placing text about a point feature.

Where a point feature is adjacent to a line feature (e.g., a city next to a river or highway), the text should be placed on the same side as the point symbol. Refer to Fig. 2 and note the proper placement for “Queenstown” and “Goodtown” and the improper placement for “Badtown.” Although there is no ambiguity in the placement for “Badtown”, a viewer later recalling what he saw on the map may think of the town as being to the right of the river rather than to the left. An exception to this – another rule! – exists in the case of “Ocean City”. It is a coastal town and by placing it “in the water” a signal is sent to the viewer that it is on the coast. Conversely, for a town near but not on a coast, this should never be done.

Line Features

Text for line features is typically placed centrally along the feature and, if the feature is curved, made to conform to the feature’s curvature, as illustrated in Fig. 6. Text may be placed above the feature or below the feature, and even curved around a 90^o corner. Points of excessive curvature should be avoided if possible. For very long features, the text may need to be placed more than once. Note the peculiar problem that can occur when a line feature curves such that the “above” side becomes the “below” side and the reading direction (left-to right) reverses as the feature is traced out from end to end, as illustrated at the lower panel of Fig. 6.

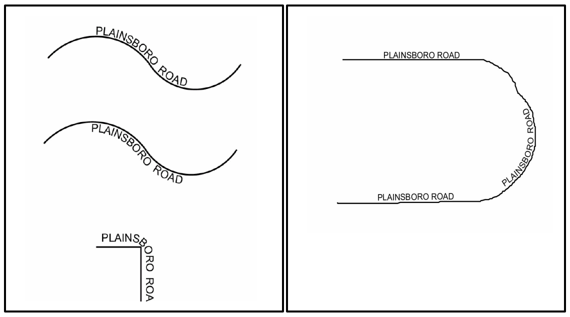

Fig. 6 Line-feature text placed to conform to feature.

For city streets, there are two major styles of labeling – either with the label *alongside* the line feature (above or below it) – the so-called *American system* -- or with the label *within* the street symbol – also referred to as the *European system.* The two styles are illustrated in Fig. 7. *Label-EZ* can be configured to utilize either style.

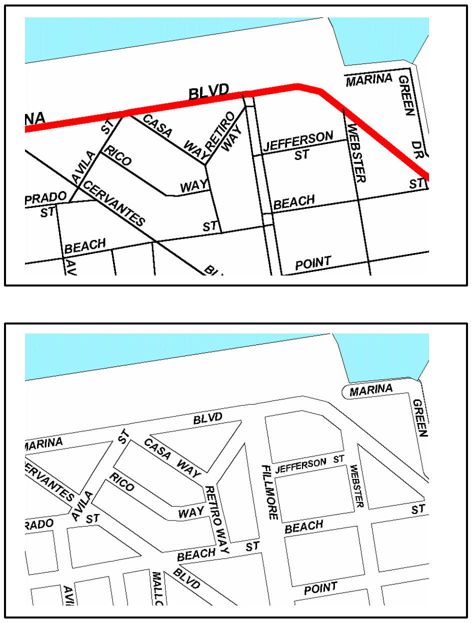

Fig. 7. Two different styles for placing street names. The so-called “American” scheme at the top and the “European” scheme at the bottom.

4 Design of the Placement System

The design for the current automated text placement software evolved over many years. Some of the first implementations, dating back to the early 1980s, were written in FORTRAN and intended to work with geographic data in only one particular format. As a better understanding of the problem was gained, the software was redesigned virtually every year, eventually culminating in a system (now written in C) that was used by the U.S. Census Bureau for labeling the millions of maps required for the 2000 national census. Then starting in 1998, a new version, called *Label-EZ*™, was designed and released for the commercial market. To be commercially viable, it was necessary that the software be compatible with all the major Geographic Information Systems (GIS) – ArcGis®, ArcInfo®, ArcView®, GeoMedia®, MapInfo®, and MicroStation®. This required that the system consist of three major units: an input unit that would accept geographic data from a large number of diverse systems, the text placement engine itself, and an output unit that would return the text placement information to the appropriate GIS system for display, printing, and storage. This is illustrated in Fig. 8.

Label-EZ utilizes two control files to accomplish its task. The *Specification File* is used by the input unit to extract the required geographic data from the GIS system. All features that are to appear on the map must be extracted, whether or not text is to be placed for them since they determine what free space will be available and where and how text can be placed. The file specifies how the features are to be organized into feature classes, what priorities are assigned to the feature classes, where the attributed text for the features can be found in the database, the symbolization, and other parameters affecting the map design. Under control of this file – which is generic in the sense that it can be used over and over again for different maps of the same style, the geographic data, together with its accompanying text, is converted into a standard input format and made available to the Text Placement Engine.

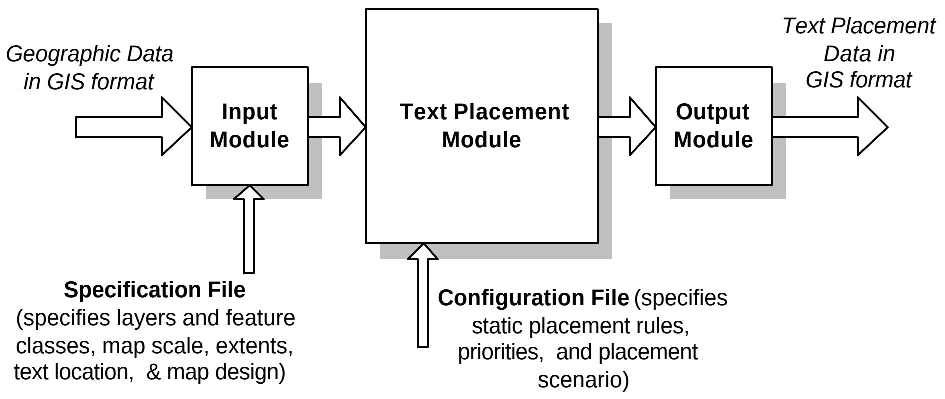
Geographic Data in GIS format
Input Module
Text Placement Module
Output Module
Text Placement Data in GIS format
Specification File (specifies layers and feature classes, map scale, extents, text location, & map design)
Configuration File (specifies static placement rules, priorities, and placement scenario)

Fig. 8. The 3 major units of the automated text placement software system.

The *Text Placement Engine* operates under control of the *Configuration File*, as shown in Fig. 8. It processes features from the feature classes according to the priority specified in this file. The Text Placement Engine consists of a large collection of algorithms, many freshly developed and others drawn from the fields of computer graphics, image processing, pattern recognition, and artificial intelligence. We can regard the Text Placement Engine as a kind of "factory" with a large variety of machines (algorithms) and conveyor belts that allow the output from one machine to be passed to another. The *Configuration file* controls the work flow, determining, for each feature class, which machines (algorithms) are to be used and in what sequence to process the map data. All output is passed to the output unit, where it is again converted to the format required by the particular GIS system for display and any post-editing that may be desired.

For each feature class, the *Configuration File* contains a set of placement specifications. These will be of two kinds, static and dynamic specifications. The static specifications refer to the fixed placement requirements, such as whether the words of a line-feature text are or are not to be spaced apart, the length, which if exceeded, will require that the text of a line feature be placed a multiple number of times, the fontsize to which the text is allowed to be reduced if font reduction is elected as one of the possibilities for achieving placement in a tight space, and similar.

The dynamic specifications or execution commands are used for the placement scenario, the sequence of steps to be followed if the initial ideal placement cannot be realized. A sample scenario is shown in Fig. 9. It is intended for placing the text for a line feature. The initial ideal placement has been specified as in the *center* of the feature and *above* the feature (relative to the bottom edge of the map), as shown at the top left in Fig. 6. If such placement is successful, the scenario terminates, the actual placement data is computed and passed to the output unit. But if this placement cannot be realized, then the placement engine will proceed to execute the next command, namely to place the text *below* the feature. If this is also unsuccessful, the next attempt is to split the text vertically. That is, if the text is "Oxford Street", the word "Oxford" would be placed above the feature and the word "Street" would be placed directly below.

```
CENTER
ABOVE
BELOW
VERTICALSPLIT
FONTLOOP=3
FONTRESET
LEADER
KEYNUMBER
SUPPRESS
```

Fig. 9. Possible execution scenario for line feature text.

If even this last attempt is unsuccessful, the next command (FONTLOOP=3) will cause the font to be reduced by a fixed amount (set by one of the static specifications) and execution to loop back to the beginning of the placement scenario; i.e., it will again try placement above, below, and vertical split but now with the text reduced in size. If this is successful at any step, execution terminates. Else it will loop again, reducing the font by the specified amount up to 3 times. Only if success has still not been achieved, will it move on the next command.

According to Fig. 9, the next command (FONTRESET) will cause the font to be reset to its original size and then an attempt is made to *leader* the text; that is, to place the text some distance from the feature and have it point to the feature with an arrow. If even this proves impossible, then the command KEYNUMBER would attempt to place a small sequential number near the feature and write the text together with this number into a keynumber file. The contents of this file can then be displayed in tabular form elsewhere on the map sheet.

The scenario of Fig. 9 is just one simple example of how the software tries to emulate a human cartographer in attempting to place the text for a particular feature (a line feature in this case.) *Label-EZ* contains some 40 execution commands from which a particular scenario can be constructed. In doing so, the map designer attempts to write out exactly what a cartographer would do under the same circumstances – "if the text will not fit here, shall I reduce its fontsize? abbreviate the text? try stacking the text in two lines? leader the text? or shift some other text to make room for this text?"

In trying to place the text the software must answer two critical questions: (1) will the chosen location satisfy the specified placement rules applicable to this feature class, and (2) is the space free from obstacles, such as other features and other, earlier-placed text? Text-on-text overlap is never allowed. Text-on-feature overlap may be permissible at some stage in the execution scenario for line features and the boundaries of area features, especially if the text and the lines are of different color. It is in responding to these two questions that the Text Placement Engine expends most of its effort.

The ultimate test for an automatic map labeling system is how well its results compare with those produced by an expert cartographer. Map labeling is considered an art form, and cartographers pride themselves on their artistic skill – which any automatic system must also somehow embody. In developing the automatic text placement described here, tools were drawn from pattern recognition, computer graphics, and image processing. They were carefully assembled, over many years and with much trial and error, until a system evolved that would satisfy cartographic conventions and yield a map that would be free of label-to-label conflict, would avoid ambiguities, and would facilitate easy visual comprehension.

5 Some Examples

Small sections of a diverse set of maps are shown here to illustrate the capability of the automated text placement software. All the maps were labeled *totally automatically*. There was no editing or human intervention of any kind. Typically about 95% to 98% of the labels were placed successfully; those that could not be placed were suppressed. (When a label is suppressed, the label information is written to a log file from where it can later be retrieved for manual editing.) In virtually every case the reason is inability to find satisfactory space under the rules specified. What to do in such a case must be left to a cartographer – possibly another name can be deleted to make space free for the suppressed label, the map scale can be increased, or one or more of the placement rules modified and the data then re-run through the software.

Fig. 10 shows a city-street map. Note how the names for small streets were placed by using curved leaders.

In Fig. 11 we show a soil map produced for the Natural Resource Conservation Agency, U.S.D.A. One of the challenging aspects of this kind of map is that the polygons can be extremely convoluted and that the polygon label may need to be placed a multiple number of times if the polygon is large.

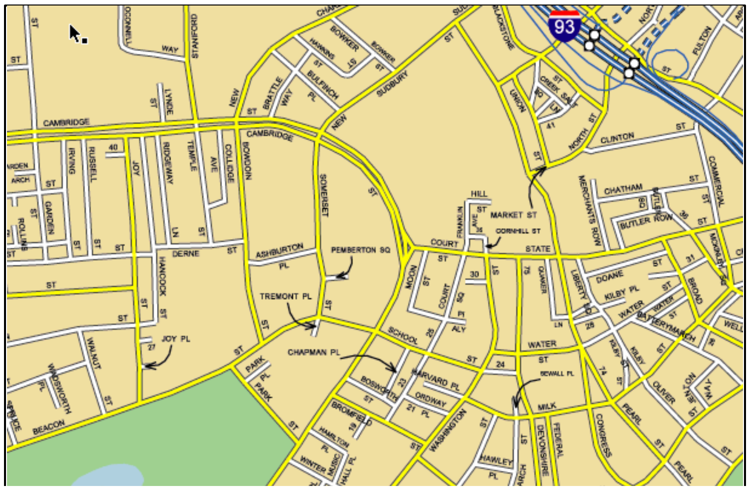
Fig. 10. Section of a city street map. Note curved leaders used to place text for small streets.

Fig. 12 shows a section of a nautical chart. Note the depth soundings, the labeled depth contours and the buoys and beacons. Nautical charts (as well as aeronautical charts) present special challenges in that for some of the text there is no flexibility as to its location, e.g., a depth sounding must be in pre-cisely the designated location; it cannot be moved. Also these kinds of charts contain features that require multiple text placements, e.g., for a buoy the buoy color, buoy name, and buoy attribute must be placed according to separate specifications, with minimum latitude as to where the text may appear.

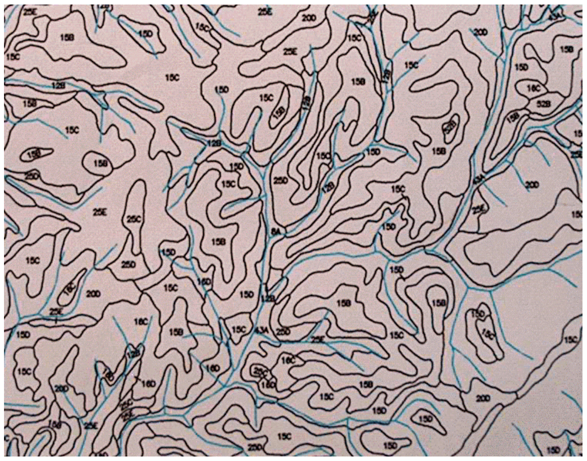
Fig. 11. Section of an automatically labeled soil map. Note the highly convoluted polygons

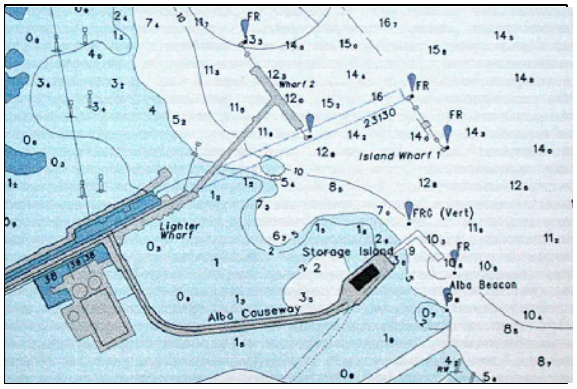
Fig. 12. Section of a nautical chart.

In Fig. 13 we show a terrain contour map. The contour lines represent elevation values. The automated labeling of contour maps is particularly challenging because labeling must take account of the complex relationships existing among the contours and the rigid cartographic conventions that need be followed. The contour labels must be placed in a well-coordinated manner and such that the upper edge of each label faces the next higher-value contour; i.e., such that as one reads up in the labels one also moves upward in elevation, even though this may on occasion result in labels being placed upside-down on the map.Complex pattern recognition algorithms, many freshly developed, were required to master this particular label placement problem.

6 Conclusion

This paper has described the design of software for solving the automatic cartographic text placement task. The task, done for centuries by artistically skilled cartographers had long defied all attempts at successful automation. This can be attributed to the infinite diversity of maps, which must depict geographic reality and serve a multitude of purposes, and the complexity of communicating spatial relationships via the human visual system. The problem has been exacerbated by the mapmaker's desire to cram as much information as possible into the smallest possible space. Now finally, maps can be labeled in seconds or a few minutes as against the weeks the task required when done manually.

The challenge for automated map labeling in the future lies in fast, dynamic labeling of electronically displayed maps and charts. With the increasing demand for viewing maps on the web and having the capability to pan and zoom over the display, text placement must be fast – requiring no more than one or two seconds for an update. Fortunately maps to be displayed electronically tend to have far fewer features than typical paper maps and the quality requirement is not as high as for a paper map. Much progress in this area is being made and the day of 1-second quality labeling of an electronically displayed city map or aeronautical chart is not far off.

Acknowledgment

The design of the text placement software and the successful solution of this long-vexing problem is the result of the dedicated efforts of a large number of people, too many for all to be mentioned here. The following individuals, however, need to be singled out for special recognition: John Ahn, Shailesh Chanekar, Jeff Doerschler, Sean Marrinan, Rishi Mehra, Mark Tyberg, Martin Tyberg, Aruna Vedula, and Weiping Yang.

Bibliography

[1] Imhof, E., "Die Anordnung der Namen in der Karte," *Annuaire International de Cartographie II,* Orell-Füssli Verlag, Zürich, 93-129, 1962.

[2] Yoeli, P., "The logic of automated map lettering," *The Cartographic Journal, 9* (2), 1972, 99-108.

[3] Freeman, H.,, Map data processing and the annotation problem, *Proc. 3rd Scandinavian Conf. on Image Analysis,* Chartwell-Bratt Ltd. Copenhagen, 1983.

[4] Ahn, J. and Freeman, H., "A program for automatic name placement," *Proc. AUTO-CARTO 6,* Ottawa, 1983. 444-455.

[5] Freeman, H., "Computer Name Placement," ch. 29, in *Geographical Information Systems, 1,* D.J. Maguire, M.F. Goodchild, and D.W. Rhind, John Wiley, New York, 1991, 449-460.

[6] Doerschler, J. and Freeman, H., "A rule-based system for automated name placement," *Commun. ACM, 35* (1), 1992, 68-79.

[7] Colquist, J., Mehra, R., Tyberg, M., and Freeman, H., "Automated name placement for digital nautical charts," *Proc. GIS/LIS,* ACSM, Cincinnati, OH, Oct. 1997.