AUTOMATIC DATA ACQUISITION FROM TOPOGRAPHIC MAPS USING A KNOWLEDGE-BASED IMAGE ANALYSIS SYSTEM*

N. Ebi, B. Lauterbach, Ph. Besslich
Working Group Digital Systems,
University of Bremen, FB1,
P.O.Box 330440, D-2800 Bremen 33, Germany

ABSTRACT:

A system for automatic data acquisition from topographic maps using knowledge-based image analysis methods is presented. The investigations are part of the interdisciplinary project *Environmental Planing System*. The goal is to generate a symbolic description of the map contents that may be imported into the ARC/INFO GIS for supporting the geographical tasks of the project.

High resolution color scanned topographic maps (scale 1:25 000 and 1:5 000) serve as a data source. Binary color map layers are produced by a HSI color space guided multi-level segmentation. The processing of each layer includes vectorization as well as application of methods like neural network-based symbol and object recognition for the extraction of attributed structure primitives. Subsequent analysis is based on a hierarchical structuring of the map scene with map objects and their relations. A frame mechanism is utilized for modeling the concepts of all types of map objects. A control module driven by the data model supervises the creation of instances of the concepts. The map objects located at the lowest hierarchy level correspond to the attributed structure primitives. The interface to ARC/INFO is represented by the instances of the upper levels of hierarchy.

KEY WORDS: Topographic Maps, Raster Image Analysis, Attributed Structure Primitives, Knowledge-Based Interpretation, Data Base, GIS

1. INTRODUCTION

Geographic information systems (GIS) are gaining importance for environmental planning tasks. For an efficient and flexible use of these systems it is necessary to combine data acquisition, creation of a valuation scheme and GIS in an integrated concept. This is the objective of the interdisciplinary project *Environmental Planing System* *. For an efficient spatial valuation it is necessary to have an adequate data base. Maps are an important source of information for this data base. At present a lot of relevant maps have still to be digitized manually, which is a time consuming and error prone process. To improve the situation we are developing image analyzing methods for automatic data aquisition from maps. These methods are described in this paper as part of the above mentioned project. As primary data German topographic maps of scale 1:25 000 and 1:5 000 are used.

2. SYSTEM OVERVIEW

An overview of the proposed system is shown in Fig. 1. The system kernel contains modules for storage, modification, manual digitization, graphic representation and evaluation of spatial data. Considering the complexity of the system kernel we preferred to use a commercial GIS. We decided to use ARC/INFO from ESRI, because it is a powerful tool and is in wide-spread use over Germany.

The acquisition of spatial data is done by means of knowledge-directed map analysis and interactive input of additional information obtained from soil analysis, remote sensing techniques, terrain mapping or other sources. The system user will have the opportunity to evaluate the spatial information via an expert system connected to the data base.

The principle of the proposed knowledge-directed image analysis is shown in Fig. 2. A raster image of the map is created using a color scanner. Symbolic image information is extracted by splitting the map image into color layers which are processed using raster object recognition and vectorization. The extracted symbolic information, called *attributed structure primitives* serves as data source for the knowledge-directed image analysis. The analysis is realized by the control module, the image model, the instance storage and the image based conflict solving module. The instances represent the extracted map information and have finally to be converted to the ARC/INFO data base format.

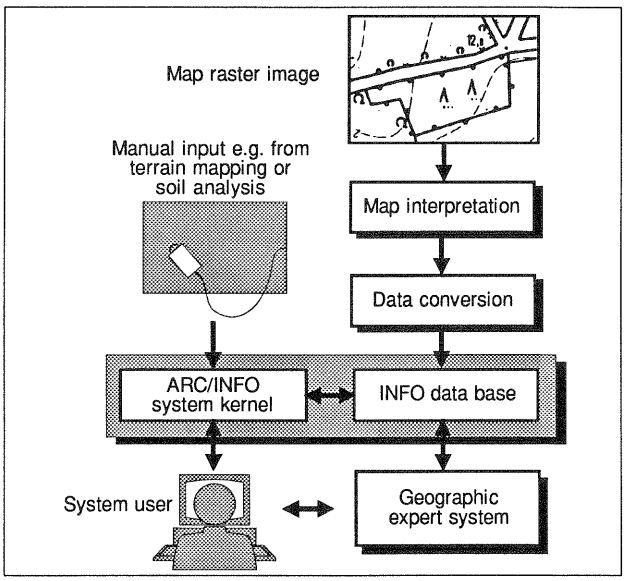

Fig. 1: System overview.

3. SCANNING OF TOPOGRAPHIC MAPS

The topographic maps we use are mainly printed in four colors printing technique (Schoppmeyer, 1991). The colors are cyan, magenta, yellow and black. Therefore, a 24-Bit-RGB color scanner is necessary to create the raster images. The smallest objects contained in the maps are

* This work was supported in part by DFG (German Research Association) under contract Stäblein/Besslich: Environmental Planing System (Sta 126/19-1).

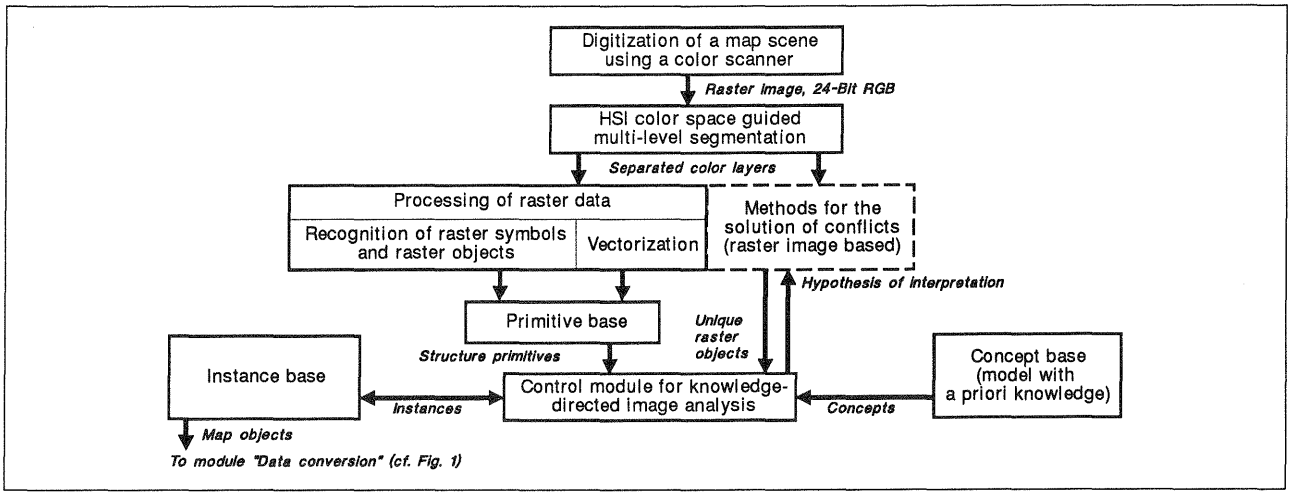

Fig. 2: Proposed scheme of knowledge-directed image analysis.

raster dots (e.g. meadow texture) of size 0.05 to 0.1 mm. The minimum scanning resolution is therefore 800 to 1000 dpi. We use an OPTOTECH overhead repro scanner which is connected to the host computer via SCSI interface. The scanner allows a resolution of 2000 dpi or more in all scanning modes.

Using the technique described in this paper, it is not necessary to scan a whole map or the complete set of binary map layers of which the final map is produced. It is possible to scan and process only the area of interest in the final map. If binary map layers are used, the complete layers have to be digitized. Otherwise there would not be any control marks available to support spatial registration of raster data.

4. PROCESSING OF RASTER DATA

The raster data of the map is processed to create a symbolic attributed description (*attributed structure primitives*) of all basic elements contained in the map (vectors, symbols, regions). This is done in five steps. The first step is the separation of the color layers contained in the map (cf. Fig. 3).

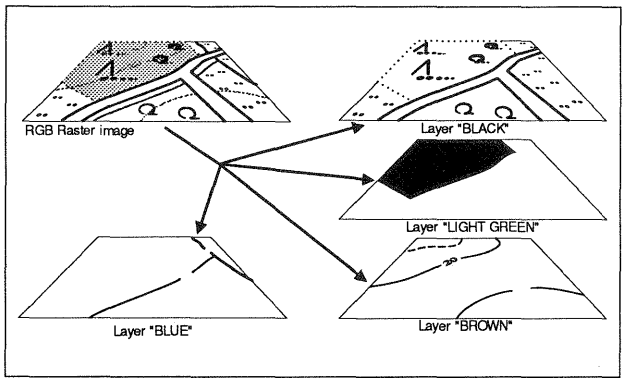

Fig. 3: Separation of color layers.

The separation corresponds to the reverse process of the composition of binary map layers during printing. In opposition to map production process, the separation will result in layers containing information printed using the same color instead of layers containing information of the same type (e.g. symbols, roads). The second step is the recognition of raster symbols and objects in the color layers. Recognized raster symbols and objects are then removed from the layers. The third step is separation of layers containing mainly region and line information. These two types of layers have to be processed in a different way. A different strategy has to be used to cut off symbols (e.g. houses) from lines (e.g. road borders) if they are connected to each other.

The fourth step consists of vectorization of the line layers. The region layers are processed using a contouring technique. In the final step of raster image processing, vector data is refined to reduce redundancy.

4.1 Separation of color layers

The color layer separation is based on an unsupervised classification technique. The location of clusters in color space is different for every map. It depends on the type of the map, the production process, the condition of the map, the type of illumination and the spectral characteristics of the scanner unit. The number of clusters to be created, i.e. the number of colors contained in the map has to be defined manually. Typically there are five or six colors contained in a topographic map including background. The separation process may be subdivided into four principal steps:

- Preliminary determination of color class centers using a 3-D histogram,
- Improvement of the cluster center positions using a topological colormap technique based on Kohonen's self organizing feature maps (Lippmann, 1987),
- Classification of map raster data, and
- Region growing of classified data to remove unclassified regions.

For a preliminary determination of centers of color clusters a 3-D histogram is calculated from

$$H_{ijk} = \sum_{y=1}^{n} \sum_{x=1}^{m} d\left((r_{xy} - i), (g_{xy} - j), (b_{xy} - k)\right) \qquad (1)$$

$$\text{with} \quad d(a,b,c) = \begin{cases} 1 & \text{for } a = b = c = 0 \\ 0 & \text{otherwise} \end{cases},$$

where H is the 3-D histogram with indices i, j, and k. m and n are the image dimensions and r_{xy}, g_{xy} and b_{xy} are the color stimulus values of the pixel represented by the coordinates x and y. Although the color stimulus values are in the range [0,255] they are limited to [0,31] by a right shift operation due to restrictions in memory space and processing time.

After calculating the 3-D histogram, all maxima of the histogram are determined. This is done using an algorithm described in (Chaudhuri et al., 1986) which we expanded for 3-D arrays. The idea in the algorithm is to find values

in the 3-D histogram that are not maxima, find connected components of same value and delete them. The undeleted values constitute the local maxima.

The number of histogram maxima has to equal the number of predefined map colors. Therefore, the existing maxima have to be coalesced iteratively until this condition is satisfied. This is done using the following procedure:

1) Calculation of a distance measure of the maxima to all the others using a weighted sum of the distance in the RGB and IHS color space. The intensity value i, hue value h and saturation value s are determined using the following equations derived from (Data Translation, 1989):

$$i = \frac{r+g+b}{3}, \tag{2}$$

$$s = 31.0\, s_t \tag{3}$$

$$\text{with} \quad s_t = \begin{cases} 0 & \text{for } r=b=g \\ 1 - \dfrac{\min(r,g,b)}{i} & \text{otherwise} \end{cases} \quad \text{and}$$

$$h = 4.9338\, h_t \tag{4}$$

$$\text{with} \quad h_t = \begin{cases} 0 & \text{for } r=b=g \\ \dfrac{\pi}{2} + \tan\left(\dfrac{r-0.5g-0.5b}{0.5\sqrt{3}\,(g-b)}\right) & \text{otherwise} \end{cases}.$$

These equations include a scaling of i, h and s to the range [0,31]. The distance d_{rgb} between two maxima in the RGB color space is determined by

$$d_{rgb} = \sqrt{\Delta r^2 + \Delta g^2 + \Delta b^2}\,. \tag{5}$$

The distance d_{ihs} between two maxima in the IHS color space is determined by

$$d_{ihs} = \sqrt{\Delta s_n^2 + \Delta i^2 + h_n^2} \tag{6}$$

$$\text{with} \quad s_n = \frac{s}{2} \quad \text{and} \quad h_n = d_h \frac{s}{32},$$

where d_h is the minimum angular distance (clockwise or counter-clockwise) between two hue values. Finally the distance measure d is calculated by

$$d = d_{rgb} + d_{ihs}\,. \tag{7}$$

2) Mark couples consisting of m maxima that have to be coalesced in a coalition table. m is determined by

$$m = m_a - m_p\,, \tag{8}$$

where m_a is the present number of maxima and m_p is the predefined number of colors including the background. If the distance measure of a couple of maxima is less than a predefined threshold, this tuple is a privileged candidate for coalition.

3) The table with marked couples is processed recursively to find tuples of maxima that have to be coalesced.

4) A new maximum is created out of the previously grouped maxima using the following equation:

$$c = \frac{1}{H_s} \sum_n c_n H_n \quad \text{with } H_s = \sum_n H_n\,, \tag{9}$$

where H_n is the histogram value of the maximum n of the processed tuple, c_n is the R, G or B value of the maximum n and c is the resulting new R, G or B value.

5) If the present number of maxima after the coalition is above the number of predefined colors, the algorithm is continued with step 1.

The next step in classification is improvement of initial estimates of cluster centers. This is done using a topological color map algorithm (Springub et al., 1990) which is based on Kohonen's self organizing feature map. The color map size is set to $n_c * n_c * 3$ with

$$n_c = f_{ceil}(\sqrt{m_p}) \tag{10}$$

$$\text{with} \quad f_{ceil}(x) = \begin{cases} x & \text{if the fractional part} \\ & \text{of x is equal 0} \\ x+1 & \text{otherwise} \end{cases}$$

In contrast to the algorithm of Springub, the color map is not initialized with random values but with the predetermined cluster center positions, and the color map size is kept constant. The color map algorithm is executed in the following iterative steps:

1) Creation of an index image. The index of the color map entry with minimum Euclidian distance will be assigned to each pixel.

2) Calculation of mean value x of color stimuli of each index group.

3) Calculation of new color map entries z_{t+1} using the equation

$$z_{t+1} = z_t + w_t (x - z_t)\,, \tag{11}$$

where z_t is the present value of a color map entry and w_t is a weight factor, which is calculated from the present number of color map hits h_p in this iteration and the overall number of color map hits h_o for this entry using

$$w_t = \frac{h_p}{h_o + h_p}\,. \tag{12}$$

Using this weight factor the influence of large clusters to the learning process is smaller. Therefore, suppression of small clusters produced by small color spots is prevented.

4) Creation of a new index image.

The iterative steps 2 to 4 are repeated until less than 5% of the pixels of the new index image have changed compared to the index image of the previous iterative step. After finishing the iterations, the cluster center positions are determined.

The classification is done using a minimum distance classificator with a fixed rejection threshold (Richards, 1986). For each cluster the rejection radius is set to a value that guarantees that cluster spheres are not overlapping. Thus, the radius r_i of cluster i is set to

$$r_i = 0.45 \min(d_{ij}) \quad \text{for all } j \neq i\,, \tag{13}$$

where d_{ij} is the distance between the centers of clusters i and j. This kind of classification will result in a lot of unclassified pixels mainly on the borders between two regions of different colors. If a classification of these border pixels is done using a larger rejection threshold, this will result in a high misclassification rate and in the creation of corroded regions of different colors along the object contours. In this case the following image analysis algorithms would extract a lot of wrong information that prevents the knowledge-directed system from a sensible data interpretation. Therefore, it is better to fill the unclassified regions by a region growing technique. This is done using the following algorithm for each unclassified pixel:

1) Calculation of a histogram of the 8-element neighborhood.

2) If the pixel has no classified neighbors, continue with step 1 for the next pixel.

3) If the pixel has already classified neighbors, the class of the most frequent neighbor will be assigned.

4) If there are several histogram entries having the same frequency, the class of the pixel located in the northwest direction will be assigned.

The steps 1 to 4 are repeated until all image pixels are classified.

From the classified image m_p-1 binary color layers may be separated. Some of these layers still include textured regions or they have some defects caused by overprinting with other layers. If for example a tree symbol (black) is printed over a wood region (light green), the assignment of the symbol to the black layer will result in an equally shaped defect in the light green layer. These defects may be corrected using region growing techniques with a defined set of rules, as for example,

Set a 0-pixel in the light green layer to 1 if it belongs to a closed 0-region and there is a 1-pixel either in the black or brown layer.

A textured region like a lake area, which is printed using blue raster dots, may be filled using structural texture analysis methods in combination with a texture element grouping algorithm (Fumiaki et al., 1990). With this step the separation of color layers is completed.

4.2 Recognition of raster symbols

A rotation and size invariant recognition of separate, not overlapping raster symbols and objects (e.g. tree symbols, characters) can be obtained using a neural network based technique (Lauterbach et al., 1991). The major algorithm extracts rotation and size invariant feature vectors based on polar distance measures. Several types of these measures may be combined for the classification of a single raster symbol or object, for example

- the distance from the center of gravity (CG) of the raster object to its outmost border,
- the distance from the CG to the change of first pixel value,
- the sum of the raster object pixels counted from the CG.

All these measurements are determined for a predefined number of directions depending on the object size. The direction for the polar measurements starts from the main axis of inertia of the object, using additional contour or diameter measurements that are necessary to distinguish between an object rotation of $\pm\pi$.

The feature vectors are evaluated using a hierarchical structure of multi-layer perceptrons. There is one perceptron for the direct evaluation of each feature vector (stage-1 network). The number of network inputs corresponds to the vector size, the number of outputs corresponds to the size of the object set. The outputs of the stage-1 networks are combined using the following equation

$$o_n = \frac{1}{o_{max}} \sum_{m=1}^{n_f} i_{mn}\, w_m , \quad \text{with } w_m = \frac{ls_{min}}{ls_m} , \qquad (14)$$

where n is the index of the output or input unit, m is the index of the stage-1 network and n_f is the overall number of the stage-1 networks. o_{max} is the output with the maximum activity. w_m is a weight factor, ls_m is the number of learning steps necessary to train the stage-1 network m and ls_{min} is the minimum number of learning steps that has occured.

The output of this combination stage is fed into a further perceptron, which makes the final decision about the raster object classification. After recognizing a raster object, it is deleted from the layer and the recognition result is put into a temporary data base, where it is available for further interpretation.

4.3 Separation of region-based and line-based layers

The region data and the line data included in a layer has to be processed in different ways. Region data must be contourized while line data must be vectorized. Thus, for every layer it is necessary to detect whether it contains mainly region or line structures.

This task is performed using a distance histogram based on a medial axis transformation (Pavlidis, 1987). The histogram values D_i are calculated using the equation

$$D_i = \sum_{y=1}^{n} \sum_{x=1}^{m} d\,(f_{med}(p_j(x,y)) - i) \qquad (15)$$

$$\text{with} \quad d(x) = \begin{cases} 1 & \text{for } x = 0 \\ 0 & \text{otherwise} \end{cases} ,$$

where m and n are the image dimensions and $p_j(x,y)$ is the value of the pixel represented by the coordinates x and y in the layer j. Function f_{med} yields the minimum distance of the pixel at position (x,y) from the raster object border.

The histogram of a line-based layer has a tall shape whereas a region-based layer yields a wide histogram.

4.4 Vectorization

Vectorization is performed on one pixel wide line structured images. Therefore, the region-based layers have to be contourized. This is done using a contour tracing algorithm described in (Pavlidis, 1987). The line-based layers have to be thinned before vectorizing them. Most line thinning algorithms are critical to use, because they produce a number of short line fragments connected to the skeleton which do not really exist in the line image. Therefore, we use an algorithm which is not very fast but produces a clean medial line of the raster objects in the input image. This algorithm is based on a smoothing and stripping technique with a skeleton adjustment to the medial line of the pattern (Chu et al., 1986).

The vector data is based on nodes and vertices. In the first vectorization step the nodes are extracted from the line image. A node is represented by a pixel that has either less or more than two neighborhood pixels belonging to a line segment. The second step is the conversion of the line segments connecting the nodes into Freeman chain codes. Some line structures like circles cannot be converted to nodes and segments because they consist only of pixels with two neighbors. These line structures are converted in the third vectorization step. The vertices connecting the nodes are created using a split and merge technique on the Freeman coded line segments. Nodes that are directly neighbored in the raster image have to be coalesced and the vertices connected to them have either to be corrected or deleted. Finally, attribute data like color, line width or variance of the line width is extracted from the raster image for each vertex.

4.5 Refinement of vector data

Although the skeleton created by the line thinning algorithm of (Chu et al., 1986) is of high quality, there may be some unnecessary lines and nodes in the thinned image. These lines will also be vectorized. They may be removed

in a vector data refinement step. This can be obtained applying a set of rules. In a first step short branch vertices (cf. Fig. 4, vertice A) are removed. Subsequently correspondingV-shaped arrangements of vertices (node B) are straightened by deleting the center node. After refining the vector data, the nodes and vertices will be stored in the primitive base.

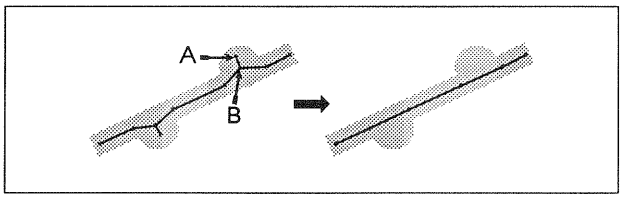

Fig. 4: Illustration of vector refinement.

5. BASIC IDEA OF KNOWLEDGE-DIRECTED IMAGE ANALYSIS

The attributed structure primitives extracted by the raster image processing methods described above are the data source for subsequent analysis strategy. The description level of the analysis is based on a hierarchical structuring of the map with map objects and relations between these objects. These relations may be of topological type as well as of thematical type. Associative nets (Quillian, 1969; Brachman, 1979; Minsky, 1979; Hayes, 1979) based on frames are used as a formalism for representation of knowledge which is characterized by these objects and their relations.

For the following description of the system it is important to distinguish between map objects representing more or less complex cartographic facts (e.g. terrain area) and simple raster objects. The structure primitives are defined at the lowest hierarchy level. At higher levels a map object represents the composition of one or more map objects of the lower levels of abstraction.

Knowledge-directed image analysis tries to attach a meaning to an image scene. One way of doing so is to use an explicit *model* of what the image can contain and then construct a mapping between the model and the image. Hereby the model represents the necessary *a priori* knowledge. Since a particular image scene is only an instance of the class of possible image scenes, the model must be in some sense larger than the image. That is, a useful model of the domain of the image typically contains a large amount of information on possible image contents. The mapping processed during analysis normally uses only a small part of the model. This surplus of knowledge guarantees a flexible image interpretation.

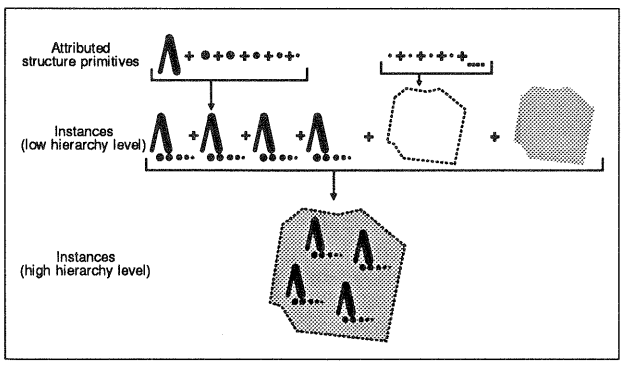

Fig. 5: Example of a hierarchical map description (coniferous forest).

Fig. 5 shows the principle of the hierarchical map description by example of a coniferous forest. The map object *coniferous forest* consists of a combination of coniferous tree objects, forest border and the forest area signature. Each coniferous tree object is again described by a composition of an inverted V symbol and several dots in a defined topology. The forest border is composed of a sequence of dots. Hereby, the area signature, the dots and the inverted V symbols are the attributed structure primitives.

6. KNOWLEDGE REPRESENTATION

6.1 Concepts and Instances

The *a priori* knowledge necessary for map interpretation is provided by a model acting as long term memory. As mentioned previously, an associative net serves as knowledge representation scheme. The basic structure of the net is the data structure *concept*. A concept contains the intensional description (Sagerer, 1985) of a term which is necessary for the model of the given problem. The intension of a term is the abstract definition of its meaning. It includes a characterization of properties which must be satisfied by a concrete fact to be valid for this term. On the other hand the extension encloses the set of all concrete facts of a case which satisfy the definition of meaning. The elements of the extensional set of a term are called *instances* of the corresponding concept. For applications of map interpretation, the concepts represent cartographic objects as well as abstract notions necessary for solving conflicts in interpretation. As an example the concept *virtual continuation of a contour line* may be considered. This clause characterizes connections of contour lines that cannot be derived from the existing attributed structure primitives in case of overlapping line segments.

In the present state of our system the intensional description of a concept is given completely by *necessary parts*, *structure relations* and *attributes*. For generation of an instance of a concept the following conditions have to be considered:

- Instances of concepts have to be made available. These instances are related to the concept to be instantiated by the relation *necessary part*.
- The defined structure relations have to be satisfied.

If both conditions are met, the valuation of the possible instance is performed. This valuation is a measure for the similarity of the instance with the intensional description, i.e., the concept. The valuation represents the certainty factor (cf) for the membership of an instance to the set of realizations of the concept. Thus, the valuation depends on the actual problem and is therefore a part of the *a priori* knowledge given by the model. The procedure to obtain the valuation of an instance has to be defined within the concept. The instantiation is successful if the valuation is above a threshold cf_{th} also defined within the concept. In this case, the attributes of the instance will be evaluated using information in the concept. The instance is then stored in the instance base, which acts as a short term memory. A reference to the instance is also made available within the concept.

Concerning evaluation of the structure relations, attributes and valuations, the model encloses declarative knowledge as well as procedural knowledge, i.e., algorithms. Using the procedural knowledge, a quantitative characterization of the qualitative facts of a case represented by the associative net may be performed. For this purpose, the structuring of the instances is analogous to the one of the concepts. The procedures of the model correspond to concrete values of the instances.

6.2 Frames

Frames (Minsky, 1979; Rich, 1983; Harmon et al., 1985) are used for representation of both concepts as well as instances. The aspects of an instance or a concept are described with a set of *slots*. These slots may be filled by other frames describing different aspects. An inheritance mechanism is integrated in the frame description of a concept. That is, more concrete concepts summarize their own slots and those of the concepts at higher levels. This inheritance economizes redundancy in defining concepts. In case of more concrete concepts only some specific declarations are needed. The inheritance is realized by the slot *generalization*. This relation *generalization* and the relation *necessary part* along with their inversions represent hierarchies of the net. Fig. 5 shows the hierarchy with regard to the relation *necessary part*.

A simplified definition of the concept for a coniferous tree is presented in Fig. 6. The corresponding ideal shape of the coniferous tree is shown in Fig. 7.

CONCEPT *Coniferous Tree*

Generalization	value:	{Tree}
Necessary Parts	value:	{T, D1, D2, D3, D4, D5}
Structure Relations	value:	{SR1, SR2, SR3, SR4, SR5}
Attributes	value:	{Position, Size}
CF	value:	[(sr1 & sr2 & sr3 &sr4 & sr5) ifTrue: [cf = 100] ifFalse: [cf = 0]]
	arguments:	{(SR1), (SR2), (SR3), (SR4), (SR5)}
CF-Threshold	value:	99

T	value:	{Inverted V}
	restriction:	nil
D1	value:	{Dot}
	restriction:	[10 < diameter < 12]
D2	value:	{Dot}
	restricton:	[8 < diameter < 10]
D3	value:	{Dot}
	restriction:	[6 < diameter < 8]
D4	value:	{Dot}
	restriction:	[4 < diameter < 6]
D5	value:	{Dot}
	restriction:	[2 < diameter < 4]
SR1	value:	[Procedure S1]
	arguments:	{(T Position), (D1 Position)}
SR2	value:	[Procedure S2]
	arguments:	{(T Position), (D2 Position)}
SR3	value:	[Procedure S3]
	arguments:	{(T Position), (D3 Position)}
SR4	value:	[Procedure S4]
	arguments:	{(T Position), (D4 Position)}
SR5	value:	[Procedure S5]
	arguments:	{(T Position), (D5 Position)}
Position	value:	[Procedure ConiferousTreePosition]
	arguments:	{(T Position), (D1 Position), (D2 Position), (D3 Position), (D4 Position), (D5 Position)}
Size	value:	[Procedure ConiferousTreeSize]
	arguments:	{(T Size), (D1 Size), (D2 Size), (D3 Size), (D4 Size), (D5 Size)}

Fig. 6: Simplified definition of concept ***coniferous tree.***

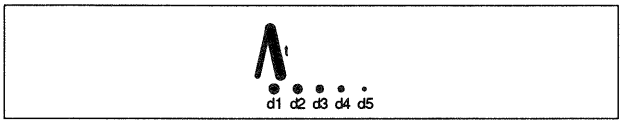

Fig. 7: Ideal shape of a coniferous tree.

For a successful instantiation of the concept *coniferous tree* shown in Fig. 7, an instance (t, d1, d2, d3, d4, d5) of the corresponding concept (*inverted V* and *dot*) has to be available for each part of the tree in a defined topology with necessary attributes. For the intensional description, the necessary parts (T, D1, D2, D3, D4, D5) are defined in the slot *necessary parts*. The single elements of the list are references to further substructures represented by slots. Each substructure owns a facette *value* and a facette *restriction*. The facette *value* contains a reference to the concept and therefore also to the instances of interest. Considering the entry of facette *restriction*, a subset of instances may be determined that is relevant for the instantiation. Thus, T, D1, D2, D3, D4 and D5 characterize lists of relevant instances.The entry of slot *structure relations* defines the structure relations that have to be satisfied for a combination of instances (t, d1, d2, d3, d4, d5) to execute a successful instantiation. The combination of instances is determined from the lists T, D1,..., D5. With regard to the example, SR1 defines the necessary topology of t and d1. For testing of structure relation SR1, facette *value* (of slot SR1) contains the corresponding procedure S1. The arguments are determined by the argument list defined by the facette *arguments*. Each element of the list represents a relational description. The relational description (T Position) for instance means that the position of t has to be transfered to the procedure S1. If a combination of instances (t_k, d_m, d_n, d_o, d_p, d_q) exists, which satisfies the structure relations, a valuation using the procedure of facette *value* located in slot *cf* is executed. The necessary arguments are determined analogously to the testing of the structure relations. For that, the argument list located in the facette *arguments* of slot *cf* is used. The instantiation is successful, if the result of valuation exceeds the threshold given by the facette *value* of slot *cf-threshold*. Subsequently, the attribute values are evaluated according to the testing of structure relations. The relevant attributes are defined by the slot *attributes* and specified by the slots *position* and *size*. In case of successful instantiation, an instance *Ix* (i.e. now x instances of the concept *coniferous tree* are existing) will be created and stored in the instance base using the data structure shown in Fig. 8. The list of instances located in facette *value* of slot *instances* will be extended by the instance *Ix*.

INSTANCE I_x

Instance Of	value:	Coniferous Tree
Necessary Parts	value:	{(T t_k), (D1 d_m), (D2 d_n), (D3 d_o), (D4 d_p), (D5 d_q)}
Structure Relations	value:	{(SR1 true), (SR2 true), (SR3 true), (SR4 true), (SR5 true)}
CF	value:	100

Fig. 8: Example for an instance I_x of the concept ***coniferous tree.***

The presented mechanism for knowledge representation is a simplified description. The following system extensions give an idea of the additional features integrated in the present system. They are necessary for professional utilization of map interpretation.

6.3 Extensions of concept definition

6.3.1 Inclusion of Topological Alternatives The concept definition introduced so far allows only an interpretation of ideal map scenes. With regard to the topology shown in Fig. 7, an instantiation is not possible if dot d3 is not present. This is contradictory to the flexible and fault tolerant human capability of reading and interpreting a map. To increase flexibility of analysis, the concept definition has been expanded. Using a disjunction of combinations of instances in the slot *necessary parts*, topological alternatives can be included. The corresponding definitions located in slots *structure relations* and *cf* have also to be expanded. Fig. 9 shows the revised concept definition

for the case of a successful instantiation even if dot d3 is absent. If instantiation of several combinations of instances is possible, the one with the largest number of necessary parts will be prefered.

```
Necessary Parts      value:   {(T, D1, D2, D3, D4, D5) OR
                               (T, D1, D2, D4, D5)}
Structure Relations  value:   {(SR1, SR2, SR3, SR4, SR5) OR
                               (SR1, SR2, SR4, SR5)}
CF                   value:   [(sr1 & sr2 & sr4 & sr5)
                                 ifFalse: [cf = 0]
                                 ifTrue: [sr3 isNil
                                             ifTrue: [cf = 100]
                                             ifFalse: [sr3
                                                         ifTrue: [cf = 100]
                                                         ifFalse: [cf = 0]]]]
                     arguments: {(SR1), (SR2), (SR3), (SR4), (SR5)}
```

Fig. 9: Section of concept definition ***coniferous tree*** *that includes topological alternatives.*

6.3.2 Definition of Recursive Structures For analysis of recursive structures like contour lines, a special concept definition exists, which is based on exactly two necessary parts. The first part describes the non-recursive basic element. With its help, the recursive structure is defined. In the case of a dashed line, this concept describes an individual line segment. The second part contains a reference to the concept itself and therefore represents the recursive structure.

6.3.3 Introduction of Constraints So far an assumption was made during instantiation process that instances of concepts acting as necessary parts of the superior concept are already existing. To obtain flexibility in image analysis it is not always necessary and desirable to make this assumption. If, for example, an instance of the concept *connection line* is necessary to process the instantiation of a concept, it is normally not possible to generate all instances of *connection line* in advance for the current map scene. Trying to do this would result in an overflow of the instance base and an unacceptable long processing time. But in general, it is not necessary to generate all instances, because only one of them is of interest and this one depends normally on the other necessary parts concerned. To solve this problem *constraints* are introduced supplementary to the *restrictions* of necessary parts. These constraints describe properties which the necessary part has to possess for successful instantiation. These properties therefore depend on the other necessary parts concerned in contrast to the properties forced by the restrictions. If constraints are used, it must be guaranteed that the necessary parts determining the desired properties are independent of the corresponding parts. Otherwise consistency of the definition is violated. To distinguish between concepts that may be instantiated in a direct manner and those that are instantiated due to constraints, an additional slot has been introduced. This slot defines the type of the concept. The possible entries in the facette *value* of slot *type* are *normal* and *goal driven*. Fig. 10 shows an example of the definition of a constraint.

In this example the instantiation of the concept *connection line* is based on the information given by the necessary parts *NP1* and *NP2*. Therefore, only solutions satisfying the constraints *C1* and *C2* are considered for instantiation. The constraint *C1*, for example, forces the start coordinate *start* given by facette *variable* of slot *C1* to be identical with the center coordinate *center* of *NP1* given by facette *reference value*. The identity of both values is enforced by the entry "=" of facette *operator* (also possible: $<$, $>$, $\geq$, $\leq$). Analogously *C2* specifies the end coordinate of the desired connection line.

Necessary Parts	value:	{(NP1, NP2, NP3)}
NP1	value:	Line Segment
	restriction:	nil
NP2	value:	Line Segment
	restriction:	nil
NP3	value:	Connection Line
	restriction:	nil
	constraints:	{C1, C2}
C1	operator:	=
	variable:	Start
	reference value:	(NP1 Center)
C2	operator:	=
	variable:	End
	reference value:	(NP2 Center)

Fig. 10: Example for the definition of a constraint.

7. CONTROL MODULE FOR KNOWLEDGE-DIRECTED IMAGE ANALYSIS

The control module supervises and controls instantiation of concepts. Two operation modes exist. In the *interactive mode* a concept of interest is given by the user as a goal. Thereupon, the control module determines the minimal set of necessary concepts at the lowest hierarchy level. Based on this set the instantiation of superior concepts is performed successively until the goal concept is reached. In the *automatic mode* the instantiation is obtained in a bottom-up manner using all instances at the lowest hierarchy level. Thus, all concepts of superior layers will be instantiated. In both modes the instantiation of a single concept is performed in accordance to the methods described in the previous sections.

Normally instantiation of a concept results in several instances. Therefore, the control module is able to handle different alternatives during analysis. This feature is important because normally a definite interpretation of a map scene requires consideration of the context of surroundings. Existence of different alternatives leads to instances that are in competition with each other. For management of the interpretation hypotheses a graph controlled by a belief-revision-algorithm (Puppe, 1987) is used. This algorithm is based on aspects of truth-maintenance-systems (Doyle, 1979; deKleer 1986a, deKleer, 1986b; Dressler, 1988; Petri, 1989). The instance that is relevant for further instantiations is selected by an evaluation algorithm. The selection depends on the type of concept. For all types the certainty factor *cf* is used. For recursive concepts the number of non-recursive basic elements is considered. In case of recursive and simultaneous goal driven concepts the constraints satisfied by the actual instance are compared to those of the underlying preceded hypothesis.

8. KNOWLEDGE-DIRECTED INTERPRETATION SUPPORT FOR HIGH COMPLEXITY MAP SCENES

Problems in map interpretation may occur if the raster image is too complex for the context-independent raster processing methods presented so far. In such cases of conflicts, the instantiation of concepts of map objects may not be possible because of lack of appropriate structure primitives. A possible reason for complexity may be the overlapping of different map symbols. For the solution of this problem a hypothesis is generated, that states which map symbol is expected in the specific image region (refer to concept *virtual continuation of a contour line* in section 6.1). Hereby, the actual situation of instantiation is the decisive criterion. Based on this hypothesis a more specific raster analysis method is used to detect the expected symbol in the corresponding color layer and image region of interest. Depending on this analysis the subsequent instan-

tiation uses the recognition result obtained in the previous step.

9. INTERFACE BETWEEN ANALYSIS SYSTEM AND GIS

The facts extracted by the map interpretation are stored in the instance base. For further use of the corresponding information the data has to be converted to data structures specified by the INFO-Database of the GIS. This conversion has to be done by the module *data conversion* (cf. Fig. 1).

10. IMPLEMENTATION AND RESULTS

Our system for knowledge-directed analysis of maps contains currently the basic ideas and methods presented here. So far the module *conflict solution*, described in section 8, is not yet implemented, however this will be done in near future. Algorithms for processing raster data are implemented using High-C (from Metaware). The knowledge-directed system is realized by the object-oriented programming language and development environment Smalltalk-80 (Goldberg et al., 1989). 486-PCs are serving as host computers.

Fig. 11 shows a test scene scanned from a topographic map of scale 1:5 000 containing the colors black and brown. Figs. 12 and 13 show the two color layers BROWN and BLACK that have been separated from the original RGB-image of Fig. 11 using the methods described in section 4.1.

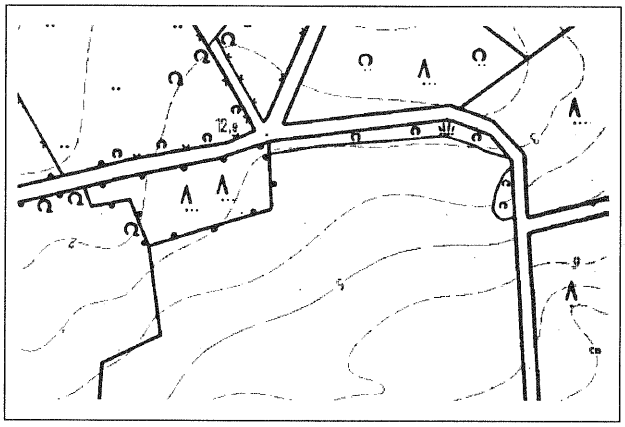

Fig. 11: Example of a topographic map scene (original size is 70x44 mm resp. 1104x700 pels).

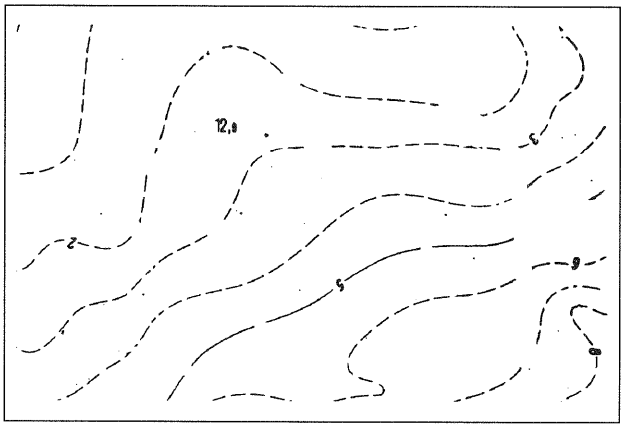

Fig. 12: Color layer BROWN separated from image shown in Fig. 11.

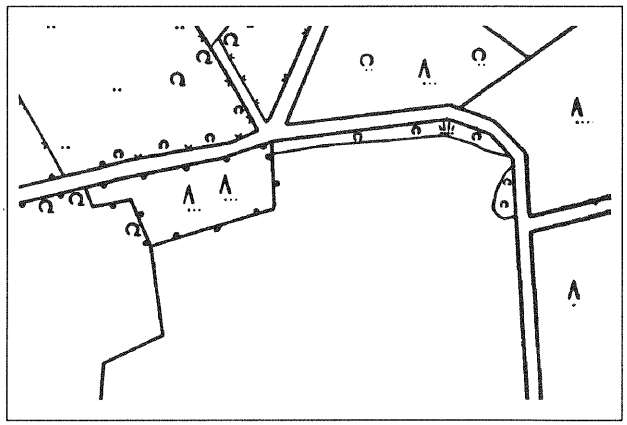

Fig. 13: Color Layer BLACK separated from image shown in Fig. 11.

For first experiments the model represents concepts for interpretation of *text symbol, coniferous tree, deciduous tree, bush, heath, road section, crossing, contour line (1m interval)* and *contour line (5m interval)*. The corresponding interpretation result for the above example is shown in Fig. 14. The image parts without marking symbols are not recognized in the present system.

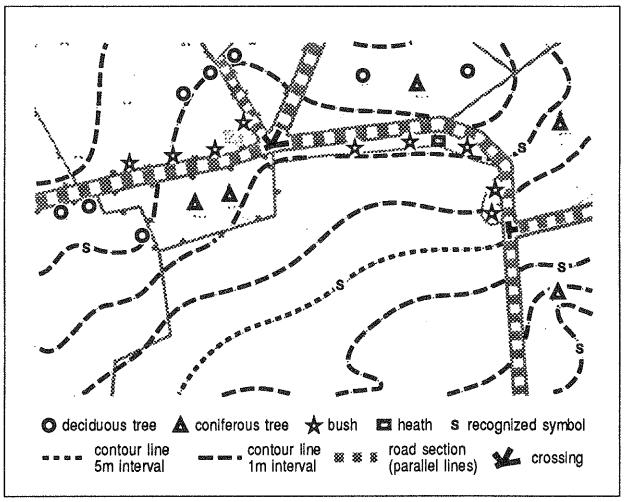

Fig. 14: Interpretation result for map shown in Fig. 11.

11. SUMMARY AND CONCLUSIONS

An overview of an image analysis system for interpretation of topographic maps was presented. Methods for raster data processing, knowledge organization and knowledge use were discussed. The main ideas of raster data processing are scanning using a 24-Bit-RGB-scanner, separation of color layers, raster symbol and raster object recognition and vectorization. The principles of knowledge-directed interpretation are those of prototypes (concepts) as the basic representation building block, generalization, and aggregation as interacting abstraction mechanism. Finally the capabilities of the system are demonstrated on a map scene. Further work will be directed towards the conflict solution module and the improvement of the already existing methods. Eventually, the system will be tested for more map scenes. Furthermore, it must be investigated how more complex raster processing steps for extraction of structure primitives (Ebi et al., 1991) may improve the overall system performance.

REFERENCES

Brachman, J.R., 1979. On the Epistemological Status of Semantic Networks. In: Findler, N.V., (ed.), 1979. Associative Networks - Representation and Use of Knowledge by Computers. Academic Press, New York, pp. 3-50.

Chaudhuri, B.B., Shankar, B.U., 1989. An Efficient Algorithm for Extrema Detection in Digital Images. Pattern Recognition Letters, 10 : 81-85.

Chu, Y.K., Suen, C.Y., 1986. An Alternate Smoothing and Stripping Algorithm for Thinning Digital Binary Patterns. Signal Processing, Elsevier Science Publishers, North-Holland, 11 : 207-222.

Data Translation, Inc., 1989. DT7910 and DT7911 RGB/IHS and IHS/RGB Converter Chips Data Sheet. Data Translation, Inc., Marlboro, pp. 6-14.

deKleer, J., 1986. An Assumption-Based TMS. Artificial Intelligence, Elsevier Science Publishers, North-Holland, 28 : 127-162.

deKleer, J., 1986. Extending the ATMS. Artificial Intelligence, Elsevier Science Publishers, North-Holland, 28 : 163-196.

Doyle, J., 1979. A Truth Maintenance System. Artificial Intelligence, Elsevier Science Publishers , North-Holland, 12 : 231-272.

Ebi, N., Besslich, P., 1991. Extraktion paralleler Linienstrukturen am Beispiel topographischer Karten. In: Radig, B., (ed.), 1991. Mustererkennung 1991, Proc. 13. DAGM-Symposium, Informatik-Fachberichte 290. Springer-Verlag, Berlin, Heidelberg, New York, pp. 312-319.

Fumiaki, T., Saburo, T., 1990. Computer Analysis of Visual Textures. Kluwer Academic Publishers, Norwell, Massachusetts, pp. 71-92.

Goldberg, A., Robson, D., 1989. Smalltalk-80: The Language. Addison-Wesley, Menlo Park.

Harmon, P., King, D., 1985. Expert Systems - Artificial Intelligence in Business. Wiley & Sons, New York, pp. 44-46.

Hayes, P.J., 1979. The Logic of Frames. In: Metzing, D., (ed.), 1979. Frame Conceptions and Text Understanding. de Gruyter, Berlin, pp. 46-61.

Lauterbach, B., Besslich, P., 1991. A Neural Network Based Recognition of Complex Two-Dimensional Objects. In: Proc. of the Internat. Conf. on Industrial Electronics, Control and Instrumentation., Kobe-Japan, IEEE Publishing Services, New York, pp. 2504-2509.

Lippmann, R.P., 1987. An Introduction to Computing with Neural Nets. IEEE Transactions on Acoustics, Speech and Signal Processing Magazine, April : 4-22.

Minsky, M., 1979. A Framework for Representing Knowledge. In: Metzing, D., (ed.), 1979. Frame Conceptions and Text Understanding. de Gruyter, Berlin, pp. 1-25.

Pavlidis, T., 1987. Algorithms for Graphics and Image Processing. Computer Science Press, Rockville, pp. 142-214.

Petri, C., 1989. Reason Maintenance in Expert Systems. In: KI - Künstliche Intelligenz, Oldenbourg-Verlag, München, 2 : 54-60.

Puppe, F., 1987. Belief Revision in Diagnosis. In: Morik, K., (ed.), 1987. GWAI-87 - Proc. of. 11th German Workshop on Artificial Intelligence. Springer-Verlag, Berlin, Heidelberg, New York, Tokyo, pp. 175-184.

Quillian, M.R., 1968. Semantic Memory. In: Minsky, M., (ed.), 1968. Semantic Information Processing. MIT Press, Cambridge and London, pp. 227-270.

Rich, E., 1983. Artificial Intelligence. McGraw-Hill, Berkeley, pp. 229-233.

Richards, J.A., 1986. Remote Sensing - Digital Image Analysis. Springer Verlag, Berlin, pp. 190-205.

Sagerer, G., 1985. Darstellung und Nutzung von Expertenwissen für ein Bildanalysesystem. Springer-Verlag, Berlin, Heidelberg, New York, Tokyo.

Schoppmeyer, J., 1991. Farbreproduktion in der Kartographie und ihre theoretischen Grundlagen. Polygraph Verlag, Frankfurt, pp. 42-74.

Springub, A., Scheppelmann, D., Engelmann, U., Meinzner, H.P., 1990. Generierung von optimalen Falschfarben aus Echtfarbbildern mit Hilfe der Topologischen Karte. In: Großkopf, R.E., (ed.), 1990. Mustererkennung 1990, Proc. 12. DAGM-Symposium, Informatik-Fachberichte 254. Springer-Verlag, Berlin, Heidelberg, New York, pp. 267-275.