International Journal of Image and Graphics, Vol. 2, No. 3 (2002) 453–479

A COMPLETE SYSTEM FOR INTERPRETATION OF COLOR MAPS

S. ABLAMEYKO*, V. BEREISHIK, M. HOMENKO, D. LAGUNOVSKY, N. PARAMONOVA and O. PATSKO

Institute of Engineering Cybernetics, Belarussian Academy of Sciences, Surganov str., 6, 220012 Minsk, Belarus
**abl@newman.bas-net.by*

Received 28 February 2001
Revised 17 January 2002
Accepted 14 March 2002

The paper describes a system and technology for the automatic/interactive interpretation of color maps. Due to the fact that a completely automatic solution is impossible, a combination of automatic and interactive techniques is used. Firstly, color recognition and separation is performed. For automatic map vectorization, a special fast scheme is used. To make digitization more automatic and user-friendly, an interactive digitizing mode has been developed to extract line-based cartographic objects from vectorised data. The system has a powerful and friendly user interface that is described in the paper. The system is used successfully in the digitization of topographic and other types of map.

Keywords: Map Interpretation; Image Vectorization; Cartographic Object Recognition; Interactive Map Digitizing; Graphics Recognition.

1. Introduction

At present, Geographic Information Systems (GISs) are being applied in all areas of map generation, processing and usage. However most maps are still in paper form, and thus unusable by GISs, which provide easy to use, powerful and flexible interfaces through which their users can manipulate various electronic representations of the relevant information. A GIS user can perform many map-related tasks much more quickly and efficiently than would be possible using only paper-based representations. If the benefits of this technology are to be maintained, many important paper-based maps must be input into a GIS and represented in a suitable format.

This is a potentially huge task. It is also complicated by the number of different map types involved, and their differing requirements (application areas include such diverse fields as agriculture, ecology, geology, forestry, and the military).

*Phone: +375 (17) 284-21-44; Fax: +375 (17) 231-84-03

Moreover, new maps have to be produced, and old maps modified, and many tasks have to be performed which rely on, or otherwise involve, the use of some kind of map. Manual map production is a very slow and expensive process, which should be as automated as possible. The obvious benefits to be accrued from automatic map interpretation have attracted a significant amount of industrial attention and academic research effort.

Map digitizing systems use two main types of input devices: (1) digitizers that still are very much in use, and (2) scanners that started to be used during the last ten years. The advantages and drawbacks of these two types of devices and data representations for inputting cartographic information are given in Ref. 1. In this paper, we consider map input that is done only by scanners.

The scanned data are vectorized and represented in a GIS format. Generally, the process of scanned map vectorization can be divided into three main stages: color separation, raster-to-vector transformation with the aim of obtaining a structural representation, and automatic/interactive recognition of cartographic objects to obtain the required final map representation.

Color separation is an important task because several colors and objects can be overlapped leading to many errors in the output representation. There are several papers and systems that propose effective solutions for this task[2–4] but these depend on region-specific cartographic conventions. In some countries, only a few of the more common colors are used in cartography, however other countries use many colors, often leading to maps whose quality leaves much to be desired.

There are many systems for performing raster-to-vector (r-t-v) transformations. In Sec. 2, we analyse commercial image vectorization systems as well as research approaches. Automatic recognition of cartographic objects still remains within the realms of research, and is not widely adopted.[5–8] Experience has shown that it is impossible to recognize automatically a whole map because it is produced for human interpretation, and includes objects with different fonts, orientations, sizes, etc. Moreover, automatic object recognition is usually followed by a significant amount of interactive post-processing to correct recognition errors. This is why existing systems in current usage are intended to vectorize maps, and represent them in terms of simple primitives.

We have worked in the area of map interpretation for about last 20 years and created several generations of map interpretation systems. This has allowed us to accumulate considerable experience in this area. The current version of the map interpretation system was developed during last 3–4 years. In this system, we developed a combined technology of color map digitization that includes the following stages:

- automatic or interactive map color separation;
- automatic binary image vectorization;
- semi-automated object interpretation controlled by a human operator.

Such a combination minimizes digitization errors, decreases the timescale of digitization, and the complexity of the operator's task. It permits an acceptable ratio of quality, time and automation in the digitization process, and avoids the need for the correction of errors. This technology and the resulting system are presented in this paper.

We start by analyzing map digitizing systems and approaches (Sec. 2). Main principles for color map interpretation are presented in Sec. 3. Then, we consider map digitizing technology and system structure (Sec. 4). The binarization module and algorithms are considered in Sec. 5. Vectorisation process is described in Sec. 6. Interactive digitizing techniques as well as automatic isoline recognition techniques are described in Sec. 7. System interfaces are considered in Sec. 8. In Sec. 9, we discuss some results of our work, with a critical analysis of the results obtained. Section 10 contains conclusions and main distinctions.

2. Map Interpretation Approaches and Systems

2.1. *Map digitizing approaches*

Existing commercial line drawing digitization systems typically involve an operator and incorporate various types of interactive procedures. One can identify the following main types of the technologies (moving from manual to automatic ones)[1,9]:

- "Blind" digitization occurs when the drawing to be vectorized is pasted onto a digitizing table and the operator indicates the start and end points of each line by clicking on them with a mouse. She/he concentrates on the drawing and has no direct view of the result of the process. This is a fully manual drawing digitization technology which is used for already 30 years to input maps to computers. A major drawback of this approach is the difficulty of providing visual control to the digitization process. The user typically finds it very hard to tell which objects have been digitized and which are awaiting input. The operator is also unaware of any errors made until the whole drawing has been input and a hard copy of the result produced on a suitable plotter.
- In "interactive" vectorization a similar procedure is used, the difference here is that the operator receives immediate feedback as the points and lines entered are displayed on a high resolution computer screen. She/he can see and correct any errors during the process. The disadvantage of this approach is that the operator must constantly move his/her head between the screen and the digitizing table. This may not be desirable from a human engineering point of view.
- Scanning and "heads-up" vectorization involves the drawing being scanned and displayed in raster format on a high resolution screen. Vectorization is achieved manually using a mouse as above. The operator uses a mouse to traverse the screen, marking key points with clicks. The resulting vector data is displayed as an overlay on the original image, providing a better opportunity for error detection than interactive vectorization without the need for the operator to constantly change viewpoints. Raster unprocessed data are displayed on the screen in one (dim) color and an other (bright) color is used to denote entities that have already

been digitized. The user can be provided with a range of input tools (raster background, "microscope," moving "glass," changing pick square, etc.) and can select those that make his/her work more comfortable.

- Scanning, automatic vectorization and interactive object digitization by using vectorized data. The vectorized data must be processed to extract cartographic objects. It can be done by the system under operator control. When the system is not able to deal with the ambiguity present in the image to decide where a particular vector should be placed, the operator is prompted to provide his/her judgement. The software then continues to process the image. This combination of interactive and automatic processing allows minimization of automatic digitization errors, decrease digitization time and make the operator's work at least a little more interesting and less prone to error.

This is especially important for digitization of elongated, twisting line objects. The vectorized data contain the coordinates of points lying on these objects: to identify elongated objects in a vector database the user need only indicate component segments in some appropriate order, perhaps by selecting them with a mouse. This simplification of the digitization task makes map input both faster and less wearisome. Moreover, some line objects can be identified automatically. For example, there are quite stable dashed line recognition algorithms.[10,11] Automatic line recognition techniques can be combined with interactive tools.

The last possible variant of map digitizing is fully automatic vectorization and object recognition. However, it can be done for only very simple maps (some special types of cadastral maps for example). In any case, the operator should be attracted to edit errors appearing during automatic recognition.

2.2. *Commercial map vectorization systems*

Document processing systems capable of storing forms and other structured documents, performing OCR on typewritten text and compressing drawings are now commercially available. There are many systems on the market which perform or support vectorization of line drawings. It is not possible to describe all these products, but we can acknowledge systems developed by the following companies: Able Software Company, AccuSoft Corp., Alpharel, Arbor Image Corp., Audre, ColorSoft Inc., CPI, Digist Software, ERDAS Inc., Grumman InfoConversion, GTX Corporation, Horizons Technology, InfoGraphix Technologies, Intergraph Corp., MicroImages Inc., MST, Pacific Gold Coast, Rorke Data Inc., Softdesk Imaging Group, Sovereign C. S. Ltd., Vidar Systems Corp.[12]

Many vectorization systems are currently available in the Russian and Eastern European markets. In the USA and Western Europe the primary development of GIS technologies took place at a time when scanners were very expensive and hence rarely used. As a result, most cartographic data was input to the new systems via manual digitizers. In Russia, interest in GISs has developed only recently, after scanners and scanning technology had become inexpensive and readily available.

Document image processing and vectorization systems were therefore developed early and transferred to personal computers very quickly. Moreover, the rapid growth of GIS systems in the East has made scanning technologies very popular there, with scanners being used in practically all map digitization.

2.3. *Laboratory map interpretation systems*

Current research approaches to map drawing interpretation can generally be classified as either bottom-up or top-down.[13] The bottom-up approach that could be qualified as "traditional" is characterized by an emphasis on the analysis of small groups of connected or otherwise physically closely related pixels and relies upon data-driven, local processing. Bottom-up interpretation systems tend to start with the image and move towards abstract, entity-level descriptions. Papers[14,15] describe interactive systems for interpretation of graphic images, based on vectorization and extraction of primitives. An interesting approach was proposed in paper[16] which suggested a map analysis system endowed with learning capacities. This system uses a model-based approach for the extraction of primitives. During the processing, it is proposed for the human operator to add new learning rules and the domain knowledge representation is separated from the extraction process, as far as it is possible.

Another example of bottom-up strategy is the Japanese MARIS system that has been used to digitize large, reduced scale maps of Japan.[17] A vector database is created and forms the basis of object recognition. Interactive editing and correction of mis-recognised and unrecognised objects refines the automatically produced results. MARIS supports the digitization of three map layers: building lines, contour lines and lines for railways, roads and water areas.

In contrast, top-down approaches concentrate on the relationships among graphical primitives, objects, and scenes. Systems built around this type of architecture typically begin with some description of the entities they expect to find and proceed by seeking evidence for the presence of those entities in the input drawing. Such systems therefore rely more on model-driven, global processing.

Top-down strategy was used in German PROMAP system that was developed to digitize German topographic color maps with a scale of 1:25000.[2] The maps are scanned with a 24-bit red, green, blue scanner and separated into four layers. Symbols and objects are identified at the raster level and further vectorization is performed. A particular strength of PROMAP is its use of large amounts of *a priori* knowledge.

A system that automatically extracts information from paper-based maps and answers queries related to spatial features and the structure of the geographic data they contain is described in Ref. 18 Algorithms to detect symbols, identify various types of line and closed contour, compute distances and find shortest paths have been realized within this system. Its query processor analyzes queries presented by the user in a predefined syntax, controls the operation of the image processing algorithms, and interacts with the user.

Document image processing and vectorization systems were therefore developed early and transferred to personal computers very quickly. Moreover, the rapid growth of GIS systems in the East has made scanning technologies very popular there, with scanners being used in practically all map digitization.

2.3. *Laboratory map interpretation systems*

Current research approaches to map drawing interpretation can generally be classified as either bottom-up or top-down.[13] The bottom-up approach that could be qualified as "traditional" is characterized by an emphasis on the analysis of small groups of connected or otherwise physically closely related pixels and relies upon data-driven, local processing. Bottom-up interpretation systems tend to start with the image and move towards abstract, entity-level descriptions. Papers[14,15] describe interactive systems for interpretation of graphic images, based on vectorization and extraction of primitives. An interesting approach was proposed in paper[16] which suggested a map analysis system endowed with learning capacities. This system uses a model-based approach for the extraction of primitives. During the processing, it is proposed for the human operator to add new learning rules and the domain knowledge representation is separated from the extraction process, as far as it is possible.

Another example of bottom-up strategy is the Japanese MARIS system that has been used to digitize large, reduced scale maps of Japan.[17] A vector database is created and forms the basis of object recognition. Interactive editing and correction of mis-recognised and unrecognised objects refines the automatically produced results. MARIS supports the digitization of three map layers: building lines, contour lines and lines for railways, roads and water areas.

In contrast, top-down approaches concentrate on the relationships among graphical primitives, objects, and scenes. Systems built around this type of architecture typically begin with some description of the entities they expect to find and proceed by seeking evidence for the presence of those entities in the input drawing. Such systems therefore rely more on model-driven, global processing.

Top-down strategy was used in German PROMAP system that was developed to digitize German topographic color maps with a scale of 1:25000.[2] The maps are scanned with a 24-bit red, green, blue scanner and separated into four layers. Symbols and objects are identified at the raster level and further vectorization is performed. A particular strength of PROMAP is its use of large amounts of *a priori* knowledge.

A system that automatically extracts information from paper-based maps and answers queries related to spatial features and the structure of the geographic data they contain is described in Ref. 18 Algorithms to detect symbols, identify various types of line and closed contour, compute distances and find shortest paths have been realized within this system. Its query processor analyzes queries presented by the user in a predefined syntax, controls the operation of the image processing algorithms, and interacts with the user.

A system capable of the robust identification of drawings superimposed on maps is described in Ref. 19 First, graphic and character regions are decomposed into primitive lines, the definition of which includes their orientation and connections to other lines. Objects are extracted from these primitive lines by recognition techniques that use shape and topological information to group them into meaningful sets such as character strings, symbols, and figure lines. The system has been used to interpret equipment diagrams superimposed on maps.

At present, bottom-up approaches remain the most commonly used, particularly for lower level interpretation tasks such as map vectorization. Top-down approaches are usually referred to as knowledge-based because they use a priori knowledge to guide object recognition.[8]

3. Map Interpretation Principles

Our experience has shown that in developing map interpretation technology, adhesion to the following principles is profitable[1]:

(i) *Combined automatic/interactive interpretation.* As one can expect, only the simplest maps can be successfully digitized by a fully automatic system. That is why a combination of automatic and interactive techniques should be used to digitize other maps. Maps can be scanned and vectorized automatically, but object recognition, and map representation should be performed using a combination of automatic and interactive techniques.

(ii) *Color separation and layer processing.* In the first stage, colors should be extracted and separated. This extracts several map layers, each containing a separate color. The order in which the various layers are to be processed should be clearly defined. Imposing a fixed order on the interpretation of these strata allows information obtained from one layer, to be exploited during the interpretation of the next. Experience suggests the most effective interpretation sequence is that layers containing area objects should be processed first, followed by isoline and hydrography layers. The black layer should be processed last.

(iii) *Multi-level recognition.* The recognition system should be multi-level. At the first level, object components are extracted, recognized and relations between them established. At the second level, simple cartographic objects are recognized, and at the third level complex objects and scenes are identified.

(iv) *Use auxiliary information.* Auxiliary information from other map layers or sources (neighboring map sheets, neighboring or otherwise related objects) should be exploited wherever possible. In particular, interpretation should start with objects that have associated auxiliary information.

(v) *Proceed from the simple to the complex.* In the absence of auxiliary information, interpretation should begin with objects having clear, simple structures and only later move toward more complex entities. In practice this usually means that interpretation should begin with line objects, which are generally simpler than symbols and textured areas to allow the collection of valuable auxiliary information.

(vi) *Exploit all available knowledge.* Knowledge of map conventions, cartographic objects, scenes etc. should be used wherever possible. During map production, domain-specific rules and knowledge are used to define the object, as well as scene representations upon which the map is based. During map interpretation this same knowledge can support object recognition. Use of knowledge in the analysis of complex situations can significantly increase the proportion of objects that can be recognised automatically.

(vii) *Separate knowledge and algorithms.* The interpretation system should be organized so that information and knowledge about objects and map properties is clearly separated from the recognition algorithms employed. Hence if new map types must be interpreted, the knowledge base may have to be updated but, ideally, the recognition algorithms will not be affected. Perfect separation is hard to achieve but could allow significant changes to be made to the scope of the system without large modifications to the system software.

(viii) *Take a structural approach.* As there are very many different types of maps in common use it is not surprising to find that there is as yet no unique recognition method that can be applied to all map images. Experience suggests, however, that of the various forms of pattern recognition currently available (statistical, structural, hybrid etc.) structural pattern recognition is best suited to recognition of line objects and symbols from vector data.

4. Map Interpretation Technology

The developed map interpretation technology includes three main modules that are three consecutive stages:

- *Binarization module* — a raster-oriented module for the separation of the color image into several binary images corresponding to different color layers;
- *Vectorization module* — a binary image processing oriented module for raster-to-vector transformations of one or more binary images;
- *Digitization module* — a vector-oriented module to digitize map objects on the basis of vectorized data with/without using raster data.

The common technological basis for these modules is that they use coordinated data for information exchange. These are files that we call raster-to-vector projects — *r2v–projects. r2v–projects* contain raster and vector data represented in a format internal to the system. Such projects are used as input/output for all system modules. They can also be used as an intermediate archive for partially binarized, vectorized, or digitized data. In accordance with the state of processing there are partially, or entirely binarized, vectorized and digitized r2v–projects. The brief scheme of the system and data exchange is shown in Fig. 1.

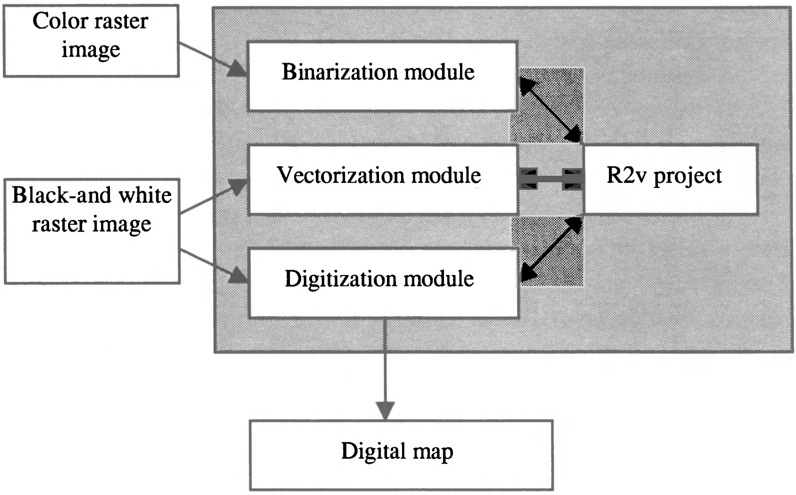


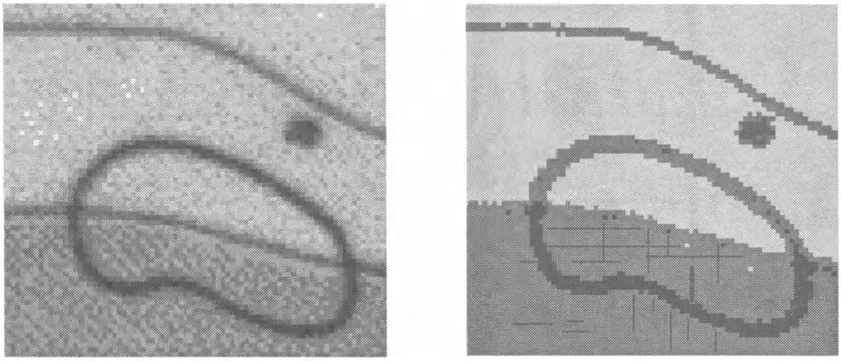
Fig. 1. Map interpretation technology and the scheme of data exchange between its modules.

5. Color Image Binarization

The *Binarization module* takes a scanned color raster image as input in BMP, PCX, TIFF, or whatever format. The input image is then represented with 8, or 24 bits per pixel. This module separates color image into a set of different color logical layers (CLL), each of which is a binary image corresponding to one of color layers on the map (forests, roads, rivers etc.). Each CLL is given a level of priority, in order to reflect the contents more correctly. In addition to a set of CLLs the *Binarization module* produces prepared color image (PCI), which is an original color image with enhanced quality. The PCI, and the set of CLLs together form a binarized r2v–project that is the output of the *Binarization module.* Map color layers are extracted automatically. At the same time the user can manage the quantity of automatically extracted colors. The color separation algorithm is based on color clustering, using statistical processing of color space data, and color neighborhood analysis. The statistical data includes such parameters as color histograms, the quantity and distance between colors in the image.

Every color layer is represented as a group of closed colors in color space. At the same time this "color group" forms some "area object" in the image that can be called a "color object." The layer extraction algorithm looks for these "color objects" and tries optimally to recolor the image plane using the parameters of these objects.

An interactive correction of color layers is provided by adding/deleting colors to/from the selected layer on the basis of clustering analysis. The maximal quantity of color layers that can be extracted (using automatic or interactive tools) does not exceed 15. An example of automatic color image binarization is shown in Fig. 2(a) (initial image) and Fig. 2(b) (processed image).



(a) (b)

Fig. 2. (a) Initial image and (b) automatic color extraction result.

(a) (b)

Fig. 3. (a) Initial image and (b) interactively extracted color layers.

In order to process lower quality color images interactive tools can be used for color layer extraction. An example of interactive color image binarization is shown in Fig. 3.

6. Image Vectorization

The *Binarized r2v–project* obtained from *Binarization module* is the main input for the *Vectorization module*, although single binary (black and white) maps can also be input into this module. In the first case binary images corresponding to different CLLs are loaded into separate MDI-windows, and the user works with them, independently of each other; in the second case, the user works only with one binary image.

Map-drawings are usually in a large format (up to A0). To process such large-size map images on a personal computer, a scan-line scheme was developed, that allows the main vectorization operations to be performed during a single image scan.[14,20]

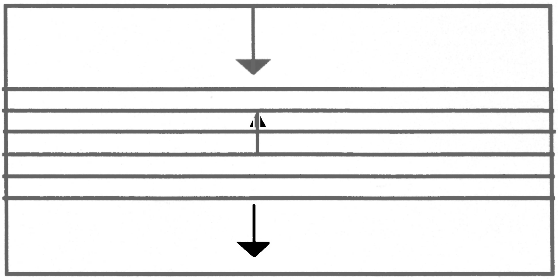


Fig. 4. Line stripe stored in the buffer and its movement.

Since all vectorization tasks are usually solved using small (3×3 pixel) local masks or windows, each operation on a line of data is fully determined only by two adjacent lines (upper and lower). These lines could be stored in an especially reserved memory buffer (Fig. 4). Each program has access to all of these lines, and can check and change line pixels. The lines in the buffer are processed from bottom to top. Consequently the buffer moves over the whole image from top to bottom, and all operations are performed during this move. Therefore after every inspection of the buffer, the corresponding operation is fully executed for the top line buffer. This line is removed from the buffer and a new bottom line is introduced. Use of this principle to each operation of r-t-v transform allows a significantly increase in the speed of the r-t-v.

The main operations at this stage are: image filtering, thinning, or contouring depending on the required output object representation; transformation to a vector form with segments and their characteristics; extraction and forming an intermediate vector database for further object recognition.

To process different map-drawing image types, the following variants of raster-to-vector transforms have been developed:

(i) The region-wise image vectorization variant — which produces a contour description of objects.
(ii) The line-wise image vectorization variant — which produces a skeleton description of line objects and symbols.

Let us consider these variants and the main operations (contouring and thinning) in some more detail. The contouring algorithm stores two lines in the stripe, and extracts the situation contained therein and its resolution. The possible image situations are the beginning of an object, its continuation, splitting, merging, and its end.[14] With contour extraction, special buffers are reserved. They are intended for the assembly and storage of information about every contour. These buffers can be merged or split depending on the processing situation. Simultaneously with the object contour extraction, the geometrical parameters of objects are calculated. The relations between processed objects (e.g. if an object is located inside another) are computed.

To thin image objects, $w + 3$ lines are stored in the line stripe, where w is a maximal thickness of objects.[14] Thinning is performed from the first line (bottom) to the last (top) line. Every line in the stripe is thinned a variable number of times. The lowest line is completely unthinned and the highest is fully thinned, all middle lines are only partially thinned. For the thinning of one line, two adjacent lines (lower and upper) are used. For every processed pixel, its 8-neighborhood is analysed. Using 8-neighborhood code, an input in a look-up-table is performed where a new value of the central pixel is defined.

Thinned image vectorization is performed in a single scan storing three image lines in the stripe. In the stripe medial line, black pixels are analyzed using a Crossing number.[1,14] Depending on the Crossing number value, the following situations are extracted: object beginning, end of object, continuation, merging, splitting, node, and an isolated point. In the vectorization output, the segments of objects bounded by end points and nodes are extracted. The segment approximation is performed simultaneously.

As a result, the vector representation of the image consists of a set of vector segments. Each such segment is described in vector form by contours and/or parts of the skeletons of image objects bounded by feature (end and node) points. It contains the coordinates of an approximating polyline, and some parameters (coordinates of bounding rectangle, types of feature points, length, middle width etc.), which are useful for further recognition and automated tracing of map objects.

All the extracted data are recorded in the *Vectorized r2v–project* which forms the output of the *Vectorization module*, and is the main input of the *Digitization module*. This project contains the PCI, a set of CLLs and united vector data including sets of vector segments belonging to all CLLs, and obtained as a result of raster-to-vector conversion. Every segment of vector data contains a label recording the CLL to which it belongs. It allows both the separation of segments from different CLLs during recognition, and the assembly of objects containing segments from different CLLs in accordance with output format requirements.

7. Interactive Interpretation of Vectorized Maps

The *Digitization module* has two different types of data input. The first one is a *vectorized r2v–project* obtained from the *Vectorization module* where both raster and vector data can be used. In this case both manual and automated digitization modes are possible. The second one is a *binarized r2v–project* containing only raster data. It can contain color or binary raster images, in which case, only manual digitization is possible.

The main task of the *Digitization module* is to digitize objects in order to obtain a digital map in the required format. There are three main types of map objects: line objects, area objects, and symbols. Each map object is defined by three main parameters: object code, metrics (geometrical coordinates), and semantics (object characteristics). The object code is determined by its graphical representation on the image, and implies a concrete object or object class on the map drawing. Metrics are determined from objects' dispositions on the image, and vary with different object types.

Automatic object recognition techniques are not always suitable for map object extraction. The reason being that, in many cases, the number of errors after automatic recognition is large, leading to considerable error correction effort, possibly compromising the benefits of automation. That is why it is better to use a combination of automatic and interactive techniques. We first describe interactive modes used in our system and then show how we automatically extract line objects.

7.1. *Main digitizing modes*

A drawback of the traditional manual digitization process is the difficulty in digitizing line objects, usually realized using a manual tracing technique. In a system based on automatically vectorized data it is possible to eliminate this drawback. As a result of raster-to-vector conversion the coordinates of object components (segments bounded by special points) are formed automatically. To digitize line objects consisting of a set of segments, a user can identify segments in the appropriate order using a simple pointer device (e.g. a mouse). Such work simplification makes a map digitization faster and less difficult. Furthermore, some automatic recognition techniques (seeking the best chain continuation described above) were incorporated into this technology. For example, if the operator is confident of the correct continuation of an emerging chain line, he/she needs only confirm the decision of the system.

Several forms of object digitization have been realized within our system.[21,22] For simplicity they are referred to as the Draw, Pick, Go, Run and Jump modes.

- Draw mode allows the user to input object co-ordinates with a mouse, and is mainly used to digitize text, symbols, and poorly scanned line and area objects. The user can work over a raster image background, which provides contextual information in order to aid the digitization process.
- Pick mode allows the operator to input line and area objects by choosing vector segments to be connected, again using a mouse. The system automatically selects the manner of connection between the developing chain and each newly picked segment and proposes this form of continuation to the user. The nature of the proposed connection is determined after consideration of the relationships between segments already comprising the chain, and the new segment (see next sub-section).
- In Go mode, the system automatically extracts possible continuations, ranks them using the criteria employed in Pick mode and proposes the best one to the user. Some five or so ranked options are usually provided. The operator may select one of these or, if his/her preferred solution is not among the list, view other variants, or switch to Pick mode to indicate the required segment.
- In Run mode, the system tries automatically to join segments to form a chain of some predetermined length (around 15 segments are typically combined at any

one time). The connectivity criteria used in Pick mode are again employed. The difference between the connectivity score for the best and next best solutions provides a measure of confidence in each continuation. If the confidence of a given continuation is high the system will proceed to attempt the next extension; otherwise it switches into Go mode and asks the operator for confirmation.

- Jump mode is a combination of Pick and Run: the user specifies a start segment and an end segment which the system then tries to connect with a high confidence chain line. The solution is presented to the user for confirmation. To this end a tree of possible continuations is formed. Every possible continuation has an associated weight and maintains a reference to its parent, the previous segment of the proposed chain. Initially, the tree comprises only one segment having zero weight. To grow the tree, a segment with no children and minimal weight is selected. Other vectors, which might form acceptable continuations to the selected line, are then identified and added to the tree as children of the current segment. The weight of each new child is calculated as a linear function of the weights of the segments in the chain, the length of the new segment and its distance from both its parent and the end segment. The procedure terminates when the user-defined end segment is added to the tree. Backtracking through the tree to the start node then produces a chain line. The method has proven effective for objects that have many connected segments and not many gaps. An example of tracing in Jump mode is given in Fig. 5.

7.2. *Automatic extraction of line objects*

Line objects in maps may be considered as curve lines having a repeated structure with different length and thickness of segments, and different length of gaps. The simple case is isolines when they are not crossed by other objects; they can be considered as dashed lines in this case. However, intersections of line objects with other objects often exists on maps.

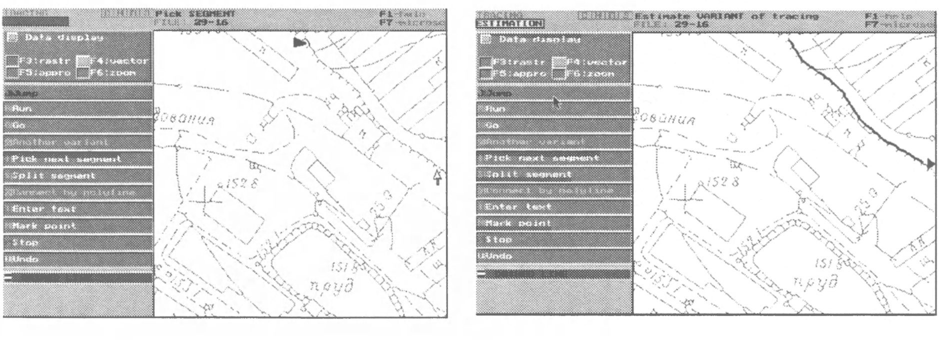


Fig. 5. Example of "Jump" operation implementation (current chain tail is shown by triangle, picked segment shown by "mouse" in (a)) and automatically extracted line object is shown by thick line in (b).

The most common approach to recognition of line objects is to thin a binary image and vectorize the resulting skeleton.[17,23] The approach produces representations that are both closely related to the required output format and well suited to approximation and/or structural/syntactical line recognition. Among its drawbacks are that it is very sensitive to small irregularities in the line's contour; small branches are produced which must be pruned away afterwards, junctions also may not be properly represented without some noise reduction.

A number of techniques for dashed line recognition are available. The algorithm proposed in Ref. 24, for example, allows dashed lines to comprise either single length segments or alternating long and short lines. A two-stage algorithm to detect dashed lines is described in Ref. 11. Line segments are first grouped together using local constraints. Context-dependent, global syntax rules are then applied to resolve any conflicts in interpretation. An approach based on a similarly global view of the drawing is described in Ref. 25. Here the input image is divided into rectangular regions and intersections sought between drawing lines and region borders. An effective sparse-pixel algorithm for bar recognition is proposed in Ref. 26. This also performs a careful sampling of the image before focusing attention on areas identified as key. The Hough transform provides a useful general method of grouping together broken line segments. Any construct hypothesised by a Hough technique must, however, be checked for consistency in both segment length(s) and intersegment gaps.

In order to compare and promote development of better techniques, a dashed-line detection contest was held at International Workshops on Graphics Recognition.[27] Performance evaluation of dashed-line detection algorithms, as well as description of the test images, is given in Paper.[28]

The algorithm described here differs from previous approaches by using a slightly deeper analysis of the situations likely to arise in a map to guide the combination of line segments; the result being a marked increase in reliability. The technique was specially developed to deal with the complex situations that arise within map images when contour lines interact with other entities and individual segments are situated very close to each other.

To recognize such object types, we use a structural approach. This approach is based on two main stages: forming a description of an object and analysis of this description to classify the object. To describe the line and other complicated objects, we use an approach based on formal grammars. The formal grammar approach requires a definition of the following:

- terminal elements,
- nonterminal elements,
- starting nonterminal elements,
- grammatical rules.

Consider how these concepts could be determined to extract line objects.

As terminal elements, the primitive line segment (s), gap (g), point (p), and closed contour (c) are used. All the terminal elements are divided into two groups:

constructive and connective. The constructive elements are used to obtain geometric coordinates of the line objects, and the connective elements are used to connect the constructive elements in a chain. Each terminal element has some attributes (features) like length, thickness, and type. The description of a line is then obtained by using different combinations of the elements, For example, the description of an isoline formed by the repetition of

- open line segments and gaps is given by the rule:

$$\mathrm{P} : \mathrm{G} \rightarrow \mathrm{sgG}$$

- open line segments, points, and gaps is given by the rule:

$$\mathrm{P} : \mathrm{G} \rightarrow \mathrm{sgpgG}\,,$$

where G is a nonterminal element and starting symbol.

The terminal elements have the following attributes:

$$\begin{aligned} \mathrm{s} &=:: \langle\text{thickness}\rangle\langle\text{type}\rangle\langle\text{length}\rangle\,, \\ \mathrm{g} &=:: \langle\text{length}\rangle\,, \\ \mathrm{p} &=:: \langle\text{diameter}\rangle\,. \end{aligned}$$

To extract a line object from a vector representation of a map, we consider each line segment that corresponds to some template description and, using that segment as a starting element in the grammar, attempt to assemble a chain of segments of the appropriate type. Starting from a single initial segment we search for other segments which, according to the grammar and template, may be used to extend the developing line object. During each (attempted) extension, the most recently identified segment of the chain is analyzed and an effort is made to connect its free point (the chain tail) to some other segment. If this attempt is successful, the newly added segment replaces the chain tail and the process is repeated until either an acceptable line termination pattern is found or continuation becomes impossible. In the latter case, the chain is reversed and an attempt made to continue it in the opposite direction. If the number of segments comprising the final isoline is insufficient, the chain is discarded. Attention then returns to the initial segment and attempts are made to assemble any other line types whose templates match the initial segment.

At each stage in the above process the developing line object is characterized by a description, associated with its tail, which contains its end point coordinates and orientation with respect to the horizontal. When seeking a continuation, the algorithm examines a local neighborhood of the corresponding end point, looking for vectors that might form part of the current object.

To simplify and speed up the line extraction process, the available vectors are first classified on parameters such as segment length, thickness, and type. Thirteen such classes have been found useful in the interpretation of topographic maps.[1] As the starting segment of a given line object, we choose a line segment bounded by ideally two but at least one end point. If both ends of the segment are free, the segment's length must fall within the predetermined limits stored in the appropriate template. If only one end is free, the segment must either be shorter than the upper limit specified or part of a simple graphical primitive whose length satisfies the first requirement.

Line assembly begins at an end point. During tracking two basic situations arise and must be resolved:

- continuation through a gap;
- continuation through a node.

In the first case, a segment of the same class, offset by an amount corresponding to the length of the expected gap, is sought. A square region of interest is defined with side length 2S, where S is the length of the longest gap allowed by the template. Only a part of the square is actually considered, its location and angular extent depending upon the slope of the last segment of the line (angle A in Fig. 6). All segments belonging to the desired class are considered first. If there are no segments of this class within the search area, segments from other classes are examined. Should there be no segments in the zone, the side length of the square is increased.

Segments are sought whose endpoints are close to the current chain tail and which are collinear with the current segment. All segments lying within the search area are examined and those whose parameters (length, width and distance from the tail) correspond to the current template are considered to be candidates for chain continuation. For each candidate we calculate the distance (D) and angular (A) measures illustrated in Fig. 7. The space of possible A and D values is divided into numbered zones. The zone number associated with each candidate is used to rank

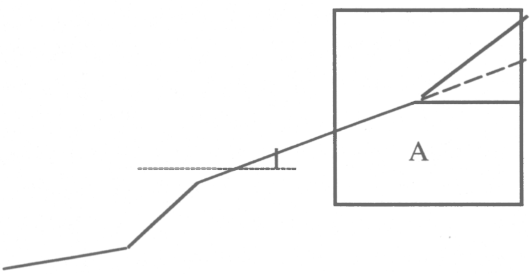


Fig. 6. Searching for a line continutation.

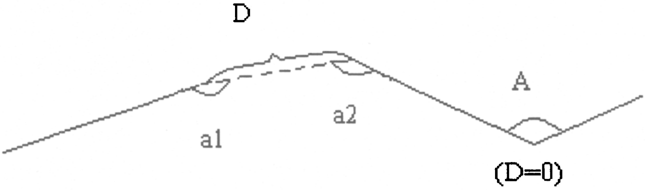
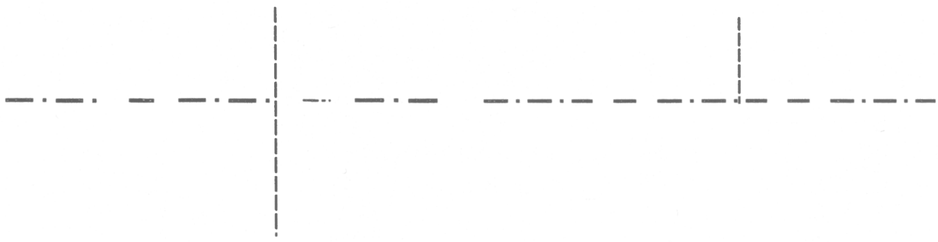


Fig. 7. Parameters of the potential continuations.



(a) (b)

Fig. 8. Degeneracy in the recognition of a dot-dashed line, (a) type 1, (b) type 2.

ɔossible extensions; lower zone numbers are preferred, with the nearest candidate ɔeing selected when two candidates lie in the same zone. The effect is to choose he segment with the most similar orientation. The chosen segment is then added ;o the developing line description and tracking continues.

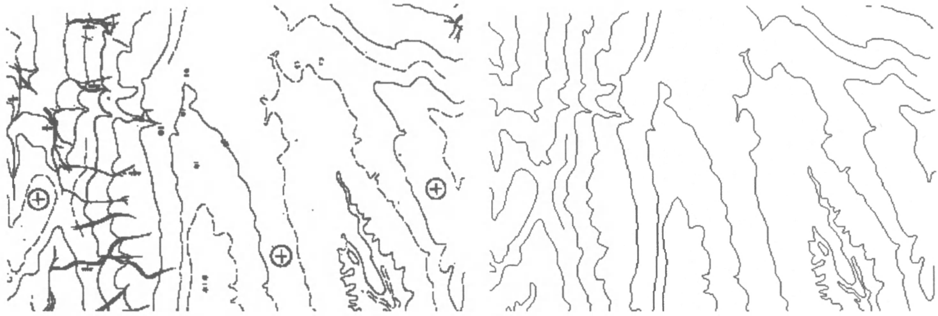
When the current line terminates in a node, the algorithm first examines all ;he vectors meeting at the node and seeks a segment that meets the template requirements of length and orientation. If no acceptable extension presents itself, we assume that we have found a type 1 degenerate case [Fig. 8(a)] and attempt ;o extend the tail line through the discontinuity. Tracking is terminated if we fail ;o find an extension of the line that satisfies the restrictions listed earlier or if we choose, as an extension, a segment whose vectors make a relatively small angle at ;he junction. In the latter case a type 2 degenerate case is signaled [Fig. 8(b)]. When this occurs, an attempt is made to extend the line in the opposite direction.

During the search for an extension to pure lines, the restrictions imposed on the choice of possibilities are removed in part, and we can concurrently test several possibilities for extending the line through nodes, end points, or in the potential degenerate cases. At this stage, it is permissible to extend lines through discontinuities nd nodes, even if the angle at the junction is not sufficient.

When searching for an extension through a discontinuity, we examine local nodes s well (using lists of connecting nodes attached to the vectors concerned), so as to detect type 1 degenerate cases. We also analyze carefully any situation in which ype 2 degeneracy might occur. All the possible extensions through the discontinuity are ranked on total length of the constituent segments and gaps and the shortest alternative is considered first. A possible extension is considered valid if it involves he end of another line or a vector that complies with the template. In this way, we find the shortest path joining the end of the line to elements included in the emplate description.

Search is terminated if the shortest alternative is longer than permitted by the emplate.

Figure 9 shows an example of isoline recognition. Interactive and automatic techniques were used to produce these results.



(a) (b)

Fig. 9. Isolines extracted (b) from an initial map image (a).

8. System Interfaces

The developed system MAPSCAN includes three main modules: the *Binarization module*, *Vectorization module*, and *Digitization module*.

MAPSCAN modules' implementation is based on the Multiple Document Interface (MDI) of the Windows system. The header of the main window of the system contains name and standard control buttons. The main menu of every module contains topics to define parameters and to call different functions of the module. At the bottom of the main window there is a status bar to display status messages of the module. An area between the toolbar and the status bar is used to display dialog and image viewing windows.

Inside of the main window different kinds of dialog windows are used to interact with the user. There are several types of controls in the dialog windows, they are:

- edit control to type a string of symbols;
- combo boxes and scrolled lists to choose a string from a list;
- check buttons with two states ("on" or "off") to set/reset appropriate processing modes;
- radio buttons (a set of round switches with "on" or "off" states) to choose one possibility from a mutually exclusive set of possibilities.

All dialog windows have Help buttons to get help information about these windows.

Dialog windows have the following components:

- Menu oriented components: most of the system functions are activated by the use of usual menu bars and pull-down menus.
- Icons, toolbars (with icons and icon palettes) are used in all three components of the system.
- Dialogue boxes: the setting and viewing of parameters and attributes are carried out using dialogue boxes in all three modules of the system.

All system actions are displayed on a monitor and are performed under the control of an operator. The operator can change system decisions at any time and can easily correct its results. The full details of the System User Guide is described in Ref. 22.

One of the drawbacks of traditional manual input methods is the difficulty in providing visual control to the digitization process. The user typically finds it very hard to tell which objects have been digitized and which are awaiting input. In the MAPSCAN system, this problem is solved by displaying raw (i.e. unprocessed) data on the screen in dim colors, and using bright colors to denote entities that have already been digitized. The user is provided with a range of input tools (the original image in raster form providing the background, microscope, moving glass, changing pick square, etc.) all implemented in software, and can select those that make his/her work more comfortable. The system window displaying both raster and vectorized data is shown in Fig. 10.

In addition to the direct digitization of map objects, the user has the ability to enter general map information to indicate control points (sheet corners and gratitude intersections) with their Latitude/Longitude or ground coordinates.

An example of the system interface is shown in Fig. 11.

The interface with the operating system is realized using function libraries. The system uses three types of data exchange formats:

- data exchange formats handling only raster images;
- data exchange formats handling both vector and raster data;
- data exchange formats handling only vector objects.

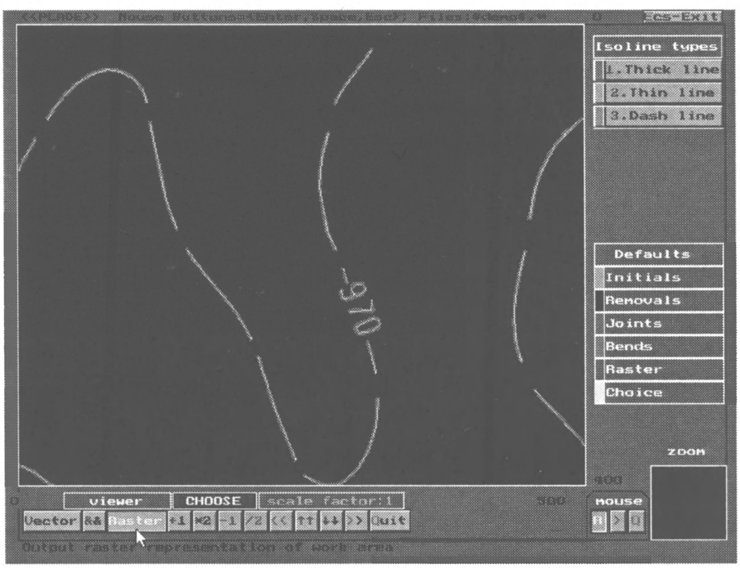


Fig. 10. The system window displaying both raster and vector cartographic data.

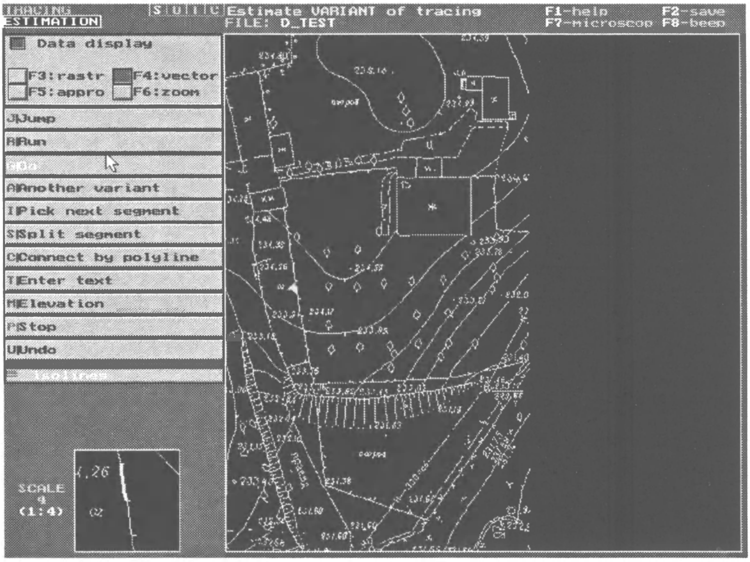


Fig. 11. Example of the system interface for the map digitizing process.

The system interfaces with other systems only through data exchange formats.

During the digitizing of map objects, the user indicates point-like objects, draws or traces line-like and area-like objects, and enters text inscriptions. Simultaneously the user sets the following object's attributes:

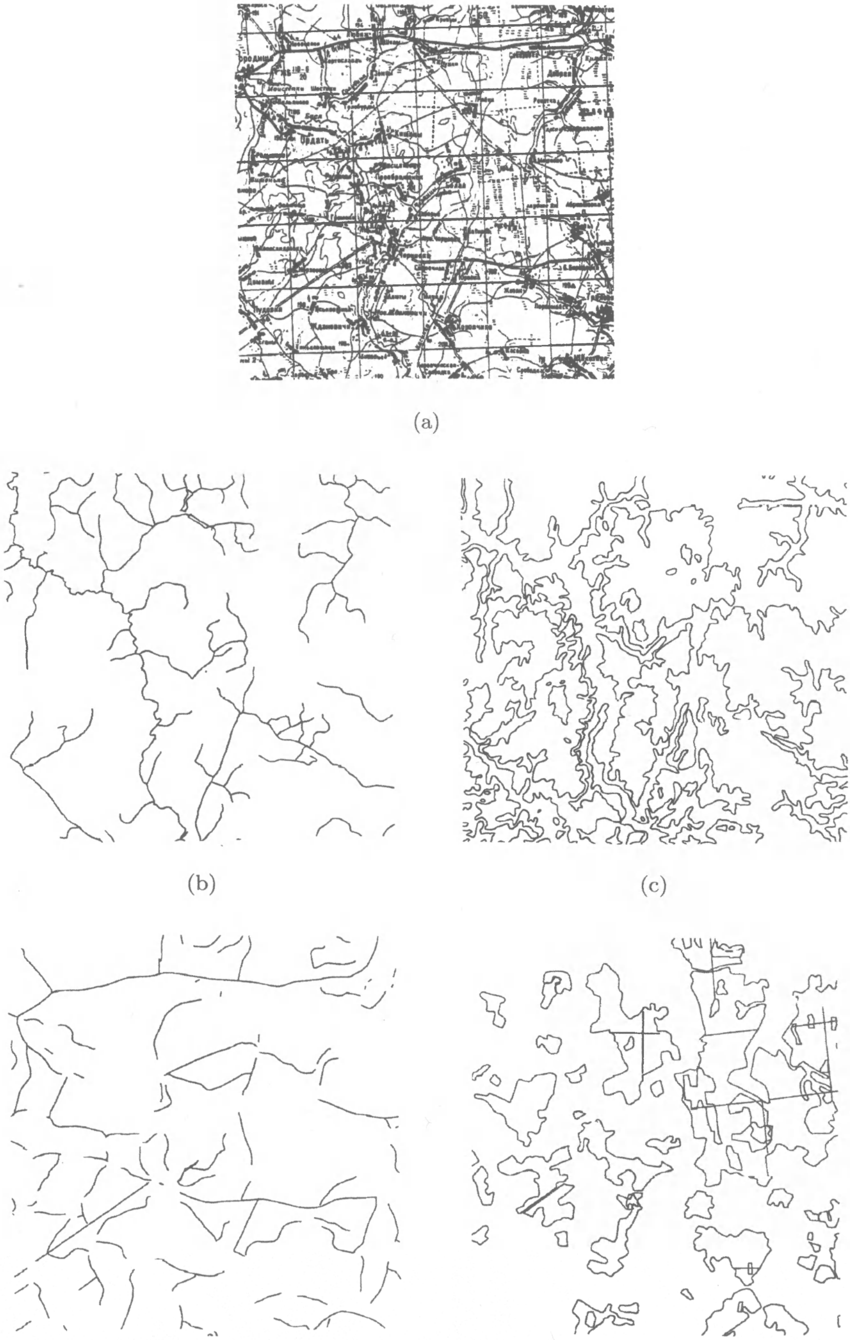
- major and minor codes;
- the height of the object, orientation angle for point-like objects and texts, text justification and size.

The *Digitization module* contains some facilities for allowing creation of data structures required by the output format. They are:

- the ability to include one vector segment (either vectorized by the *Vectorization module* or manually drawn in the *Digitization module*) into several elongated objects with the aim of representing some common parts of different map objects;
- the possibility of associating text with digitized objects, with the aim of representing semantic characteristics;
- the ability to use common points belonging to several objects simultaneously — such points can be calculated automatically in the case of object intersections, or can be entered using a suitable capture technique (approximate point indication in the limits of the search area).

9. Results and Discussion

The system MAPSCAN for map interpretation was implemented for a PC using the Windows platform. The input binary images were obtained from maps of size



(a)

(b) (c)

(d) (e)

Fig. 12. Example of map digitized by our system: (a) an initial map (it was originally in color), (b) the hydrography layer, (c) the isoline layer, (d) the road layer, and (e) the forest layer.

A4–A1, usually scanned at 400–800 DPI resolution. The raster data for processing are represented as PCX, TIFF, MSP, or other formats. Output data are represented in AutoCAD DXF files or other special map formats (MID/MIF, DVD).

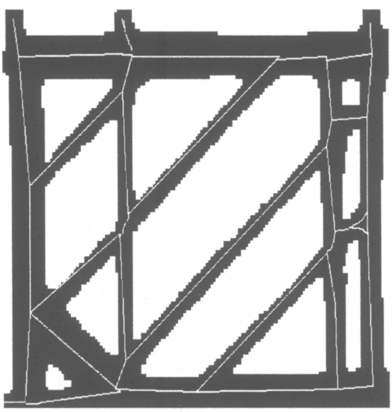
Figure 12 shows an example of the map parts digitized by our system: (a) an initial map (it was originally colored), (b) the hydrography layer, (c) the isoline layer, (d) the road layer, and (e) the forest layer.

Our system has been tested on processing of various types of maps and from various countries. Although it showed good quality results in general, we can say that different components of the system do not have the same level of achievement.

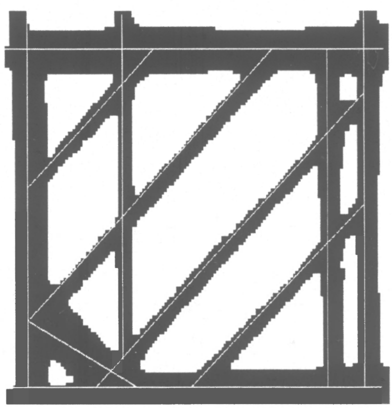
(i) Color image binarization shows good results for color maps of good enough quality. For most of the Russian maps that we processed, color separation was done sufficiently correctly and the vectorization module accepted the obtained binary images and vectorized them. However, in some countries and maps, color quality was bad which led to non-qualitative binary images and required many operations to enhance the quality of the obtained binary images.

(ii) Vectorization algorithms can be considered as most mature and stable in this technology. They were developed by using our long-term experience in the image vectorization area. However, it is always possible to see artefacts in thinned images. To minimize their influence, we developed a set of pruning techniques. An example of a edrawing section before and after pruning is shown in Fig. 13.

In spite of all our efforts, we cannot say that the vectorized image always has good quality. There are defects that influence further object recognition. This is one of the reasons why automatic recognition techniques cannot be applied after vectorization. Here, there is room for improvement and more sophisticated vectorization and pruning techniques like those described in Ref. 29.



(a)



(b)

Fig. 13. A drawing section (a) before and (b) after pruning.

(iii) Object interpretation is based on powerful interactive techniques with inserted automatic tools. We believe that this is the best solution in the interpretation of such complex images like maps. It works, and is quite stable and good, but of course, here there is much room for improvement. The main further steps should be to incorporate more automatic tools instead of interactive ones. Another improvement can be further development of the user interface. We plan to continue to work in both directions and integrate new automatic recognition methods when they become mature enough.

To digitize Russian topographic maps and map layers, resolutions of 300–1000 DPI were used. Our experience shows that in most cases 600–800 DPI is the more suitable resolution. If the chosen resolution is very low (e.g. 100–300 DPI) drawing objects will appear disconnected in the image. In our system, a resolution of 300 DPI can be used to process layers or maps containing area objects only.

If, on the other hand, the scan/image resolution is very high (e.g. more than 1000 DPI) each penstroke will generate a thick band of maybe 10–15 dark pixels. This can greatly increase subsequent processing time. In our system, a resolution of 1000 DPI was used very rarely for the processing of map parts containing complicated groups of objects, or very thin objects. In general, the resolution should be chosen after consideration of the map contents.

We used two types of scanners: drum scanner (A0 format) described in our paper,[20] and flatbed scanner (A4 format). Both scanners have resolutions that are enough for any map scanning. Only in the case of using the A4 scanner for inputting large-size maps should the scanned images be merged to form a full image. This can be done at a raster or vector image level and requires additional expenses. The choice of scanner size and type depends on the concrete tasks and money of the concrete user.

10. Conclusion

In this paper, a technology and system for automatic/interactive map interpretation is proposed. The main distinctions of the approach presented here from those already in existence are:

- The combination of different technological processes to digitize maps which allow a high level of automation;
- A fast and efficient color separation algorithm;
- A specialized image vectorization scheme, and original raster-to-vector transform techniques;
- Newly developed interactive object digitizing techniques.

The system was widely tested in interpretation of various map types and can be considered as a convenient and powerful tool in map digitizing applications.

References

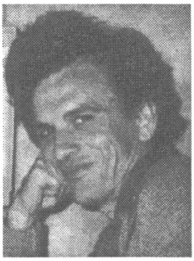
1. S. Ablameyko and T. Pridmore, *Machine Interpretation of Line Drawing Images* (Springer, 2000), p. 284.
2. N. Ebi, B. Lauterbach, and W. Anheier, "An image analysis system for automatic data acquisition from colored scanned maps," *Machine Vision and Applications* **7**, 148–164 (1994).
3. Q. Gao, J. He, G. Qiu, and Q. Shi, "A color map processing system PU — GMPS," *Proc. Int. Conf. Document Analysis and Recognition*, Japan (1993), pp. 874–877.
4. L. Li, G. Nagy, A. Samal, S. Seth, and Y., Xu, "Integrated text and line-art extraction from a topographic map," *Int. J. Document Analysis and Recognition* **2**(4), 177–185 (2000).
5. L. Boatto, V. Consorti, M. del Buono *et al.*, "An interpretation system for land register maps," *Computer* **25**(7), 25–33 (1992).
6. K. Stapor, "Recognition of cartographic symbols based on structural model of shape," *Machine Graphics and Vision* **8**(1), 129–142 (1999).
7. K. Stapor, "Geographic map image interpretation — survey and problems," *Machine Graphics and Vision* **9**(1/2), 497–518 (2000).
8. J. M. Ogier, S. Adam, A. Bessaid, and H. Bechar, "Automatic topographic color map analysis system," *Proc. 4th Int. Workshop of Graphics Recognition* (Canada, 2001) pp. 229–244.
9. R. D. T. Janssen and A. M. Vossepoel, "Adaptive vectorization of line drawing images," *Computer Vision and Image Understanding* **65**(1), 38–56 (1997).
10. G. Agam, H. Luo, and I. Dinstein, "Morphological approach for dashed lines detection, Graphics Recognition," (eds.) R. Kasturi and K. Tombre, Lecture Notes in Computer Science **1072**, 92–105 (1996).
11. C. P. Lai and R. Kasturi, "Detection of dashed lines in engineering drawings and maps," *Proc. 1st Intl. Conf. Document Analysis and Recognition* (1991), pp. 507–515.
12. IAPR TC10 Graphics Recognition home pages: http://graphics.basit.com/iapr-tc10/products/.
13. T. Watanabe and T. Fukumura, "Towards an architectural framework of map recognition," *Proc. 2nd Asian Conference on Computer Vision* (Singapore, 1995), pp. 617–621.
14. S. Ablameyko, V. Bereishik *et al.*, "System for automatic vectorization and interpretation of graphic images," *Pattern Recognition and Image Analysis* **3**(1), 39–52 (1993).
15. P. M. Devaux, D. B. Lysak, and R. Kasturi, "A complete system for the intelligent interpretation of engineering drawings," *Int. J. Document Analysis and Recognition* **2**(2–3), 120–131 (1999).
16. S. Satoh, H. Mo, and M. Sakauchi, "Drawing image understanding system with capability of rule learning," *Proc. 2nd Int. Conf. Document Analysis and Recognition* (Japan, 1993) pp. 119–124.
17. S. Suzuki and T. Yamada, "MARIS: Map Recognition Input System," *Pattern Recognition* **23**, 919–933 (1990).
18. R. Kasturi *et al.*, "A system for interpretation of line drawings," *IEEE Trans. Pattern Analysis and Machine Intelligence* **12**, 978–992 (1990).
19. S. Shimotsuji *et al.*, "A robust recognition system for a drawing superimposed on a map," *IEEE Computer* **25**, 978–992 (1992).
20. S. Ablameyko, B. Beregov, A. Kryuchkov, "Automatic map digitizing: problems and solution," *IEE Computing and Control Engineering Journal* **5**(1), 33–39 (1994).

21. S. Ablameyko, V. Bereishik, O. Frantskevich, M. Homenko, and N. Paramonova, "Interactive interpretation of map-drawing images," *Proc. IAPR Workshop on Graphics Recognition* (France, INRIA, 1997), pp. 185–193.
22. MAPSCAN User Guide, Minsk, IEC, p. 84 (2000).
23. K.-H. Lee, S.-B. Cho, and Y.-C. Choy, "Automated vectorization of cartographic maps by a knowledge-based system," *Engineering Applications of Artificial Intelligence* **13**(2), 165–78 (2000).
24. C.-C. Shih and R. Kasturi, "Extraction of graphic primitives from images of paper-based drawings," *Machine Vision and Applications* **2**, 103–113 (1989).
25. D. G. Elliman and M. Sen-Gupta, "Automatic recognition of linear features, symbols and textured areas within maps," in *Proc. IAPR Workshop on Machine Vision Applications* (1994), pp. 239–242.
26. D. Dori *et al.*, "Sparse-pixel recognition of primitives in engineering drawings," *Machine Vision and Applications* **6**(2 and 3), 69–82 (1993).
27. D. Dori, L. Wenyin, and M. Peleg, "How to win a dashed line detection contest, graphics recognition," (eds.) R. Kasturi and K. Tombre, Lecture Notes in Computer Science, **1072**, 286–300 (1996).
28. W. Y. Liu and D. Dori, "A protocol for performance evaluation of line detection algorithms," *Machine Vision and Applications* **9**(5 and 6), 240–250 (1997).
29. E. Bodansky and M. Pilouk, "Using local deviations of vectorization to enhance the performance of raster-to-vector conversion scheme," *Int. J. Document Analysis and Recognition* **3**, 67–72 (2000).

Sergey Ablameyko was born in 1956. Received Dip.Math in 1978, Ph.D. in 1984, DSc in 1990, Prof in 1992). He is the head of the Image Processing and Recognition Laboratory and Deputy Director of the Institute of Engineering Cybernetics of the National Academy of Sciences of Belarus, and half-time Professor at the University of Informatics and Radioelectronics. He has more than 200 publications including six authored/co-authored books (one published by Springer, one by SPIE) and eight edited books in image processing and pattern recognition. In his academic career he was a visiting scientist in Italy, Japan, Sweden, Finland, England, Germany, UK, Greece and Spain.

He was a chair/co-chair, member of Programm Committees of numerous International conferences held worldwide. He is in the Editorial Board of two international journals: *Pattern Recognition, Pattern Recognition Letters and Machine Graphics and Vision.*

He is a Fellow of the IEE, Senior member of the IEEE, Fellow of the IAPR, Fellow of two Academies, Chairman of the IEE Belarusian Centre. He is a member of the Governing Board of the International Association for Pattern Recognition, President of the Belarusian Association for Image Analysis and Recognition.



Vladimir Bereishik was born June 12, 1957. Received his diploma of higher education in applied mathematics from Belarussian State University, Minsk, in 1980. He has been with the Institute of Engineering Cybernetics (IEC), Belarussian Academy of Sciences, since 1980 and also researcher at the Laboratory of Image Processing and Recognition. His scientific interests are binary image processing, 2D line-drawing interpretation, recognition of graphical entities and cartographic objects. He is the author of about 30 publications, including one book (in co-authorship), 15 papers and five technical reports.

Maria Homenko was born November 12, 1953. Received her diploma of higher education in mathematics from Belarussian State University, Minsk, in 1976. She has been with the Institute of Engineering Cybernetics (IEC), Belarussian Academy of Sciences, since 1987. Currently, she is a leading engineer at the Laboratory of Image Processing and Recognition. Her scientific interests are image processing and recognition. She is the author of 12 publications.

Dmitry Lagunovsky was born in Minsk, Belarus, in 1970. He graduated from Belarussian State University, Radiophysics and Electronics department in 1993. In 1998, he received the Ph.D. in computer science for the development of fast edge detectors for industrial applications. Till 2000, he was at the Institute of Engineering Cybernetics, Belarussian Academy of Sciences as a senior research scientist. Currently he is head of the Vision Department at Semitech company. He has two published books and more than 20 papers on image processing. His scientific interests are fast techniques of image processing and industrial applications of image processing.

Nadezhda Paramonova was born November 24, 1946. Received her diploma of higher education in electrical engineering from the Radio Engineering Institute, Minsk, in 1970. She has been with the Institute of Engineering Cybernetics (IEC), Belarussian Academy of Sciences, since 1971. She is currently a researcher at the Laboratory of Image Processing and Recognition. Her scientific interests are image processing and 2D line-drawing interpretation. She is author of about 20 publications.

Oleg Patsko was born 21 August, 1965. Received his diploma of higher education in applied mathematics from Belarussian Polytechnic Institute, Minsk, in 1987. He has been with the Institute of Engineering Cybernetics (IEC), Belarussian Academy of Sciences, since 1987. He is also a researcher at the Laboratory of Image Processing and Recognition. His scientific interests are color image processing, 2D line-drawing interpretation, recognition. He is the author of about 20 publications.